ICSNC 2016

The Eleventh International Conference on Systems and Networks Communications

ISBN: 978-1-61208-499-2

August 21 - 25, 2016

Rome, Italy

ICSNC 2016 Editors

Eugen Borcoci, University Politehnica of Bucarest, Romania

Carlos Becker Westphall, Federal University of Santa Catarina, Brazil

ICSNC 2016

Forward

The Eleventh International Conference on Systems and Networks Communications (ICSNC 2016), held on August 21 - 25, 2016 in Rome, Italy, continued a series of events covering a broad spectrum of systems and networks related topics.

As a multi-track event, ICSNC 2016 served as a forum for researchers from the academia and the industry, professionals, standard developers, policy makers and practitioners to exchange ideas. The conference covered fundamentals on wireless, high-speed, mobile and Ad hoc networks, security, policy based systems and education systems. Topics targeted design, implementation, testing, use cases, tools, and lessons learnt for such networks and systems

The conference had the following tracks:

- TRENDS: Advanced features
- WINET: Wireless networks
- HSNET: High speed networks
- SENET: Sensor networks
- MHNET: Mobile and Ad hoc networks
- AP2PS: Advances in P2P Systems
- MESH: Advances in Mesh Networks
- VENET: Vehicular networks
- RFID: Radio-frequency identification systems
- SESYS: Security systems
- MCSYS: Multimedia communications systems
- POSYS: Policy-based systems
- PESYS: Pervasive education system

We welcomed technical papers presenting research and practical results, position papers addressing the pros and cons of specific proposals, such as those being discussed in the standard forums or in industry consortiums, survey papers addressing the key problems and solutions on any of the above topics, short papers on work in progress, and panel proposals.

We take here the opportunity to warmly thank all the members of the ICSNC 2016 technical program committee as well as the numerous reviewers. The creation of such a broad and high quality conference program would not have been possible without their involvement. We also kindly thank all the authors that dedicated much of their time and efforts to contribute to the ICSNC 2016. We truly believe that thanks to all these efforts, the final conference program consists of top quality contributions.

This event could also not have been a reality without the support of many individuals, organizations and sponsors. We also gratefully thank the members of the ICSNC 2016 organizing committee for their help in handling the logistics and for their work that is making this professional meeting a success. We gratefully appreciate to the technical program committee co-chairs that contributed to identify the appropriate groups to submit contributions.

We hope the ICSNC 2016 was a successful international forum for the exchange of ideas and results between academia and industry and to promote further progress in networking and systems communications research. We also hope Rome provided a pleasant environment during the conference and everyone saved some time for exploring this beautiful historic city.

ICSNC 2016 Advisory Chairs

Eugen Borcoci, University Politehnica of Bucarest, Romania
Sathiamoorthy Manoharan, University of Auckland, New Zealand
Leon Reznik, Rochester Institute of Technology, USA
Masashi Sugano, Osaka Prefecture University, Japan
Zoubir Mammeri, IRIT, France
Xavier Hesselbach, UPC, Spain
Svetlana Boudko, Norsk Regnesentral, Norway
Ben Lee, Oregon State University, USA

ICSNC 2016 Research Institute Liaison Chairs

Song Lin, Yahoo! Labs / Yahoo Inc. - Sunnyvale, USA
Habtamu Abie, Norwegian Computing Center - Oslo, Norway

ICSNC 2016 Industry/Research Chairs

Rolf Oppliger, eSECURITY Technologies - Guemligen, Switzerland
Jeffrey Abell, General Motors Corporation, USA
Christopher Nguyen, Intel Corp., USA
Javier Ibanez-Guzman, RENAULT S.A.S. / Technocentre RENAULT - Guyancourt, France

ICSNC 2016 Special Area Chairs

Mobility / vehicular
Maode Ma, Nanyang Technology University, Singapore

Pervasive education
Maiga Chang, Athabasca University, Canada

ICSNC 2016

Committee

ICSNC Advisory Committee

Eugen Borcoci, University Politehnica of Bucarest, Romania
Sathiamoorthy Manoharan, University of Auckland, New Zealand
Leon Reznik, Rochester Institute of Technology, USA
Masashi Sugano, Osaka Prefecture University, Japan
Zoubir Mammeri, IRIT, France
Xavier Hesselbach, UPC, Spain
Svetlana Boudko, Norsk Regnesentral, Norway
Ben Lee, Oregon State University, USA

ICSNC 2016 Research Institute Liaison Chairs

Song Lin, Yahoo! Labs / Yahoo Inc. - Sunnyvale, USA
Habtamu Abie, Norwegian Computing Center - Oslo, Norway

ICSNC 2016 Industry/Research Chairs

Rolf Oppliger, eSECURITY Technologies - Guemligen, Switzerland
Jeffrey Abell, General Motors Corporation, USA
Christopher Nguyen, Intel Corp., USA
Javier Ibanez-Guzman, RENAULT S.A.S. / Technocentre RENAULT - Guyancourt, France

ICSNC 2016 Special Area Chairs

Mobility / vehicular
Maode Ma, Nanyang Technology University, Singapore

Pervasive education
Maiga Chang, Athabasca University, Canada

ICSNC 2016 Technical Program Committee

Habtamu Abie, Norwegian Computing Center - Oslo, Norway
M. Ilhan Akbas, University of Central Florida, USA
Fakhrul Alam, Massey University, New Zealand
Jose M. Alcaraz Calero, University of the West of Scotland, UK
Pedro Alexandre S. Gonçalves, Escola Superior de Tecnologia e Gestão de Águeda, Lisbon
Mikulas Alexik, University of Zilina, Slovak Republic
Abdul Alim, Imperial College London, UK
Shin'ichi Arakawa, Osaka University, Japan

Copyright Information

Table of Contents

Modeling IP Telephony Call with Simple Algebraic Relations

Ilija Basicevic and Miroslav Popovic
Department of Computer Engineering,
Faculty of Tech. Sciences
University of Novi Sad
Novi Sad, Serbia
email: ilibas@uns.ac.rs

Djordje Saric
RT-RK Computer Based Systems
Novi Sad, Serbia
email: djordje.saric@rt-rk.com

***Abstract*—During the evolution of telephony, many models have been proposed. However the models usually consider only certain aspects of the telephony network. This paper presents an approach to the use of algebraic relations in mathematical modeling of IP telephony networks. Three algebraic relations that represent the state of the network are defined. An interesting property of telephone network is demonstrated, that a telephony session or "call" can be interpreted as the class of equivalence of one of relations. Session life cycle is explained in the view of defined relations and the time dimension of events is briefly discussed. The relations presented here are foundations for a more comprehensive and usable model. The directions for future work are given.**

Keywords- IP telephony; telephone call; VoIP session; Session Initiation Protocol (SIP).

I. INTRODUCTION

Traditional telephony networks have been extensively researched in the period of more than a century. Majority of researches consider formal modeling of only certain aspects of the network functionality. The telephone exchange has been modeled using queuing theory and different parameters, such as call waiting time, probability of blocking, etc. that are estimated.

The IP telephony appeared in the last decade of the XX century. Two most important signaling protocols for IP telephony are Session Initiation Protocol (SIP) [1] and H.323 [2]. Again, queuing theory is used for estimating parameters such as end-to-end Voice-over-IP (VoIP) connection delay, see [3] for a recent work. In this paper, the approach to model mathematically several most important elements of the functionality of the IP telephony network is presented. The model is proposed having in mind the SIP protocol, as it is dominant today.

In the majority of available public documents in the telephony area, the call is not defined, but it is assumed that a reader already knows what the call is. Starting from the structure of the telephony network, and analyzing some aspects of network dynamics, we will try to shed new light on the concept of a telephone call.

The foundation of the telephony network is the set of network nodes (N), which are either endpoints (EP) available to users, or infrastructure points (IP) that make the infrastructure of the network. There are several types of IP:

- routing points (RP)
- conference points (CP). Those are conference servers and media mixers used for the support of conference feature with respect to signaling and media streaming, respectively.

Thus,

$$IP = RP \cup CP, \tag{1}$$

$$EP \cap IP = \varnothing,$$

$$N = EP \cup IP.$$

Nodes are connected by links. The set of links can be defined as

$$L=\{(x,y)|(x,y) \in EPxIP \vee (x,y)\in IPxIP \vee (x,y) \in IPxEP\}, \tag{2}$$

L is a symmetric relation – we consider links that ensure bidirectional communication.

The mapping of the structure of a so-called pre-IMS IP telephony network (still much in use today) to this model is straight forward. On the other hand, the IP telephony network with the IP multimedia Subsystem IMS [4] environment has richer structure with a larger number of different functional elements, but their mapping to RP and CP is possible. The fact that Call Session Control Function (CSCF) elements are realized as SIP servers makes the mapping straight forward.

The traditional telephony is limited with the fact that calls cannot be routed based only on signaling information while call processing applications require access to media stream, as well. This results in so called “tromboning issue” that has been solved with the advent of VoIP in which the signaling and media stream path can be fully separated [5].

The section II contains an overview of some related work. In section III, the mathematical relations that are the foundation of our model are presented. In section IV, the session life cycle is discussed in the view of relations defined in section III. The section V contains concluding remarks and directions for future work.

II. RELATED WORK

There are many approaches to formal modeling of communication protocols (as a more general class containing

ISBN: 978-1-61208-499-2

telephony communications). The use of Petri nets dates back several decades ago. In the field of telephony communications, the ITU-T directed efforts for a long time to establish software development methods supported by formal models. Especially in the area of feature interaction, several approaches to formal modeling which provided rapid detection of unwanted feature interactions were proposed [6]. A more recent comprehensive work on feature interaction problem is presented in [14]. Some other recent approaches are in [16] and [17]. An example of the use of Petri nets in modeling telephony networks is given in [7]. In the packet based telephony, there is also work in the development of services supported by formal models and a segment of that work is presented in [8]. In [19], authors applied formal modeling and verification of several fixes for privacy issues in 3G. In [20], authors present network topologies used by several popular chat applications. The structural model presented in this paper suits the topology used by Google+, while it differs from peer-to-peer and hybrid topologies of iChat and Skype, respectively.

