International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-2, December 2018

The Real Problem Through a Selection Making an Algorithm that Minimizes the Computational Complexity

M. Sivaram, D. Yuvaraj, Amin Salih Mohammed, V. Porkodi, V. Manikandan

Abstract: ***Computational complexity issues are gaining increasing attention as information search, retrieval and extraction techniques and methodologies mature. This paper presents the theoretical approach of computational complexity reduction which supports information related operations in an effective and efficient manner. Three-step approach to design paradigm will have different costs (time), will consume different amount of resource (space) and will have some inherent risk or uncertainty. To capture these aspects, we need economic models of software that take into account costs, space, uncertainty and schedule implications. To address this need for economic decision making, we have proposed a method of economic modeling of information-related operations centered on an analysis of their architecture. Although we consider that the computation problem belongs to the class of np-complete. In this paper, our objective is to solve the real problem by an decision making algorithm that minimizes the computational complexity.***

Keywords: Hierarchical Modeling Methodology, Partitioning,

I. INTRODUCTION

Highlight The focus of design effort on higher levels of abstraction, driven by increasing system complexity has led to the need for a efficient and effective algorithm design decision making. We can isolate three different tasks. First, we must specify algorithm's functionality and constraints. Second, we must explore various hierarchical modeling methodological alternatives, each consisting of an interconnection of procedural instructions and an assignment of functionality to them. Third, we must refine the original specification into a new backtracking description, which the algorithm focuses to create an historical/backpropagation review strategy of users for frequent accessing.

In current practice, these three steps are carried out in an informal and ad-hoc manner specification are usually written informally in English or some other natural language. Specification may be search string, webpage, image and audio-video clips. Exploration is done using mental or hand calculated estimation of quality metrics such as performance, size and computational complexity issues. The refined backpropagation and historical review process is then created informally using block diagrams. Drawbacks of such informal techniques may include the lack of early access, the lack of rapid feedback of quality metrics that result from design decisions, the lack of information-related operation tools to explore more design alternatives while requiring less design time and preserving the determinism whenever possible. A decision-making algorithm is said to be deterministic if a given sequence of inputs always produces the same sequence of outputs. The implementation of a complex algorithm is globally nondeterministic, most part of it are individually deterministic. Since deterministic programs are much simpler to analyze and debug that non-deterministic ones. An operation driven hierarchy is based on operations specified by users, experts, or the data mining system. Operations can include the decoding of information encoded strings, information extraction from complex data objects, and data clustering alternatively, mathematical and statistical operations, such as data clustering and data distribution analysis, tools, algorithms, can be used to form concept driven hierarchies. This article not only proposed a rate analysis framework but also presents its underlying algorithm design decision making, theoretical basis, and design issues. The main contributions can be summarized as follows.

A two-level system model in which the top level models the processes in the system using a page graph and the bottom level models each page using a sequencing graph We introduce the concept of an enable signal to capture the synchronization between pages. This abstraction allows us to model systems with sequencing pages. The two-level model is

useful in developing efficient algorithms for rate analysis[1].

The notion of average access rate of a page, defined as the inverse of the asymptotic mean of the sequence of time intervals between successive invocations of the pages. We present a purely graph-theoretic proof showing that the average access rate is well defined for all pages in a finite page graph. Irrespective of the initial start times of the page set. This proof also gives more accurate bounds on the periodicity of inter-access times for a page than previously reported in the literature.

Manuscript published on 30 December 2018.
***** Correspondence Author (s)
Dr. M. Sivaram, Assistant Professor, Department of Computer Networking, Lebanese French University, Erbil, Kurdistan Region, Iraq
Dr. D. Yuvaraj, Lecturer, Department of Computer Science, Cihan University/ Duhok, Kurdistan Region, Iraq.
Dr. Amin Salih Mohammed, Dean, College of Engineering and Computer Science, Lebanese French University, Erbil, Kurdistan Region, Iraq.
Mrs. V. Porkodi, Assistant Professor, Department of Information Technology, Lebanese French University, Erbil, Kurdistan Region, Iraq.
Mr. M. Manikandan, Assistant Professor, Department of Information Technology, Lebanese French University, Erbil, Kurdistan Region, Iraq.

We present efficient algorithms for computing bounds on the average access rate of pages in a page graph. The rate analysis framework for using the bounds on process execution rates to interactively modify the design of an efficient algorithm to satisfy all the rate constraints.

The rest of the article is organized as follows. Section II describes our two-level system model,

focusing mainly on the page graph and the modeling of synchronization between pages. Section III gives an overview of the rate analysis framework, defines the average access rate, and the main problems addressed in this article. Each of the remaining sections examines one of these problems, and Section IV show that the average access rate is well defined and presents efficient algorithms for computing bounds on the average access rate of page set in a page graph. Section V discusses conclusion and future enhancement[4,5].

II. THE ALGORITHM DESIGN MODEL

We model an algorithm as a set of concurrent interacting pages. Our model is a hierarchical two-level mode. The top level is the page graph that captures the interaction between the pages. Each page is modeled using a sequencing graph that represents the data and control flow dependencies between the operations within a set of pages. The sequencing graphs from the bottom level in our two-level representation.

In the page graph Gp (Vp , Ep), each vertex Pi in Vp represents a page. An edge (Pi, Pj) indicates that the page Pi enables the execution of page Pj by sending an enable signal to process Pj. Each edge (Pi, Pj) in the page graph is associated with a delay interval $\Delta ij = (dij, Dij)$ that bounds the time after the initiation of an accessing of page Pi when Pj receives the enable signal from Pi. Let $dij \leq \delta ij \leq Dij$. Fig 1 shows a simple process graph with two-page set.

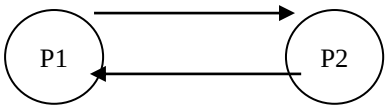

Fig 1: A Page Graph with Two-Page Set.

Each page set represents an independent thread of control. We assume that all the pages are concurrently active. The accessing semantics of our model is as follows. Let Xi (k) denote the time at which page Pi starts its (K+1)st starts access. In particular, Xi (0) is the initial start time of Pi. Then the following rules govern the subsequent access of the pages[2].

Independence

Initially, each page Pi starts its first execution at Xi(0) independently of other pages.

AND casuality

A new instance of page Pj starts executing after all its predecessors in the page issued enable signals for Pj.

Consequently,

$Xi(k)=\max Pj\in Pred(Pi) \{Xj(k-1) +\delta ij\}$

Or, since $\delta ij\in \{dij, Dij\}$

$\max Pj\in Pred(Pi)\{Xj(k-1)+\delta ij\}\leq xi(k)\leq \max Pj\in Pred(Pi)\{Xj(k-1)+Dj\}$ where K>0 and Pred (Pi) is the set of predecessors of Pi is the page graph[3].

2.1. Highlights

Our page graph model and the associated enable signal-based communications/synchronization is general enough to represent most decision making algorithm specifications. Here are some of the main highlights of our model.

Ability to model several different page synchronization mechanism

The basic primitive for synchronization in our model is an enable signal from one page to another. A page starts accessed once it has received an enable signal from all its predecessors in the page graph. This allows us to model an algorithm with several different kinds of synchronization and communication paradigms in a unified manner. Some examples are

(1). Operation-Based Enable Signal

Such an enable signal is issued by a page when it executes a particular operation in its sequencing graph. A vertex in the sequencing graph whose execution results in the generation of an enable signal is called an enabling vertex. The delay between the start of the access of a page and the generation of an operation-based enable signal can vary for different accesses of the pages because it may be value dependent.

(2) Time-Based Enable Signal.

Such an enable signal is generated after a fixed time following the start of the access of a page. It is not associated with the execution of a particular operation in the page. Such enable signals are useful in modeling a sequence of pages for which a new access starts after a fixed delay.