With respect to modeling the structure of large IP telephony networks, important sources are documents published by IP telephony vendors. Those, usually corporate networks, are covered as in [12], where single and multisite models for corporate IP telephony are presented. The taxonomy in [13] is a bit more elaborate and it distinguishes single site, centralized and distributed call processing model, but it still covers only a corporate network case. On the other hand, an example of research in the structure of network is [18] where authors analyze clustering of bipartite Internet-to-telephone network ItPBN (IP telephone calls from Internet to traditional telephone network) and show that it has power law incoming/outgoing degree distribution. The ItPBN has one giant component and a large number of satellite components. The size of satellite components follows the power law.

In the first decade of XXI century, the SIP protocol gradually claimed dominant place as VoIP signaling protocol, leaving behind H.323. A proposal for a new VoIP signaling protocol, that overcomes some deficiencies of the SIP is presented in [15].

One of the trends in IP telephony is development of peer-to-peer (P2P) IP telephony systems. Some existing commercial systems already deploy P2P or hybrid architectures (see [20]). P2P SIP is an open technology based on this concept, see [21] as an example of recent work in this area. Ref [22] presents a P2P IP telephony system for wireless ad-hoc networks. Ref [23] presents a strategy to form the overlay network of P2P IP system.

It is stated [10] that one of drawbacks of traditional telephony models (Integrated Services Digital Network - ISDN, Intelligent Network - IN being good examples) is that they are based on the notion of a call. The call is described as an obscure and difficult notion [10]. Distributed Feature Composition architecture [9] that should solve that problem was proposed in late 90s, but did not enter in wide use. In our opinion, this property of the notion of the call stems from the fact that it is perceived both from the standpoint of an end user of telephony services and from the standpoint of an engineer developing services - and these two projections sometimes collide. This paper is, among other things, an attempt to say what the call is, but in terms of mathematical reasoning.

III. MATHEMATICAL RELATIONS

The state of network is described by a Connectivity State (CS) set. This set contains the following three most important relations (physically connectable - PhyC, connected - Conn, and negotiating - Nego). Nego relation covers both negotiation of session parameters and session establishment operation.

$$CS = \{PhyC, Conn, Nego\} \tag{3}$$

The PhyC relation represents current infrastructure state. It is the set of available communication paths. The PhyC relation is the subset of the set EPxEP. We consider this relation to be symmetric and transitive, but we note that in special cases this assumption may not hold - depending on the configuration of routing tables and polices. If there is an active link between *x* and *y*, then both (x,y) and (y,x) are members of the PhyC. Also, if there are links that connect *x* with *y*, and *y* with *z*, this means that (x,z) $\in$ PhyC. It can be assumed that the PhyC is reflexive. In this case, the class of equivalence of *x* that is the member of EP represents the connectivity potential of that endpoint. The *y* that is the member of EP and the member of the class of equivalence of *x* is an endpoint that is reachable from *x*. In a normal operation, there is only one class of equivalence - any endpoint is reachable from any other endpoint.

Conn and Nego relations are the subset of the set $\cup N^i$, i=2,...,n, where n is the greatest number of users that can participate in one telephony session - in that network. Or, more strictly limited,

$$Nego, Conn \subset \{EPxEP \cup (EP^i xCP), (i=1,...,n-1)\} \tag{4}$$

Neither of the relations is intrinsically reflexive. Both are symmetric and transitive. If x,y $\in$ N and *x* is connected to *y* by a telephony session, then *y* is connected to *x* too - because of characteristics of the telephony session. Also, because of characteristics of the telephony conference, if x,y,z $\in$ N and *x* is connected to *y* by the telephony session, and *y* is connected to *z*, then *x* is connected to *z*, too. In our model, the telephony session is a SIP session.

Again, because of characteristics of the telephony session, if *x* is negotiating with *y*, then *y* is negotiating with *x*, too. Transitivity of the Nego relation is not so obvious, but if we observe x, z $\in$ EP and y $\in$ CP, it can be said that if both *x* and z are negotiating with *y*, then *y* is negotiating with *z* (because of symmetricity), and it can be said that implicitly, *x* is negotiating with *z*, too. For *y* which is not a conference point, this does not hold.

It can be seen that

$$((\forall\ (x_1,...,x_{n-1}) \in EP^{n-1},\ \forall\ x_n \in CP\ )\ |\ ((x_1,...,x_n) \in Conn) \vee$$

$$((x_1,...,x_n) \in Nego)) =>$$

$$(\forall (x_i,x_j)\ |\ 1 \le i,j \wedge\ i,j \le n => (xi,xj) \in PhyC) \tag{5}$$

While the PhyC is a relation that represents connectivity, and in terms of network technology it belongs to so called communications oriented layers of ISO OSI [11] protocol stack (physical, data link, network and transport layers), Conn and Nego are about signaling, and belong to so called application oriented layers upper layers of ISO OSI stack (session, presentation and application layers), or the application layer in TCP/IP [11] stack.

For each n-tuple that is negotiating or connected at the moment, there has to exist a network path connecting each of the members of the n-tuple $(x_1,...,x_n)$.

$$(\forall\ (x_1,...,x_n)\in \mathrm{Conn})\ \ (\forall\ (x_i,x_j),\ 1\leq i \leq n,\ 1 \leq j \leq n)$$

$$(\exists\ R \subset RP)\ (\exists\ r_k,\ r_l \in R\ |\ (x_i,r_k) \in L \wedge (x_j,r_l) \in L)$$

$$(\exists\ f : N\text{->}R \cup \{x_i,x_j\}|\ f(1)=x_i \wedge f(|R\cup\{x_i,x_j\}|)=x_j\ \ \wedge$$

$$(\forall\ i\in N)\ \ (i<|R\cup\{x_i,x_j\}|\ \text{=>}(f(i),f(i+1))\in L)) \tag{6}$$

The same is true for $(\forall\ (x_1,...,x_n)\in \mathrm{Nego})$.

Where network path is a set of links connecting nodes belonging to R, and it is determined by relation f. For that reason, for each n-tuple that is an element of Conn or Nego, there is an m-tuple (n<=m) containing both the elements of R and the elements of the n-tuple. During the session establishment, certain resources are allocated at each member of the m-tuple. Upon the successful session establishment, depending on the type of session (i.e. use of Route headers [1] in SIP), the resources allocated at the elements of R are freed. In certain applications (especially those based on use of Back-to-Back User Agents [1]) it is required that routing points stay in the routing path during the whole session duration.

We can postulate that Conn relation is reflexive, because logically users at any endpoint can hear themselves during the session (echo functionality). In that case, Conn is a relation of equivalence. A telephone call in that case is an equivalence class of relation Conn.

If we postulate, a bit artificially though, that Nego is reflexive too, in which case that relation is a relation of equivalence too, it can be said that a class of equivalence of the Nego relation is a telephony session candidate.

The network load (in the sense of a set of ongoing network activities) can be projected into two functions of time:

$$\mathrm{LoadConn}:\ t\rightarrow \mathrm{Conn};\quad \mathrm{LoadNego}: t\rightarrow \mathrm{Nego}. \tag{7}$$

One important characteristic of the network can be noted as:

$$(\forall t_2)(\mathrm{Time}(t_2))(\ \forall x,y\in EP)((x,y)\ \in \mathrm{LoadConn}(t_2))\ \text{=>}$$

$$(\exists t_1)(\mathrm{Time}(t_1))\ (t_1<t_2)\ ((x,y)\ \in \mathrm{LoadNego}(t_1) \tag{8}$$

Or, in other words, only those nodes that in the earlier moment of time belonged to the class of equivalence of x in relation Nego can later belong to the class of equivalence of x in relation Conn. We remark though, that not all negotiations result in session establishment, which means that not all members of the class equivalence of x in relation Nego will become members of the equivalence class of x in relation Conn.

IV. SESSION LIFE-CYCLE

Presented relations make a model of associations between nodes of the network during its operation. However, they tell little about events that occur during one session.

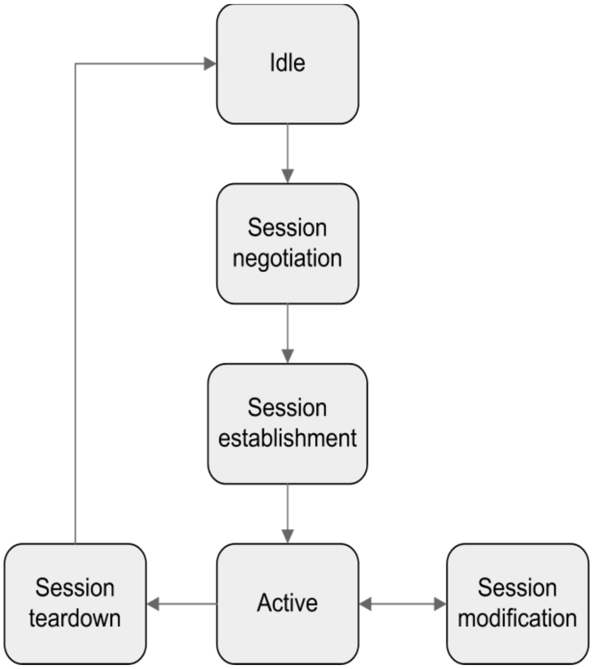


Figure 1. The session life-cycle.

Table 1 describes the most important states in the session life-cycle.

The states given in the table are additionally described with the following statements:

- After the session modification, *ep* returns to active state.
- During the session teardown, *ep* is deleted from the Conn relation.
- After the session teardown, *ep* returns to the idle state.

In this paper, we consider stable state to be the state in which the *ep* can stay unlimited time - in general case. Duration of transitional states is controlled by timeout controls specified by the telephony protocol, and this duration is on the average small in the sense of human perception, and compared to the average duration of stable states. The typical timeout duration in transitional states is in the order of seconds.

TABLE I. IMPORTANT STATES OF THE SESSION LIFE-CYCLE

State	Membership of *ep* in relations Conn and Nego	Type of the state
Idle state	*ep* is present neither in Nego nor in Conn	stable state
Session negotiation	*ep* is participating in Nego relation	transitional state
Session establishment	*ep* is participating in Nego relation	transitional state
Active state	*ep* is participating in Conn relation	stable state
Session modification	*ep* is participating in both Nego and Conn relations	transitional state
Session teardown	*ep* is participating in Conn relation	transitional state

The session life-cycle is presented in Fig.1. The figure presents the case of a successfully established call, as in a general case, the session can be terminated in any state (i.e. there can be failures in session negotiation, establishment or modification). It can be seen that the important operations that change the state of the session are:

- negotiation
- establishment
- teardown

Each of three listed operations is realized through transmission of messages between participating nodes. Messages are complex data structures and will not be further analyzed here.

The telephony session protocol defines the state space: the set of states and the set of possible graphs of state transitions. Each realized session can be represented as a graph in the state space defined by the protocol. This graph depends also on the configuration of participating nodes and end user decisions.

There is a problem when representing telephony network mathematically. This problem lies in the time dimension of events in the network. Each established connection is present in the network through resources that are allocated in nodes participating in the session. Session establishment, modification or teardown does not happen simultaneously in all participating nodes - because there is a finite time interval required for transferring messages between participating nodes. Thus, there are moments when the session is in different states at different nodes participating in the same session. This situation occurs during all operations. The following list is not comprehensive since for each operation, we list one example:

- session establishment - one node is already in the session active state (stable state) while the other is still in the session establishing state (transitional state).
- session modification - the same as above
- session termination - one node is already in the idle state, while the other is still in terminating state.

We can conclude that session is viewed from two viewpoints - two sides participating in the session. In the general case, the session state viewed from the two viewpoints is not the same.

A potentially viable approach is to consider composite state which takes into account the states of all viewpoints. For example, we can say that $(x_1, ..., x_n) \in$Conn, when all *ep* nodes in the n-tuple consider the session to be in active state. We can also say that $(x_1, ..., x_n) \in$Nego, when at least one of $x \in \{x_1, ..., x_n\}$ considers ongoing session to be in the negotiation or modification state. Thus, we say that $(x_1,...,x_n) \in$Conn $\cap$ Nego, when at least one of $x \in \{x_1,...,x_n\}$ considers ongoing session to be in the modification state while all *x* consider session to be in the active state.

V. CONCLUSION

The relations given here model some important static and dynamic aspects of the telephony network. The network structure is a crucial static aspect and telephony sessions are the most important dynamic aspect.

We have identified the following three important relations (physically connectable - PhyC, connected - Conn, and negotiating - Nego) defined over the set of network nodes. The call can be interpreted as an equivalence class of relation Conn. Some aspects of the session life-cycle are expressed in terms of two relations (Conn, Nego), as well as the nature of states in the life-cycle.

The relations given here are far from the comprehensive model. However, they can be a foundation of a more comprehensive model that would include more sophisticated features and that will make the model usable for practitioners. Those features are related to the quality of service (QoS) and security (privacy protection, authentication, denial of service (DoS) prevention and/or mitigation) – and are directions for future work.

ACKNOWLEDGMENT

This work has been partially supported by the Ministry of Education and Science of the Republic of Serbia under the project TR32031.

REFERENCES

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, and R. Sparks, “SIP: Session Initiation Protocol”, RFC 3261, Internet Engineering Task Force, 2002.

[2] ITU-T, “Visual Telephone Systems and Equipment for Local Area Networks Which Provide A Non-Guaranteed Quality Of Service”, H.323, 1996.

[3] G. Vandana, S. Dharmaraja, A. Viswanathan, Stochastic modeling for delay analysis of a VoIP network, Annals of Operations Research. Vol. 233 Issue 1, p171-180, Oct.2015.

[4] Technical Specification Group Services and System Aspects, IP Multimedia Subsystem (IMS), Stage 2, TS 23.228, 3rd Generation Partnership Project, 2006.

[5] Olivier Hersent, IP Telephony: Deploying VoIP Protocols and IMS Infrastructure, 2. edition, Wiley, 2010.

[6] D.O. Keck, P.J. Kuehn, The Feature and Service Interaction Problem in Telecommunications Systems: A Survey, *IEEE Trans on SE*, Vol 24, No 10, October, pp 779-796, 1998.

[7] N. Anisimov, K. Kishinski, A. Miloslavski, Formal model, language and tools for design agent's scenarios in call center systems, Proc of the 32nd Annual Intl Conf on System Sciences, Hawaii, 1999.

[8] M. Rocetti, A. Aldini, M. Bernardo, R. Gorrieri, QoS evaluation of IP telephony services: a specification language based simulation software tool, Systems Analysis Modeling Simulation, Vol 43 Issue 12, 2003.

[9] M. Jackson, P. Zave, Distributed Feature Composition: A Virtual Architecture for Telecommunications Services, IEEE Trans. On Software Engineering, Vol 24, No 10, Oct 1998.

[10] P. Zave, Calls Considered Harmful' and Other Observations: A Tutorial on Telephony, ACoS '98/VISUAL '98, AIN '97 Selected papers on Services and Visualization: Towards User-Friendly Design, pp 8-27, 1998

[11] A.S. Tanenbaum, D.J. Wetherall, Computer Networks, 5. edition, Prentice Hall, 2011

[12] IP Telephony Deployment Models, Cisco Unified CallManager Express Solution Reference Network Design Guide, Cisco Press, 2009

[13] R. Kaza, S. Asadullah, Cisco IP telephony: planning, design, implementation, operation, and optimization, Cisco Press, 2005

[14] P. A. Zimmer, Prioritizing Features Through Categorization: An Approach to Resolving Feature Interactions, PhD thesis, University of Waterloo, Canada, 2007.

[15] P. Zave, and E. Cheung, Compositional Control of IP Media, IEEE Transactions on Software Engineering, vol. 35, no. 1, January/February 2009.

[16] D. Lesaint, D. Mehta, B. O'Sullivan, L. Quesada, N. Wilson, Context-Sensitive Call Control Using Constraints and Rules, Principles and Practice of Constraint Programming – CP 2010, Lecture Notes in Computer Science, Vol 6308, pp 583-597, 2010.

[17] A.S. Fathabadi and M. Butler, Applying Event-B Atomicity Decomposition to a Multi Media Protocol, F.S. de Boer et al. (Eds.): FMCO 2009, Lecture Notes in Computer Science 6286, pp. 89–104, 2010.

[18] Empirical analysis of Internet telephone network: From user ID to phone, Qi Xuan, Fang Du and Tie-Jun Wu, Chaos 19, 023101, 2009.

[19] New Privacy Issues in Mobile Telephony: Fix and Verification, Myrto Arapinis, Loretta Mancini, Eike Ritter, Mark Ryan, Nico Golde, Kevin Redon, Ravishankar Borgaonkar, Proceedings of the 2012 ACM Conference on Computer and Communications Security CCS'12, Raleigh, NC, USA, pp 205-216, 2012

[20] Video Telephony for End-consumers: Measurement Study of Google+, iChat, and Skype, Yang Xu, Chenguang Yu, Jingjiang Li and Yong Liu, Proceedings of the 2012 ACM conference on Internet measurement conference IMC '12, Boston, USA, pp 371-384, 2012

[21] A Priority Based Lookup Model for VoIP Applications in Unstructured P2P Networks, Mourad Amad, Djamil Aïssani, Ahmed Meddahi, Nouria Madi, Razika Bouiche, Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication IPAC '15, Algeria, 2015

[22] P2P IP Telephony over wireless ad-hoc networks, Mehdi Mani, Winston K. G. Seah, Noel Crespi, Reza Farahbakhsh, Peer-to-Peer Networking and Applications, Volume 5, Issue 4, pp 363-383, December 2012

[23] P2P IP telephony over wireless ad-hoc networks : a smart approach on super node admission, Mehdi Mani, Winston Seah, Noel Crespi, Reza Farahbakhsh, Peer-to-Peer Networking and Applications, Springer, 2012, 5 (4), pp.363-383.

Error Resilient FEC Video Transmission Based on Optimal FEC Code Rate Decision

Tae-jun Jung, Joowon Lee, and Kwang-deok Seo
Yonsei University
Wonju, Gangwon, South Korea
e-mail: kdseo@yonsei.ac.kr

Yo-Won Jeong
Samsung Electronics Corporation
Suwon, Gyeonggi, South Korea
e-mail: suance88@hanmail.net

***Abstract*—To support high-quality wireless video transmission over IP, error-resilience techniques are important because wireless video has more stringent requirements than general video transmission for packet loss, latency and jitter. The optimal forward error correction (FEC) code rate decision is a crucial procedure to determine the optimal source and channel coding rates to minimize the overall picture distortion when transporting video packets over packet loss channels. The conventional FEC code rate decision schemes using an analytical source coding distortion model and a channel-induced distortion model are usually complex and typically employ the process of model parameter training, which involves potentially high computational complexity and implementation cost. To avoid the complex modeling procedure, we propose a simple but accurate joint source-channel distortion model to estimate the channel loss threshold set for optimal FEC code rate decision.**

Keywords- error-resilient video transmission; FEC code rate decision; joint source-channel coding; forward error correction.

I. INTRODUCTION

The problem of error control for packet loss in wireless video communication is becoming increasingly important because of the growing interest in video delivery over unreliable channels such as wireless networks and the Internet. With the growth of wireless communication infrastructure, such as Wi-Fi, more and more IP networks are taking a mixed form consisting of wireless and wired channels. One inherent problem in the wireless communications system is that information may be altered or lost during transmission due to channel noise. The effect of such information loss can be devastating for the transport of compressed video because any damage to the compressed bit stream may lead to objectionable visual distortion at the decoder. Moreover, high-quality image and high scene fidelity are extremely important in IP based video transmission to build a complete security [1] [2]. Thus, it is necessary for the video sender to provide adequate error resilience features to protect the video data from the channel errors. Two effective approaches for error resilience and protection are automatic repeat request (ARQ) and forward error correction (FEC) [3] [4] [5]. Due to the stringent delay constraint imposed by real-time video transmission, it is often considered more beneficial to use FEC than ARQ [6] [7]. Because the packet loss rate in the channel changes dynamically, FEC coding with a fixed code rate either wastes channel bandwidth at low packet loss rates or is insufficient to recover video information at high packet loss rates [8]. Thus, it is more efficient to adapt the code rate in response to time-varying packet loss dynamics in order to ensure consistent optimal video quality.

Several previous studies have focused on determining the optimal code rate for joint source-channel coding (JSCC). In [9] and [10], a source coding distortion model and a channel-induced distortion model were proposed as a means of determining a combination of the optimal intra-refresh rate and code rate. These models enable the optimal code rate to be estimated fairly accurately. However, these model equations have numerous parameters that depend on the characteristics of the input video sequences.

In this paper, we propose a practical code rate decision scheme based on a joint source-channel distortion model. The joint source-channel distortion model is used to estimate an accurate channel loss threshold set. With the accurate channel loss threshold set, we can determine the optimum code rate for various channel conditions and therefore adjust the code rate to maintain the maximum video quality under the given channel condition. In order to efficiently apply the joint source-channel distortion model to general live video transmission, we present a practical test run scheme to train the scene-dependent model parameters (SDMP) of the proposed model in real time with acceptable computational complexity. With extensive simulations, we verify the performance of the proposed model and the accuracy of the obtained code rate in various packet loss environments.