(3) Modeling the Streamlining of Pages

A new instance of a sequence of pages can start accessed before its previous access has finished. Our page graph model can handle such sequence of pages using a self-loop with delay less than the access time of the age. Thus, the page graph model allows us to model multiple parallel and sequenced access instances of a page set. If the delay associated with the self-loop is larger than the access time of the page, then new access of the page cannot begin until the previous access has terminated. Consequently, the page cannot be sequenced.

2.2 Limitations

In order to develop efficient algorithms for rate analysis, we need to impose some restrictions on the way enable signals can be issued by pages. These are not inherent to our page graph model but are limitations introduced due to the algorithmic techniques we use for rate analysis. Basically, these limitations arise due to the requirement that one access of a page should generate exactly one enable signal for each of its successors in the page graph. This results in the following restrictions on the generation of enable signals.

Enable signals should not be generated from inside loops in the sequencing graph.

Since loop bodies can be accessed multiple times for one access of a page, if there are enable vertices in a loop body then multiple enable signals would generated for the same successor page in one execution. Rate analysis for such systems is an interesting problem that would require new techniques.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Enable signals should not be generated from conditionally executed parts of the sequencing graph for a page

The presence of enable vertices in conditional branches would imply that when the conditional branch is not executed no enable signal would be generated for some successor page. At first glance, this appears to be a severe restriction on the expressive power of our page graph model. However, any deadlock free, strongly connected page graph that user own synchronization semantics must have this property. The possibility of deadlock in the presence of conditional enable signals is illustrated in the following example. It is possible to have conditional enable signals without causing deadlock if they span two different strongly connected components in the page graph.

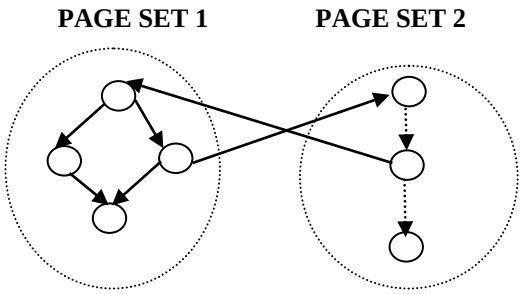

Fig 2 Two Pages (with the Corresponding Sequencing Graphs Shown in the Dotted Regions). Page P1 issues a Conditional Enable Signal for page P2.

Example 2.1

Consider the figure 2. Page P1 issues an enable signal for page P2 from an enabling vertex in the FALSE branch of a conditional branch and page P2 issues an unconditional enable signal for page P1. If the computation ever takes the TRUE branch in page P1 , no enable signal is issued for P2. Thus P2 is not invoked, and consequently, P1 is not invoked , leading to a deadlock. The same scenario can occur in any strongly connected page graph that has enable vertices in conditional branches.

III. RATE ANALYSIS / COMPUTATIONAL COMPLEXITY ANALYSIS FRAMEWORK

In this section, we define average execution rate, give a precise formulation of the problems that are addressed in the article, and then give a a brief overview of our rate analysis/ computational complexity analysis framework.

3.1. Average Access Rate

The rate of access of page is defined as the number of access of the pages per unit time. Let

Xi(k)= time at which page Pi starts executing for the (K+1)the time .

We are concerned with algorithm that exhibit non-deterministic behavior. Such systems behave in a cyclic fashion in steady state after the initial transients. The following limit has been used by researchers as a standard way of summarizing the performance of a page access in such a system . we use the same definition and define the average inter-access time Ti and the average execution rate ri of a page Pi as

If the preceding limit exists . we show in Example 3.1 that for a finite page graph with finite edge delays, the preceding limit exists and can be efficiently computed. Thus, in this article, we focus on asymptotic rates of execution. Note that the time is not constant; hence the instantaneous execution rate of a page (Interexecution /access) time is not constant; hence the instantaneous execution rate of a page keeps changing. The following example illustrates our definition of average rate of execution.