This paper is organized as follows. In Section II, we introduce the fundamental video transmission system. Section III describes the loss threshold sets. In Section IV, we present the proposed joint source-channel distortion model. In Section V, we show the experimental results, and finally concluding remarks are presented in Section VI.

II. FUNDAMENTAL VIDEO TRANSMISSION SYSTEM AND RESIDUAL PACKET LOSS RATE

The fundamental structure of the video transmission system over the Internet that supports both source and channel codings is presented in Fig. 1. The video transmission system consists of a video encoder/decoder, a channel encoder/decoder, and a channel adaptation block. The channel adaption block optimizes the video quality and maintains the end-to-end delay within a given maximum delay bound. The sender periodically receives information about the instantaneous channel status from the receiver through a RTP control protocol (RTCP) feedback report [11]. From this information, the block for the channel characteristic estimation calculates the current status of the channel packet loss rate, p_L, and the available total transmission rate, R_T.

In video coding and transmission over noisy channels, the Reed-Solomon (RS) code is one of the widely used FEC schemes [4]. The channel encoder in Fig. 1 generates n FEC packets for every k video packets by the RS code, denoted as $RS(n,k)$ where n denotes the number of total packets produced by each RS coding. The code rate of RS code is defined as

$$r = \frac{k}{n}. \qquad (1)$$

Let R_S and R_C be the source and channel coding bit rates, respectively. If we define the total transmission rate as R_T, then R_S and R_C can be expressed as

$$R_S = R_T \cdot r, \quad R_C = R_T \cdot (1 - r). \qquad (2)$$

The rate of the packet loss of all data, including video data and FEC redundant data, is generally called the residual packet loss rate (RPLR) [12]. If we discard all the packets in a transmission group when more than $(n-k)$ packets are lost, the RPLR is a good parameter to represent the channel distortion [12]. The RPLR is simply formulated as

$$RPLR = \sum_{i=n-k+1}^{n} \begin{pmatrix} n \\ i \end{pmatrix} p_L^i (1 - p_L)^{n-i}. \qquad (3)$$

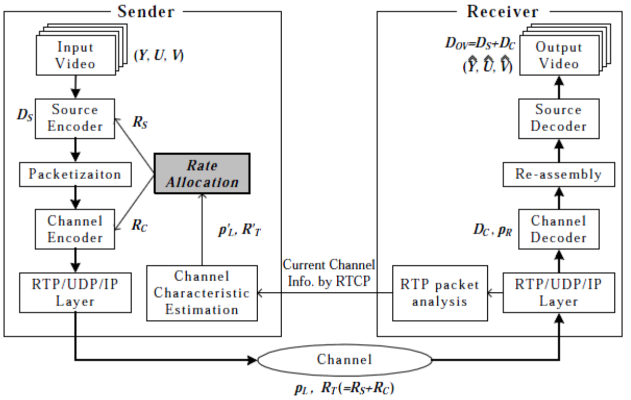


Figure 1. Fundamental transmission structure of the IP video transmission system including source and channel codings.

The overall distortion, D_{OV}, is defined as the mean square error (MSE) of the luminance values (Y component in Fig. 1) of all the pixels between the input video frame of the sender and the decoded video frame of the receiver. The overall distortion can also be represented as the sum of the source-coding distortion, D_S, and the channel-induced distortion, D_C [12].

III. Loss Threshold Sets

To analyze the overall distortion characteristics, we measure overall distortion values for a wide range of channel packet loss rates and various code rates. For this measurement, we employ an H.264/AVC codec [13] and assume that the number of slices per frame is adjusted to three or more in order to prevent the sizes of slices from exceeding the maximum transmission unit (MTU) size. Each slice then becomes one packet. The packet loss characteristic of the channel is assumed to be independent and random. The experimental results of the overall distortion values, $D_{OV}(r,p_L)$, for the test sequence *Foreman* (CIF) are shown in Fig. 2. We can see that the packet loss rates, which serve as thresholds to change the optimal code rate are crucial information. We call the set of these values a *channel loss threshold set* (*CLTS*). In Fig. 2, the CLTS is $\{PL_2, PL_3, PL_4, PL_5\}$.

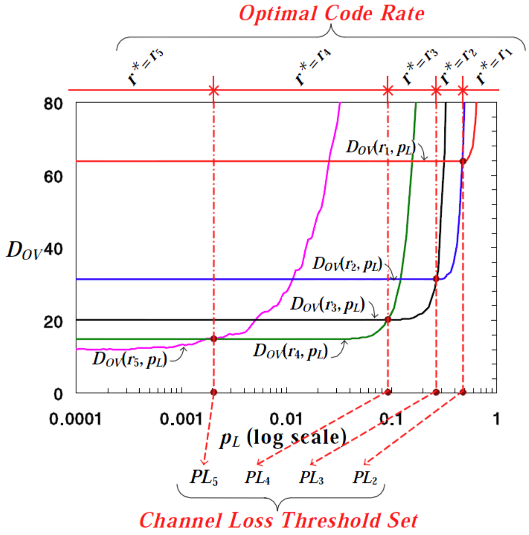


Figure 2. Relationship among overall distortion, code rate, and CLTS.

It is difficult to directly calculate the CLTS because it is impracticable to draw all necessary distortion curves in practical video surveillance applications. In order to efficiently obtain the CLTS, an analytic model is needed. Since the RVPLR is more directly coupled with video quality than the channel packet loss rate, for facile modeling, we introduce the concept of the residual loss threshold set (RLTS), a loss threshold set based on the RVPLR. We define the RLTS as

$$RLTS = \{PR_k \mid PR_k = p_R(n_0, k, PL_k),\ k = 2, 3, \ldots, n_0\}, \qquad (4)$$

where p_L=PL_k. Note that we can directly obtain the CLTS from the RLTS using (4). The RLTS can be obtained more easily than the CLTS by the proposed joint source-channel distortion model which is explained in the next section.

IV. Joint Source-Channel Distortion Model

Obtaining the CLTS by plotting all distortion curves as shown in Fig. 2 whenever the characteristics of the input video change, is extremely time-consuming. To address this problem, we propose a model for the RLTS that can derive all elements of the RLTS, and the CLTS can be easily obtained from the RLTS. This model is based on the following observation: the difference of source coding distortion between r_k and r_{k-1} is likely to be equal to the channel distortion at r=r_k and p_L=PL_k. To illustrate this, the above statement is represented in Fig. 3 for k = 2, 3, 4. Note that in Fig. 3 the notation $D_C(r,p_L)$ refers to the channel distortion when the code rate is r and the channel packet loss rate is p_L. This observation is attributed to the geometric characteristic that the $D_{OV}(r_k,p_L)$ curve abruptly decreases and the $D_{OV}(r_{k-1},p_L)$ curve maintains a nearly constant value at their intersection. We can formulate the observation as follows:

$$D_{OV}(r_{k-1},0)-D_{OV}(r_k,0)\approx D_C(r_k,PL_k), \qquad k=2,3,\ldots,n_0. \qquad (5)$$

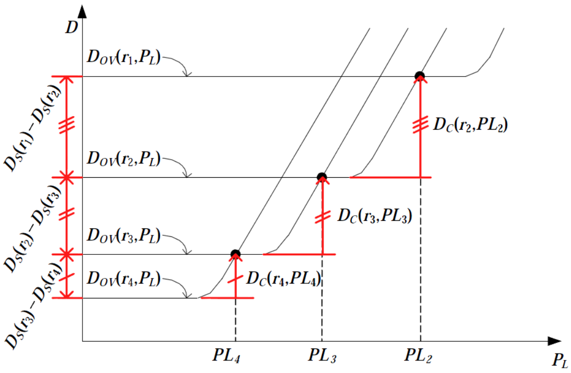


Figure 3. Conceptual distortion curves.

We can see that no ratio values between $|D_{OV}(r_{k-1},0)-D_{OV}(r_k,0)|$and $|D_C(r_k,PL_k)|$ fall out of the range between 1 and 0.75. From this result, (5) can be represented as

$$0.75\le\left|\frac{D_{OV}(r_{k-1},0)-D_{OV}(r_k,0)}{D_C(r_k,PL_k)}\right|\le 1 \qquad k=2,3,\ldots,n_0. \qquad (6)$$

Because the ratio of $|D_{OV}(r_{k-1},0)-D_{OV}(r_k,0)|$ to $|D_C(r_k,PL_k)|$ is narrowly bounded, (6) can be approximated as a constant value

$$\left|\frac{D_{OV}(r_{k-1},0)-D_{OV}(r_k,0)}{D_C(r_k,PL_k)}\right|\approx c, \quad 0.75\le c\le 1, \qquad k=2,3,\ldots,n_0. \quad (7)$$

The relationship between the source coding rate R_S and the distortion yielded by the source coding D_S is called the rate-distortion model (R-D model). In this study, we employ an inverse proportional model because it is simple and accurate [12]. $D_{OV}(r_k,0)$ can then be represented as

$$D_{OV}(r_k,0)=\frac{w_1}{R_T r_k-w_2}+w_3, \qquad k=2,3,\ldots,n_0. \qquad (8)$$

where $R_T r_k$ denotes the source coding rate, and w_1, w_2, and w_3 are SDMPs that can be varied according to input video characteristics. Note that, unlike w_1 and w_2, w_3 can be varied by the RVPLR of the previous state even if the input video characteristic is unchanged because the current distortion can be affected by the previous distortion. We can see that the distortion of the received video was proportional to the RVPLR regardless of k. Therefore, we can represent $D_C(r_k,PL_k)$ as

$$D_C(r_k,PL_k)=\widetilde{D}_C(k,PR_k)=w_4 PR_k, \qquad k=2,3,\ldots,n_0, \qquad (9)$$

where w_4 is an SDMP. If we substitute (8) and (9) into (7) and rearrange for PR_k, we finally obtain the set of PR_ks as

$$PR_k=\frac{\beta}{(k-(\alpha+1))(k-\alpha)},\ \alpha=\frac{w_2 n_0}{R_T},\ \beta=\frac{w_1 n_0}{c\cdot w_4 R_T}, \qquad k=2,3,\ldots,n_0, \quad (10)$$

where α and β are the final SDMPs of the joint source-channel model.

If we equivalently express (10) as

$$PR_k=\frac{\beta}{k\left(1-\left(\frac{\alpha}{k}+\frac{1}{k}\right)\right)\cdot k\left(1-\frac{\alpha}{k}\right)}, \qquad k=2,3,\ldots,n_0, \qquad (11)$$

and assume that k is much larger than α, namely $|\alpha/k|\ll 1$, we can approximate (11) to (12).

$$PR'_k=\frac{\beta}{k(k-1)}, \qquad k=2,3,\ldots,n_0. \qquad (12)$$

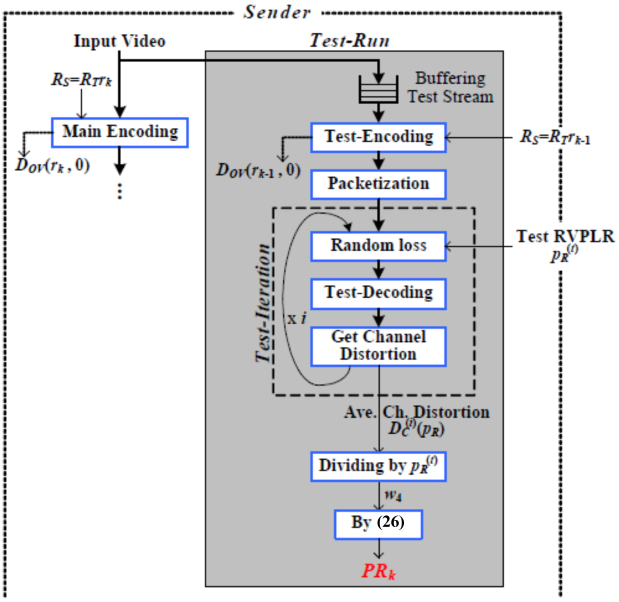


Figure 4. The overall process of the test run in the sender.

To obtain the SDMPs, α and β of (11), we need PR_i and PR_j where i and j are different values among 2, 3, …, n_0. In the case of the approximate model of (12), we need only one PR_k. The test run is a real-time process for obtaining the sample PR_ks. Because the test run should not disturb the real-time performance of the basic process of the sender of Fig. 1, it is performed using residual computing resources whenever the basic process of the sender is in an idle state. Since such residual computing resources are generally insufficient, the computational load of the test run should be reduced.

The overall process of the test run for obtaining a sample PR_k is based on

$$\left|\frac{D_{OV}(r_{k-1},0)-D_{OV}(r_k,0)}{w_4 PR_k}\right| \approx c, \quad 0.75 \le c \le 1, \qquad k = 2,3,...,n_0, \qquad (13)$$

which is derived by substituting (9) into (7). To obtain a sample PR_k in (13), we have to obtain three values, $D_{OV}(r_{k-1},0)$, $D_{OV}(r_k,0)$, and w_4, in advance. The overall process of the test run, including the processes for obtaining these three values, is shown in Fig. 4.

Since there is a high possibility that the characteristics of an input video will change significantly when scene-changes occur or a scene is long, the test run needs to be performed periodically. We buffer the test stream whenever a test run is started. The additional encoding by the test run generally requires a huge computational load, because the video encoding includes various complex operations, such as a motion estimation, a discrete cosine transform (DCT), and an inverse DCT. Of these operations, motion estimation takes the most time in the encoding process [14]. To reduce the computational load of motion estimation, we reuse the motion vector information evaluated via main encoding. The main encoding and the test encoding are performed using the same input frames, and the source coding rates $R_T \cdot r_{k-1}$ and $R_T \cdot r_k$ are so similar that motion vectors of the same macroblock position resulting from the main encoding and the test encoding are very similar [15].

V. Experimental Results

The experiment was performed using a desktop computer system with a Quad-Core 4.0 GHz CPU and 2 GByte memory. For input video, we used eight test sequences with 720p HD (1280x720) resolution, *Stockholm*, *Sunflower*, *Tractor*, *Troy*, *Blue Sky*, *Rush Hour*, *Park Joy*, *Citybus*, for which the frame rates are 15 fps. The basic parameters of the experiment are as follows: n_0 is 10 and R_T is 1,536 Kbps. For the test run, we stored each test sequence for the first second as a test stream. We assumed that the main encoding was performed with k=8. The test RVPLR was fixed at 1/15. For each experiment, we compare the three types of systems discussed below.

1) Optimal system (OS): In this system, the optimal code rate is considered to be the code rate that minimizes the sum of D_S and D_C, which are formulated as in Eqs. (16) and (18), respectively. In the test encoding, we perform two additional encodings to obtain three SDMPs of (16) for the test stream with r_k=0.2 (k=2) and 0.9 (k=9). In the test-iteration, the loss rate is uniformly set to 1/15 regardless of class, and the number of iterations is set to 30.

2) RLTS model-based system (RMS): In this system, we calculate the optimal code rate through the proposed RLTS model in (19). To obtain two SDMPs of (19), we need two PR_k samples. One sample can be obtained by test-encoding with r_k=0.7 (k=7) because the main encoding is performed with r_k=0.8. To obtain another sample, we perform two test-encodings with r_k=0.3 and 0.4.

3) Approximate model-based system (AMS): In this system, we calculate the optimal code rate through the approximate RLTS model. To obtain the SDMP, we need one PR_k sample. It can be obtained by test-encoding with r_k=0.7. Other settings for the test-iteration are the same as RMS.

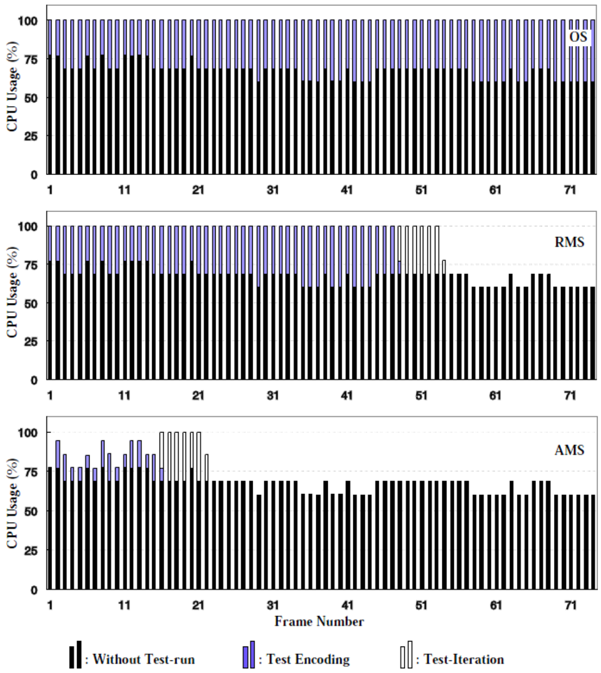


Figure 5. The Comparison of processing overheads of test runs for the three types of systems: OS, RMS, and AMS. The *Stockholm* sequence of HD resolution is used.

The experimental results of processing overheads required for the test run for each frame of the *Stockholm* sequence are shown in Fig. 5. The black bar represents the CPU usage of each frame for the entire processing without the test run. The remaining CPU resource can be used for the test run. The blue and white bars represent the CPU usage for the test-encodings and test-iteration, respectively. In the case of an OS (optimal system), the test-encodings are not completed at the end of the sequence because the OS requires two additional test-encodings that introduce a heavy computational load. In the case of RMS, the test run is completed before the end of the sequence. However, because the test run is completed too late, the period during which the output of the test run can be used is very short. On the other hand, the AMS requires only one test-encoding, and we apply all the speed-up methods described in Section 4. Thus, the test run is consequently completed very rapidly, and the output of the test run can be used over a substantial time period

The experimental results of the PSNR performance for the channel packet loss rate are shown in Table I. The PSNR values are obtained as follows. First, we perform a test run via three methods: the OS, the RMS, and the AMS. Next, we determine the optimal code rate for the channel packet loss rate through the model and its SDMPs obtained from each test run method. After applying this code rate, we simulate the video transmission system of Fig. 1 and calculate the average PSNR of the total frames of the test sequence. Finally, all the PSNR results are obtained using 50 runs of the above procedure in order to obtain statistically meaningful average values [16]. In Table I, AMS-PR and AMS-UR are the PSNR results when applying the pseudo-random loss and uniform-random loss, respectively.

TABLE I. SUMMARY OF THE PSNR RESULTS FOR TEST SEQUENCES

Test sequence	Test-Run	Avg. PSNR (dB)	Max. PSNR gap for OS
Stockholm	OS	28.653	
	RMS	28.509	1.631
	AMS-PR	28.288	2.574
Sunflower	OS	33.258	
	RMS	33.249	0.283
	AMS-PR	33.026	3.150
Tractor	OS	32.257	
	RMS	32.197	0.877
	AMS-PR	31.980	2.150
Troy	OS	30.513	
	RMS	30.456	0.567
	AMS-PR	30.173	3.198
Blue Sky	OS	33.278	
	RMS	33.222	0.575
	AMS-PR	32.932	2.746
Rush Hour	OS	29.171	
	RMS	29.105	0.472
	AMS-PR	28.836	2.199
Park Joy	OS	24.547	
	RMS	24.497	0.771
	AMS-PR	24.210	3.008
Citybus	OS	31.106	
	RMS	31.007	0.184
	AMS-PR	29.772	3.207

In Table I, the performance of the AMS-PR is very close to the performance of the OS except for the high loss rate. At about p_L=0.6, the maximum PSNR gap between the AMS-PR and the OS is 2.57dB (*Stockholm*) and 3.15dB (*Sunflower*). The range of p_L having a high PSNR gap is not long enough to be considered a serious drawback. The gap between the average values of the PSNR curve of the AMS-PR and the OS is just 0.36dB (*Stockholm*) and 0.23dB (*Sunflower*). For other test sequences, similar results are observed.

VI. CONCLUSIONS

In this paper, we propose a simple but accurate joint source-channel distortion model to estimate the channel loss threshold set for optimal FEC code rate decision. The proposed joint model exempts us from a complex training procedure for obtaining many model parameters for separate source and channel models, which is usually required in the conventional code rate decision approach. Since the proposed model is expressed as a simple closed form and has a small number of SDMPs, a video sender using the model can be easily implemented. For training the SDMPs in real time, we propose a practical test run procedure. This method increases the speed of the test run while maintaining its accuracy for training the SDMPs. When compared to the previous methods, by using the proposed simple model and practical test run method, the video sender can determine the optimal code rate for JSCC whenever there is a change in the packet loss condition in the channel. The experimental results confirm that the proposed model and its test run procedure can accurately estimate the channel loss threshold set in real time, resulting in an optimal FEC code rate with low computational complexity.