Example 3.1 consider the page graph shown in Figure 1, Assuming that the page start accessing at time 0 and the edge delays equal their lower bounds ($\delta 12=1, \delta 21=2$)

Page P1: 0 2 3 5 6 8
Page P2: 0 1 3 4 6 7

Thus the sequence of inter accessing times for both the pages are

Page P1 : 2 1 2 1
Page P2 : 1 2 1 2

and the average inter accession time is 3/2. Thus, using the definition of average rate of execution given previously, the execution rates of both pages in this example are 2/3. Notice that the limit in our definition of the average rate of implementation exists in this case because both sequences of inter-access times are periodic (with period 2)

3.2 Rate Constraints

We assume that the designer specifies upper and lower bound on the average execution rates of all the page set. Thus, each page Pi in the page graph is associated with a constant interval Ii=[Li, Ui]. Constraint intervals for page set in a page graph can be arbitrary (since they are specified independently for each page by the designer). If there is no lower bound constraint on the rate of a page then Li=0; if there is no upper bound constraint then Ui=infinity. For the specified rate constraints to be satisfied, the computed execution rate for each page set must lie within its constraint interval; that is

$Li \le ri \le Ui$

Usually, the environment in which the algorithm works imposes rate constraints on various page set of the algorithm. So the rate at which certain events happen in the environment place execution rate constraints on the page set.

3.3 Problem Formulation

Given a description of an algorithm (page graph and the sequencing graphs of the page set) , along with the associated rate constraints, the following problems need to be addressed in the rate analysis framework.

Delay analysis

Compute bounds on the accessing times of operation in the sequencing graphs, and use these estimates to find bounds on the delays of the edges in the page graph. Delay analysis requires computations of bounds on the time after the initiation of a page when it issues an enable signal for another page and the estimation of the communication delay between the pages generating the enable signal the one receiving it.

Consistency checking of rate constraints

This problem arises because the designer usually specifies the rate constraints independently for each page.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

A set of rate constraints is said to be inconsistent if they cannot be satisfied for a given page graph topology, irrespective of the delay intervals on the page graph edges. Thus, if a set of rate constraints is inconsistent, the computed rate intervals can never satisfy independent of the actual delay intervals in the page graph, we can state the consistency checking problem as follows: Given a page graph and the rate constraint intervals associated with each page in the page graph, are the constraint intervals constraint?

Section V describes our consistency checking algorithm.

3.4 Rate Analysis

The rate analysis problem is stated as follows. Given a description of an algorithm as a page graphand the associated sequencing graphs, find upper and lower bounds on the average access rate of each page in the page graph. So, rate analysis finds an intervfal [rl (Pi), ru(Pi)] for each page Pi, such that the average rate of access of the page is guaranteed to lie in this interval. The rate constraints on Pi are said to be satisfied if the rate interval computed by rate analysis is constrained in [Li, Ui] . If the computed rate interval is not contained in [Li, Ui], then one or both of the rate constraints of Pi are violated. In section IV we present our algorithm for rate analysis.

3.5 Rate Constraint Debugging and Process Redesign

If rate analysis finds that some of the rate constraints are violated, the designer needs to redesign specific page set and change the page graph topology by altering the manner in which the page set communicate. The rate analysis tools needs to give adequate information about the cause of the rate constraint violation to help in this step. We discuss possible approaches in section V. Figure 4 shows our framework for rate analysis. The user specification is first translated from a high-level description to a page graph along with the associated sequencing graphs. First, the rate constraints are checked for consistency. If the rate constraints are consistent, rate analysis is performed on the pate graph. If all the computed rate intervals are contained in the intervals defined by the rate constraints, then the system satisfied all the rate constraints, and no redesign is required. If a rate constraint is violated, the tool gives the user the reasons for the violation and this information can be used to redesign the relevant pages. Although consistency analysis of the rate constraints precedes rate analysis in the flow, we discuss rate analysis before consistency analysis in this article. This is motivated by the fact that many of the results required for describing our consistency-checking algorithm are more natural to discuss in the context of rate analysis.

IV. RATE ANALYSIS

In this section, we show that our notion of average execution rate is well defined and present algorithms for rate analysis of page graphs. For the sake of clarity, some of our results are stated for strongly connected page graphs. However, we do generalize our results to arbitrary page graphs, and there is no limitation on the topology of the page graphs that our rate analysis algorithms can handle.

4.1. Existence of Average Access Rate

In order to prove that our asymptotic definition of average access rate is well defined, we first examine the case when the page graph is strongly connected (i.e., there is a path from each process Pi to all other processes in the graph) and there is a unique delay δij associated with each edge in the page graph. The extension to the case when the page graph has several strongly connected components, and the page graph edges have delay intervals associated with them becomes clear when we discuss the algorithms for rate analysis for such page graphs in section 4.2

We define the delay of a cycle C in the page graph as

EQUATION [3]

The number of edges in a cycle C is denoted by | C |. The mean delay of a cycle C is given by

$$\frac{d(c)}{|C|}.$$

The maximum mean cycle delay is denoted by lemda. A cycle is said to be critical if it has the maximum mean delay among all the cycles in the graph. The following theorem establishes that our definition of execution rate in equation 2 is well defined, and can be related to the maximum mean cycle in the page graph. Weaker forms of this theorem can be bound in references.

THEOREM 1. Consider a strongly connected page graph and let Xi(k), k N GREATER THAN OR EQULA TO 0 be the sequence of time instances at which page Pi executes. Then there exists some N GREATER THAN OR EQULA TO 0.Such that the following are true.

(1) The sequence Xi(k) –Xi(k-1) , k N GREATER THAN OR EQULA TO N is periodic with period equal to L , where L is equal to the least common multiple of the lengths of all the critical cycles in the age graph.

(2) For any Q N GREATER THAN OR EQULA TO N,

EQUATION

Where lemda is the maximum mean cycle delay in the page graph.

(3) The average interaccess time is well defined and

EQUATION

The proof of the theorem is based on the characterization of critical paths in the page graph. The main idea used in the proof is that Xi(k) (the time when the (k+1)th execution of page Pi starts) is determined by the delay of thelongest path in the page graph with exactly k edges that ends at page Pi, and that this path can be characterized using the maximum mean delay cycles in the page graph.

Notice that Ti is not average in the probabilistic sense. As this theorem shows, the inter-access time of a page in a page graph may not stay constant but a sequence of its successive interaccess times repeats itself with a certain period, and Ti is the mean of the interaccess times within such a sequence.

4.2. Algorithms for Rate Analysis

Theorem 1 shows that for a page graph with one strongly connected component (SCC) if all the edge delays are exact, then the access rate of all the page set is equal to the inverse of the maximum mean cycle delay in the page graph. In this section, we first show how to computer upper bound and lower bounds on execution rates of page set in the presence of uncertainty in the edge delays.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Then, we give an efficient algorithm for computing execution rates for the case when the page graph is strongly connected (or, equivalently, has one SCC) and, finally,

We extend our algorithms to the case when the page graph has multiple SCCs. This case handles any page graph.

Handling delay intervals. We assume that we have a process graph Gp with a delay interval associated with each edge. The following theorem allows us to compute the rate interval [rl (Pi), ru(Pi)] for each page Pi in the page graph from the delay intervals of the edges.

Theorem 2. (Monotonicity theorem). Increase (decrease) in the delay of an edge cannot increase (decrease) the average access rate of any page.