ACKNOWLEDGMENT

This work was supported by a grant ‛Biotechnology & GMP Training Project’ from the Korea Institute for Advancement of Technology (KIAT), funded by the Ministry of Trade, Industry and Energy (MOTIE) of the Republic of Korea (N0000961).

REFERENCES

[1] Y. Abudoulikemu, Y. Huang, and C. Ye, “A scalable intelligent service model for video surveillance system based on RTCP,” *Int. Conf. on Signal Processing Systems*, vol. 3, pp. 346-349, 2010.

[2] D. Chu, C. Jiang, Z. Z. Hao, and W. Jiang, “The design and implementation of video surveillance system based on H.264, SIP, RTP/RTCP and RTSP,” *Int. Symp. on Computational Intelligence and Design (ISCID)*, vol. 2, pp. 39-43, 2013.

[3] D. Wu, Y. T. Hou, and Y. Q. Zhang, “Transporting real-time video over the Internet: Challenges and approaches,” *Proc. of the IEEE*, vol. 88, no. 12, pp. 1855-1877, Dec. 2000.

[4] Y. Wang and Q. Zhu, “Error control and concealment for video communication: A review,” *Proc. of the IEEE*, vol. 86, no. 5, pp. 974-997, May 1998.

[5] S. Jalil, M. Abbad, and R. Azouzi, “Hybrid FEC/ARQ schemes for real-time traffic in wireless networks,” *Int. Conf. on Wireless Networks and Mobile Communications (WINCOM)*, pp. 1-6, 2015.

[6] S. Soltani, K. Misra, and H. Radha, “Delay constraint error control protocol for real-time video communication,” *IEEE Trans. Multimedia*, vol. 11, no. 4, pp. 742-751, June 2009.

[7] S. Jalil, M. Abbad, and R. Azouzi, “Hybrid FEC/ARQ schemes for real-time traffic in wireless networks,” *Int. Conf. on Wireless Networks and Mobile Communications (WINCOM)*, pp. 1-6, 2015.

[8] N. Abdelhamid, T. Taleb, and L. Murphy, “Forward error correction strategies for media streaming over wireless networks,” *IEEE Communications Magazine*, vol. 46, no. 1, pp. 72-79, 2008.

[9] K. Stuhlmuller, N. Farber, and B. Girod, “Analysis of video transmission over lossy channels,” *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 1012-1032, June 2000.

[10] P. Frossard and O. Verscheure, “Joint Source/FEC Rate Selection for Quality-optimal MPEG-2 Video Delivery,” *IEEE Trans. on Image Proc.*, vol.10, no.12, pp.1815-1824, Dec. 2001.

[11] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “Real-time transport protocol,” IETF *RFC 3550*, July 2003.

[12] K. Stuhlmuller, N. Farber, and B. Girod, “Analysis of video transmission over lossy channels,” *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 1012-1032, June 2000.

[13] Joint Video Team (JVT), H.264/AVC Reference Software Version JM 19.0, available from http://iphome.hhi.de/suehring/tml/ download/, retrieved: June 2016.

[14] J. Osterman *et al.*, “Video coding with H.264/AVC: tools, performance, and complexity,” *IEEE Circuits and Systems Magazine*, vol. 4, no. 1, pp. 7-28, First Quarter 2004.

[15] H. Zhou, J. Zhou, and X. Xia, “The motion vector reuse algorithm to improve dual-stream video encoder,” in Proc. *Int. Conf. Signal Process*., pp. 1283-1286, Oct. 2008.

[16] Q. Qu, Y. Pei, and J. W. Modestino, “An adaptive motion-based unequal error protection approach for real-time video transport over wireless IP networks,” *IEEE Trans. Multimedia*, vol. 8, no. 5, pp. 1033-104, Oct. 2006.

In Defense of Stint for Dense Breach-Free Sensor Barriers

Jorge A. Cobb

Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080-3021
U.S.A.
Email: cobb@utdallas.edu

***Abstract*—A sensor barrier consists of a subset of sensors that divide an area of interest into two regions so that no intruder can move from one region to another without being detected by at least one sensor. The length of time over which the sensors protect the area can be maximized if the sensors are divided into disjoint barriers, and only one barrier is active at a time. Dividing the sensors into a maximum number of disjoint barriers can be done with the well-known Stint algorithm. However, recently, a new security problem was discovered, known as a barrier-breach, that allows an intruder to cross the area undetected by taking advantage of the time when one barrier is replaced by the next. This is dependent not on the structure of an individual sensor barrier, but on the relative shape of two consecutive sensor barriers. There have been several heuristics proposed in the literature that attempt to maximize the number of breach-free barriers. In recent work, we proposed a heuristic that outperforms earlier heuristics. In this paper, we refine our previous heuristic to deliver even better performance than before by the careful elimination of redundant nodes. In addition, me present a simple modification of the Stint algorithm that results in a method that outperforms all others when the density of sensors nodes is very high.**

***Keywords*–Sensor networks; Barrier coverage; Security breaches.**

I. Introduction

A wireless sensor network (WSN) consists of an area of interest in which a large number of sensor nodes have been deployed. We assume each sensor has a limited battery lifetime, and thus it is crucial for the network to use the limited sensor lifetime wisely. Information that the sensors collect could be relayed to a wireless base station [1].

In general, there are two types of coverage provided by a sensor network: full coverage and partial coverage. In full coverage, the entire area is covered at all times by the sensors. Therefore, any event occurring anywhere in the area is detected immediately [2] [3] [4] [5]. In partial coverage, only certain regions at a time are covered by the sensors. Thus, any event occurring outside of the current region being monitored is not detected. [6] [7] [8].

A sensor barrier consists of a subset of sensors that divide an area of interest into two regions so that no intruder can move from one region to another without begin detected by at least one sensor. Barrier coverage is thus a special case of partial-coverage. There have been extensive studies of sensor barriers due to their many applications, in particular intrusion detection [9] [10] [11] [12] [13] [14] [15] [16]. Figure 1(a) highlights a subset of sensors that provide barrier coverage to the area.

Maintaining full coverage is often counter-productive for applications such as intrusion detection, because the protection only lasts for the duration of a single lifetime of the sensors. However, if n disjoint barriers are constructed, then only one barrier needs to be active at a time. When the currently active barrier is running out of power, the next barrier is activated. In this manner, the protection lasts n times the lifetime of a sensor. Figure 1(b) shows the sensors divided into four barriers.

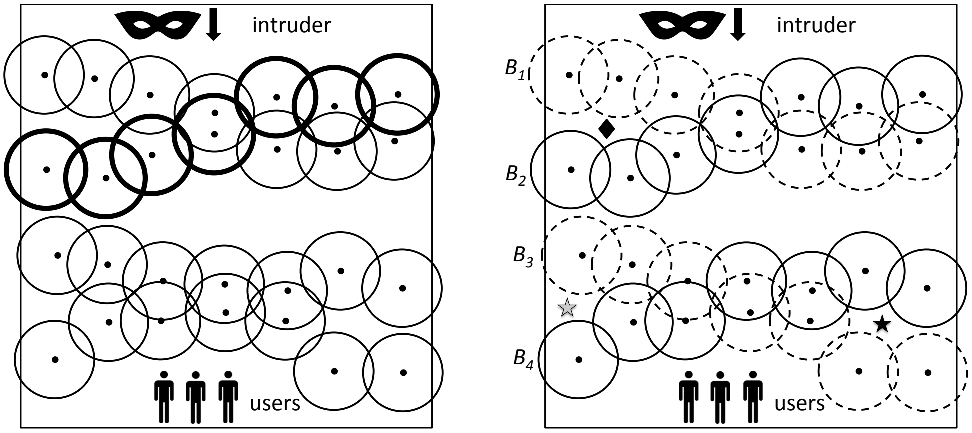
The problem of dividing the sensors into the maximum number of disjoint barriers has been solved in polynomial time in [11]. Here, two optimal algorithms, Stint and Prahari, are presented. Stint considers the case when the remaining battery level of each sensor is equal, while Prahari addresses the harder case in which each sensor may have different remaining battery level. Their approach is based on transforming the sensor connectivity graph into a maximum flow problem.

In recent years, [17] [18], a vulnerability of sensor barriers, known as a *barrier breach*, was discovered. For some barriers, it is possible for an intruder to cross the area of interest after activating one barrier and deactivating the previous one. This is true even though each of these two sensor barriers correctly divides the area into two disjoint regions, and thus provides appropriate coverage.

Although methods have been devised for dividing a group of sensors into breach-free barriers [17] [18], these are heuristics and are not guaranteed to be optimal. To our knowledge, the computational complexity of determining the largest number of breach-free sensor barriers remains an open problem.

In earlier work [19], we presented a heuristic which outperforms those in [17] [18]. In this paper, we refine our previous heuristic to deliver even better performance than before by the careful elimination of redundant nodes. In addition, me present a simple modification of the Stint algorithm that results in a method that outperforms all others when the density of sensors nodes is very high.

The rest of this paper is organized as follows. Section II reviews earlier work, such as the definition of a barrier breach and the heuristics from [17] [18] [19]. In Section III, we present our improved heuristic. In Section IV, we present a modification of Stint whose output is guaranteed to be breach-free. Finally, simulation results and concluding remarks are presented in Sections V and VI respectively.



(a) Sensor Barrier

(b) Barrier Breaches

Figure 1. Sensor Barriers

II. Background

In this section, we overview the concept of a barrier breach. In addition, we discuss Stint [11], and how its solution is vulnerable to barrier breaches. We then overview the heuristics presented in [17] [18] [19].

A. Barrier Breaches

We first illustrate the concept of a barrier breach with the example of Figure 1(b). The figure shows four different sensor barriers, with each barrier displayed with different line types.

Assume the intruder penetrates the area in the direction of the users. Assume also that all sensor nodes have the same lifetime, which we normalize to one unit. If all sensors are operational at the same time, then we form a single barrier that protects the users for a duration of one unit.

Instead, we can divide the sensors in four barriers, B_1 through B_4. If we use the barriers in a sequential wakeup-sleep cycle (B_1, B_2, B_3, and finally B_4), the users are protected for a total of four time units.

While this strategy is appealing, it suffers from the following drawback.

(a) The order in which B_1 and B_2 are scheduled affects the effectiveness of the barriers. Consider scheduling B_2, followed by B_1. In this case, an intruder could move to the point highlighted by a diamond, and after B_2 is turned off, the intruder is free to cross the entire area.
(b) Only one of B_3 and B_4 is of use. To see this, suppose that we activate B_3 first. In this case, the intruder can move to the location marked by the black star. Then, when B_4 is activated and B_3 deactivated, the intruder can reach the users undetected. The situation is similar if B_4 is activated first, and the intruder moves to the location of the grey star.

The above drawback was originally presented in [17] and given the name *barrier breach*. A barrier breach for an ordered pair (B_1, B_2) of barriers is a point p such that p is outside the sensing range of B_1 and B_2, and furthermore, B_1 cannot detect an intruder moving from the top of the area towards p, and B_2 cannot detect the intruder moving from p towards the bottom of the area.

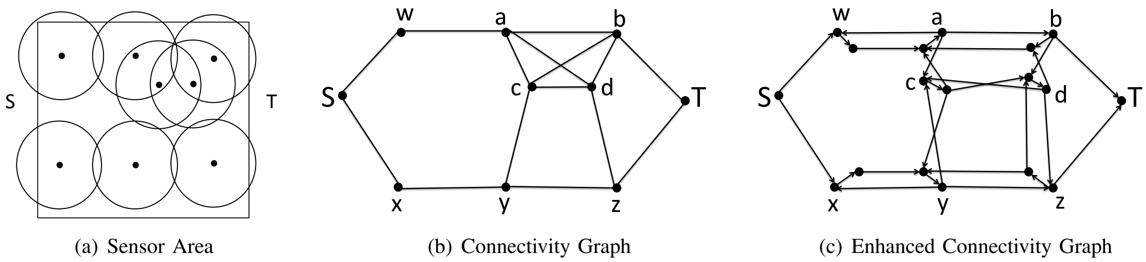
Note that this definition can be easily extended to a sequence of barriers, and thus, to the schedule used to activate the barriers.

B. Upper Bound on Breach-Free Barriers

Under the assumption that all sensor nodes have equal lifetime, finding the largest number of disjoint sensor barriers has been solved in polynomial time by Kumar et. al. [11] with their algorithm known as Stint. This provides obviously an upper bound on the maximum number of breach-free barriers, because Stint does not take barrier breaches into account. Given that the complexity of obtaining the largest number of breach-free barriers remains an open problem, several heuristics have been presented in the literature [17] [18] [19]. Some of these heuristics are based on Stint, so we briefly overview Stint below.

Consider Figure 2(a), with a sample sensor network. The first step consists of adding a virtual source node S and a virtual destination node T. Then, a connectivity graph is built containing all the sensor nodes as vertices in the graph, including the virtual nodes S and T. Two nodes are then connected via an edge iff their sensing ranges overlap. Also, an edge is added between S and any sensor whose range overlaps the left border of the area. Similarly, an edge is added between T and any sensor whose range overlaps the right border of the area. The resulting graph is shown in Figure 2(b). Finally, the maximum number of node-disjoint paths are found between S and T.

Computing the maximum number of node-disjoint paths is done with a small variation of the above graph and applying a maximum-flow algorithm. Each sensor node x in the graph is replaced by two nodes, x_{in} and x_{out}, with a directed edge from x_{in} to x_{out}. For every sensor node y that is a neighbor of x, the directed edges (x_{out}, y_{in}) and (y_{out}, x_{in}) are added to the network. Finally, an edge (S, x_{in}) is added for every neighbor x of S, and also an edge (y_{out}, T) for every neighbor y of T. All of these directed edges have a capacity one. The



(a) Sensor Area (b) Connectivity Graph (c) Enhanced Connectivity Graph

Figure 2. Stint Sensor Barriers

resulting enhanced connectivity graph is shown in Figure 2(c). The node-disjoint paths are obtained by running a maximum-flow algorithm on this enhanced connectivity graph.

C. Barrier Crossings

Several heuristics for maximizing the number of non-penetrable barriers have been proposed in [17] [18]. These heuristics are based on variations of Stint that restrict the connectivity graph of Figure 2(b) in a manner that ensures the output is breach-free. In [18], it was shown via simulation that one of these heuristics outperforms the others. Next, we briefly describe this heuristic, which will be used for comparison against our heuristic that we present in Section III.

The heuristic begins by constructing the connectivity graph of Stint, as in Figure 2(b), and removing edges that cross each other. For example, in Figure 2(b), edges (c, b) and (a, d) cross each other. One of these edges must be removed, leaving in place a graph where no edges cross. Then, as in Stint, the maximum number of node-disjoint paths are found, resulting in the desired sensor barriers.

What remains to be decided is, of the set of edges that cross each other, which ones should be removed? Note that in the example only one of the edges needs to be removed, not both. The method proposed in [18] deletes the edge that has the least impact on the number of node-disjoint paths from S to T.

That is, for every edge (x, y) that crosses other edges, the edge is removed and Stint is run on the remaining graph. The edge whose removal produces the largest number of disjoint paths is chosen for permanent removal. The process is repeated until no two edges cross each other. A more detailed description may be found in [18].

Note that this is a computationally intensive algorithm in the case of a dense sensor network where many edges cross each other. This is because if there are m number of edges that cross other edges, then Stint has to be run m times, which results in a graph with $m - 1$ crossing edges. This in turn requires Stint to be run $m - 1$ times, and so on.

D. Ordered Ceilings

In [19], we presented a heuristic which outperforms those presented in [17] [18]. We briefly overview it here, as it is the foundation for our improved heuristic in Section III.

Consider Figure 3(a), and note that a sensor barrier B divides the area of interest into an *upper region* and a *lower region*. The *ceiling* of B consists of all points p along the border of the sensing radius of each sensor in B such that one can travel from p to any point in the upper region without crossing the sensing area of any sensor. The *floor* of B is defined similarly but with respect to the lower region. Consider again the sensor barrier depicted in Figure 3(a). Its ceiling and floor are depicted in Figure 3(b).

Our heuristic finds each barrier iteratively as follows. Consider the set of all sensor nodes as a single barrier, and obtain its ceiling. The first barrier consists of all sensor nodes that take part of this ceiling. These nodes are then removed from the network. The remaining sensor nodes are again considered as a single barrier, and their ceiling is found, which in turn yields the second barrier, etc..

For example, consider Figure 3(c). The first barrier, B_1, is obtained from the ceiling of all the sensor nodes. The next barrier, B_2, is obtained by removing B_1 and obtaining the ceiling of the remaining nodes. B_3 is found the same way.

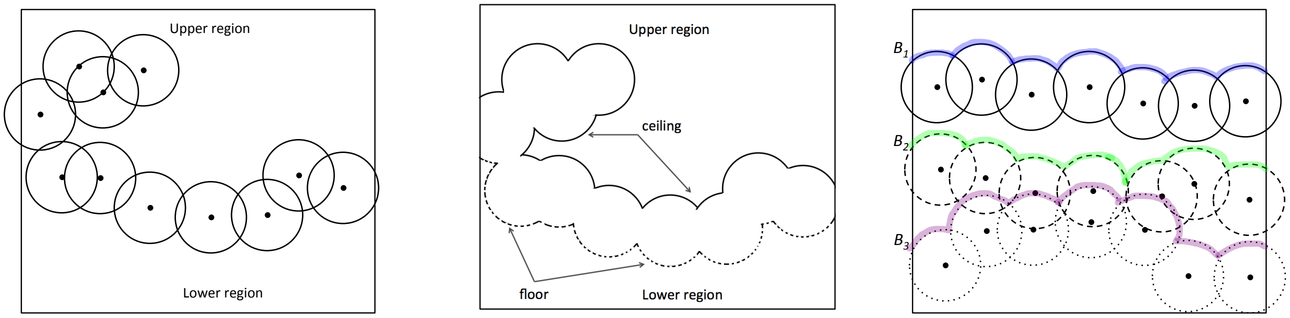
To obtain the ceiling of a set of nodes, one simply begins with the top-most node that intersects the left border of the area. Let b_0 be this node, and p_0 the top-most point where b_0 intersects the left border. Point p_1 is the *first point, clockwise from* p_0, where the sensing area of b_0 intersects the sensing area of another node b_1. This point becomes p_1. This process continues until either the right border is reached, and thus the barrier is complete, or it returns to the left border, in which case the nodes are discarded since they do not take part of any barrier.

We refer to our heuristic as the *ordered-ceilings* heuristic, and is presented in more detail in [19].

III. Compressed Ceilings Heuristic

We next present our enhancement to the above heuristic, which we refer to as the *compressed ceilings* heuristic. Consider the barrier in Figure 4(a). Assume that it was obtained using the ordered ceilings heuristic. As the barrier is constructed from left to right, the algorithm runs into node i. The next clockwise node after i is j. Similarly, k follows j clockwise. Note that the next clockwise node after k is i again. In this case, all three nodes j, k, and l, serve no practical purpose. We call these nodes a *detour* because they simply return to the original node. We can remove these nodes and still maintain a sensor barrier.

Consider now nodes x, y, and z. Again, the next clockwise node after x is y, followed clockwise by z. Note, however, that



(a) Sensor Set

(b) Ceiling and Floor

(c) Barriers Obtained

Figure 3. Ceiling-First Method

y is not needed for the integrity of the barrier. Nonetheless, our earlier heuristic will have included it in the barrier. Thus, y can be removed. We refer to y as a *spurious* node, because it connects two nodes which are already connected.

Spurious nodes and detours could be removed as the barrier is constructed, or doing a pass over the barrier once its construction finishes. We choose the latter for its simplicity. The nodes that we remove as described above are not deleted permanently from the network. Simply deleting them would serve no purpose at all. Instead, the removed nodes remain available in the pool of potential sensors for the construction of the next barrier. By doing so, we increase the likelihood of being able to construct more barriers in the iterations that remain in the algorithm.

Note that the savings can be significant, in particular, if we consider networks that are very dense, i.e., with many sensors per unit of area. Consider a sequence of sensors that are very close to each other. The location of each sensor in the sequence is only a slight amount to the right of the previous one in the sequence. In this case, a large number of spurious nodes will exist, which can be removed and then reclaimed for the next barrier created by the heuristic.