PROOF. From Theorem 1 , we know that the average access rate of a page in a strongly connected page graph is equal to the inverse of the maximum mean cyucle delay in the page graph. Increasing (decreasing) the delay of an edge in the page graph cannot increase(decrease) the average access rate of any page. From the monotonicity theorem, it follows that we can compute ru(X) by setting all the edge delays to their lower bounds and rl(X) by setting all the edge delays to their upper bounds. Hence , if the page graph is strongly connected then all the nodes have the same rate interval, where are the maximum mean cycle delays of the page graph computed by setting the edge delays of all the edges to their lower bounds andupper bounds, respectively. Hence, the rate interval for a stongly connected page graph is computed by two invocations of the algorithm for computing the maximum mean delay cycle in a graph. Notice that the lower and upper rate bounds are the tightest bounds on the average access rate but they do not bound the best and worst case access rates. If the latter are desired , our rate analysis framework would still apply; however, the average inter-access time, which is used to compute the average access rate, should be replaced with the best and worst case interaccess times. Loose bounds on them are simply the minimum and maximum arc delays in the page graph. The tightest bounds on them are given by the time separation algorithm.

Rate analysis: Single SCC Using the result in Theorem 1, we need to comute the maximum mean cycle delay tocompute the average access rate of page set in a SCC. The following theorem due to karp allows the design of an efficient algorithm for finding the maximum mean cycle delay.

Theorem 3.

Consider a strongly connected graph G(V,E) with n nodes.Let s be an arbitrary chosen vertex. For every v E V, and every nonnegative integer k, define Dk(v) as the maximum delay of a path of length k form s to v; if no suchpath exists, then Dk(v)= -oo , Then the maximum cycle mean lemda of G is given by Equation

Thus, for a stongly connected process graph, the average execution rates of all the page set are equalto the inverse of the maximum mean cycle delay, and they can be computed using Karp's characterization of the maximum mean cycle delay in O(|V|.|E|) time.

Rate Analysis: Multiple SCCs.

We now consider a page graph that has several strongly connected components. We define the component graph of the page graph to be the graph in which there is a vertex for each SCC and an edge from you to v if and only if there is an edge from a vertex in the SCC represented by you to a vertex in the SCC represented by V in the page graph. Note that the component graph is a directed acyclic graph. The following is the key observation on which the algorithm for computing rate intervals for a page graph with multiple SCCs hinges.

RATE _ANALYSIS(G)
Find the maximal SCCs in G
For each SCC, C
Construct the component graph: Gc(Vc, Ec)
END RATE_ANALYSIS

Fig 4 . Algorithm for rate analysis for a page graph with multiple SCCs; RATE is the algorithm for computing aa using Karp's characterisation.

THEOREM 4.

Let P and C be two maximal strongly connected components in a page graph with some edges from vertices in P to vertices in C (so P is a producer and C is a “consumer”). Let [rl{P), ru(P0] and [rl’ ©, ru’(C)] be the rate intervals for Pand C , respectively, computed assuming that there are no edges between P and C. Then the actual rate interval for C is [rl(C),

ru (C)], where

EQUATION

PROOF. Any cycle in P has path to any process Pi € C through edges connecting P to C. Consequently, the k critical paths for Pi (for sufficiently large values o k) are determined by the larger of the maximum mean delay cycle in P and C (refer to the theorem 1) . Hence, the average access rate of any process in C is the minimum of the rate computed assuming that there areno dependencies from P to C (this is the inverse of the maximummean cycle delay in C) and the average access rate of any page in P (this is the inverse of the maximum mean cycle dealy in P) . These observations along with the monotonicity theorem prove this theorem. Using the preceding theorem, we develop the algorithm in figure 5 for rate analysis of a page graph with multiple SCCs. The following example illustrates the steps involved in the rate analysis for page graphs. Example 4.1. Consider the page graph shown in fig 5. The delay intervals for the edges are shown. Notice that we do not associate a delay interval with the edge connecting the two strongly connected components of the graph. This is because such inter-SCC edges are not part of any cycles in the page graph, and consequently, their delay does not affect any execution rates. We now consider the steps involved in the rate analysis of the page graph in figure 5.

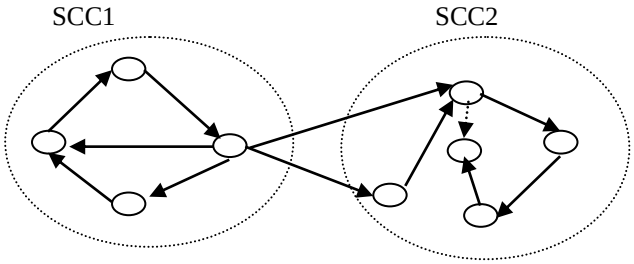

FIG 5. Page Graph used in Example 4.1

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

For SCC1:

for computing rl, set alle edge delays to their upper bounds.

= maximum mean delay cycle in SCC1

=14.67

The critical cycle is (p1 • p2 • p4 • p1) and

Rl =(14.67)

=0.068.

For computing ru, we use the lower bounds on the edge delays.

Maximum mean delay cycle in SCC1

=max {7, 4.75}

=7

Thus, ru= (7) =0.12.

Hence the rate interval for SCC1 IS [0.068,0.142].

For SCC2:

So, the rate interval for SCC2 IS [0.094,.231]. Notice that for clarity of exposition, we have computed the maximum mean ycle delay sing explicit enumeration f the cycles in this example. In our implementation instead of explicit cycle eneumeration, we use karp's characcteriazat8ionjof maximum mean cycle delay to compute the rates.

V. CONSISTENCY CHECKING

A set of rate constraints is said to be inconsistent if they cannot be simultaneously sagtisfied for a given page graph topology, irrespective of the delay intervals on the page graph edges. Thus, if a set of rate constraints is inconsistent, the computed rate intervals can never satisfy all the rate constraints. Note that a consistent set of constraints may not be contained in the corresponding constraint intervals. Theorem 4 gives sufficient conditions for a set of rate constraints to be inconsistent. Before proceeding further with the consistency analysis , we need a few definitions, Consider n rate constraint intervals in the form of Ii = [Li, Ui] for 1 i n .

Their minimumis mini Their intersection isGiven intervals i

Now consider a strongly connected component in the page graph Cj. Then the following hold.

The intersection of the rate intervals of the page set in Cj is The minimum of all the intersections that belong to Cj and alof its predecessors is [5] and for a component with no predecessors, More intuiti9vely,l and for component with no predecessors, intermediate hop in different physical group B so that the tree message is sent to the source through the source of the Physical group B, that results in a set of transitions called a Hamilton cycle. Then the multicast algorithm is NP-Complete. If there are n nodes in the physical group A and B, then there are{ N[N+1}/2 } edges for a multicast member with a self loop.

VI. CONCLUSION

We have shown a prototype of a multicast algorithm with a Turing machine. An efficient runtime multicasting algorithm is needed for high performance up gradation in order to resolve the conflict between source and receiver. We further intend to carry out our research in multiple multicast group members with source node. Tracking and refining the estimated source to multiple receivers provides a feasible solution through a Hamiltonian cycle.

REFERENCES

1. DouglasC.Schmidt, "An overview of IP Multicasting" ,Unix Network Programming.
2. Ian Ferrel, Adrian Mettler, Edward Miller, and Ran Libeskind-Hadas, Virtual Topologies for Multicasting With Multiple Originators in WDM Networks, IEEE/ACM Transactions on Networking, Vol. 14, NO. 1, February 2006
3. Padmini Vellore,Paul Gillard, Ramachandran,Venkatesan, Delivery Analysis of Multicasting in BitTorrent Enabled AdHoc Network (MBEAN) Routing, IWCMC'06, July 3–6, 2006, Vancouver, British Columbia, Canada.
4. K.Batri, M. Sivaram "Testing the Impact of Odd and Even Point Crossover of Genetic Algorithm Over the Data Fusion in Information Retrieval" Volume 74, Issue Number 4, Pages643-649,European Journal of Scientific Research,2012.
5. Dhivakar B, Saravanan SV, Sivaram M, Krishnan RA. Statistical Score Calculation of Information Retrieval Systems using Data Fusion Technique. Computer Science and Engineering. 2012;2(5):43-5.