However, removing nodes to be later reclaimed by other barriers does have its perils: barrier breaches may occur. Consider Figure 4(b). Two barriers are shown. The top barrier (i.e., scheduled first) has circles with solid lines. The lower barrier (i.e. scheduled second) has circles with dashed lines. The top barrier has a detour, and the sensors involved in the detour are marked with thicker lines. Once the detour nodes are removed, the results are shown in Figure 4(c). Note that the lower barrier *does not* have a detour because, although close, the same node is not visited twice. Because of this, the location marked by the star is actually a barrier breach. That is, an intruder can move to this location while the first barrier is active, and once the switch to the second barrier occurs, the intruder is free to move to the lower part of the area.

One method to avoid this barrier breach is to check after the completion of barrier number m whether there is a barrier breach with the previous barrier $m-1$. If so, all the nodes of barrier number m could be removed from the network. This, however, is somewhat drastic because perhaps only a few nodes of the barrier are involved in the breach, and removing the whole barrier is unnecessary.

Instead, as each node is added to the barrier, we can check if adding this node will cause a barrier breach, and if so, this individual node is removed from the network. Again, the node is not permanently removed, it is just set aside and possibly reused in the construction of subsequent barriers.

To detect if a node is causing a barrier breach, we use the following result which we proved in [19].

> *An ordered pair of barriers* (B_1, B_2) *has a breach iff the floor of* B_2 *intersects the ceiling of* B_1.

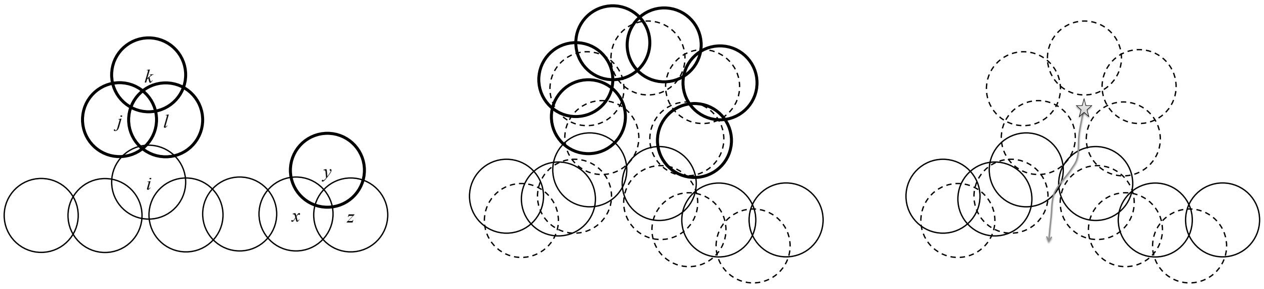
Figure 5(a) shows the first three nodes of a barrier, and also the left border of the area of interest. Note that the contribution of node 2 to the ceiling or floor of the barrier is not defined until node 3 is chosen. Thus, in our heuristic, when node $i+1$ is chosen, then the arc that node i contributes to its floor can be checked against the ceiling of the previous barrier, and see if they intersect, as shown in Figure 5(b). If so, then node i is removed from the current barrier and no longer considered a candidate for this barrier, while node $i+1$ is returned to the set of candidate nodes for this barrier.

Figure 5(a) also shows the *ceiling points* and the *floor points*. These are the points in the ceiling (respectively floor) where the sensing areas of two consecutive sensors intersect plus the points where the ceiling (respectively floor) intersects the left or right border of the area of interest. We make use of these definitions in the pseudocode of our heuristic, which is presented in Algorithm 1.

IV. Breach-Free Stint

In addition to the heuristics of [18] [19], we consider a simple modification to the Stint protocol [11]. The heuristics of [18] [19] pre-process the sensor graph before running the Stint algorithm in a manner such that breaches are avoided. We take the opposite approach of first using Stint to obtain the maximum number of barriers, which of course contain a significant number of barrier breaches, and then we eliminate the barrier breaches using the intersection of the floor and ceiling of consecutive barriers.

Thus, after we run Stint, we sort the barriers in order of their intersection with the left wall of the area of interest. If there were no barrier breaches, then the barriers would be scheduled from top to bottom. To check for breaches, we

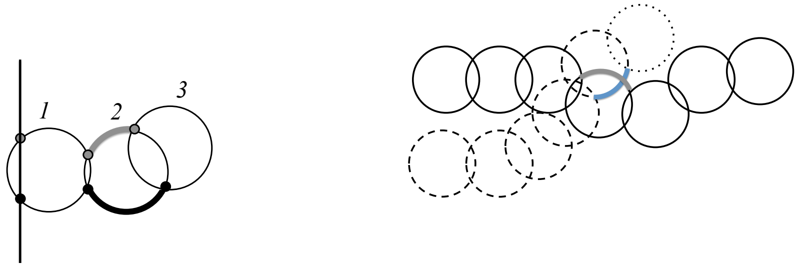


(a) Sensors to be removed.

(b) Potential breach after removal.

(c) Breach created.

Figure 4. Compressed Ceilings Method



(a) Node contribution to ceiling and floor.

(b) Detecting a barrier breach.

Figure 5. Preventing Barrier Breaches

consider each pair (B_i, B_{i+1}) of consecutive barriers. If the floor of B_{i+1} intersects the ceiling of B_i, then B_{i+1} is removed from the network. We keep repeating this until all barrier breaches are removed.

V. Simulation Results

In this section, we compare the performance of our Compressed Ceilings (CC) heuristic presented in Section III against several other heuristics: Ordered Ceilings (OC) that we presented in [19] and overviewed in Section II-D, Max-Flow-Edge-Eraser (MFEE) [18] that we overviewed in Section II-C, and Breach-Free Stint (BFS) that we presented above. We use pure Stint as an upper bound on these heuristics, since it is oblivious to barrier breaches. We only compare against the MFEE heuristic and not against other heuristics in [17] [18] as it was shown in [18] that MFEE is clearly superior to the others.

The area of interest is a square of size 100×100 meters. We also simulated rectangles of dimensions 60×100, 80×100, 100×60, and 100×80 meters, but the results were similar. Also, n sensor nodes are randomly deployed in each area, where n ranges from 30 to 80, and the sensing radius ranges from 15 meters to 25 meters. Every point in our plots corresponds to the average of 100 simulations.

Figure 6 shows our first comparison, where the number of sensor ranges from 30 to 80. It shows the results for three different sensing ranges r: 15, 20, and 25. Our CC heuristic clearly outperforms all others, being more noticeable as the number of sensor nodes increases or as the radius increases. I.e., as the density of sensor nodes increases, the CC heuristic improves its margin over the other heuristics.

Our second group of simulations are shown in Figure 7. The transmission range varies from 15 to 20, and the number of nodes is 40, 50, or 60. Again CC outperforms all other heuristics. The difference becomes more significant as the number of sensors increases from $n = 40$ in Figure 7(a) to $n = 60$ in Figure 7(c).

A curious observation in both of these figures is that BFS begins to outperform some of the heuristics as the sensor density increases. For example, in Figure 6, BFS begins to outperform MFEE when $r = 20$, and significantly outperforms it when $r = 25$. A similar pattern can be observed in Figure 7 as the number of nodes increases.

In order to explore this phenomenon, we took Figure 6 and extended the number of sensors dramatically, up to 600 sensors. The results are shown in Figure 8.

First note that our CC heuristic significantly outperforms our previous OC heuristic. In addition, in Figure 8(a), where $r = 15$, BFS begins to approach the performance of our earlier OC heuristic. In Figure 8(b), where $r = 20$, BFS outperforms OC staring around 300 nodes which yields about 20 barriers. Finally, in Figure 8(c), where $r = 25$, BFS outperforms even our current heuristic CC, again starting around 300 nodes which yields about 45 barriers. Thus, in very dense networks, a simple variation of the Stint method can actually produce more breach-free barriers than any other current heuristic.

Algorithm 1 ***Compressed-Ceilings*** (N)
Inputs: sensor node set N.
Output: set O of node-disjoint breach-free sensor barriers.

```
$O \leftarrow \emptyset$;
$N' \leftarrow N$;
while exists a sensor in $N'$ whose range crosses the left edge of the area do
    $Barrier \leftarrow$ top sensor overlapping left edge of area;
    $lastBarrier \leftarrow \emptyset$;
    $done \leftarrow$ false;
    $success \leftarrow$ false;
    while $\neg done$ do
        $s \leftarrow lastSensor(Barrier)$;
        $p \leftarrow$ last point in the ceiling of $Barrier$;
        $Q \leftarrow \{q \mid [\exists\, t \in N', q \in (\text{range}(s) \bigcap \text{range}(t))]\}$;
        $p' \leftarrow$ first point in $Q$ clockwise from $p$ around $s$;
        $s' \leftarrow$ sensor whose range overlaps that of $s$ at $p'$;
        if range of $s'$ overlaps the area's left edge then
            $success \leftarrow$ false;
            $done \leftarrow$ true;
        end if
        $Barrier \leftarrow append(Barrier, s')$;
        if $lastBarrier \neq \emptyset \wedge |Barrier| \geq 2 \wedge intersect(floor(Barrier), ceiling(lastBarrier)$ then
            $q \leftarrow next\text{-}to\text{-}last(Barrier)$;
            $N' \leftarrow N' - q$;
            $Barrier \leftarrow Barrier - \{s', q\}$;
        else
            if range of $s'$ overlaps the area's right edge then
                $success \leftarrow$ true;
                $done \leftarrow$ true;
            end if
        end if
    end while
    if $success$ then
        $O \leftarrow O \bigcup Barrier$;
    end if
    $N' \leftarrow N' - Barrier$;
end while
return $O$
```

VI. Concluding Remarks and Future Work

In this paper, we have refined our previous heuristic to deliver a higher level of performance by carefully eliminating redundant nodes. We have also presented a simple modification of the Stint algorithm that results in a method that outperforms all others when the density of sensors nodes is very high. This points to the possibility that that there is still room for improving our earlier heuristics, because the Stint protocol is oblivious to breaches, and our modification to Stint is straightforward by eliminating any barrier in its output that causes a breach with a previous barrier.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks, vol. 38, no. 4, Mar 2002, pp. 393–422.

[2] C. Huang and Y. Tseng, "The coverage problem in a wireless sensor network," in ACM Int'l Workshop on Wireless Sensor Networks and Applications (WSNA), 2003.

[3] H. Zhang and J. Hou, "On deriving the upper bound of -lifetime for large sensor networks," in Proc. of The 5th ACM Int'l Symposium on Mobile Ad-hoc Networking and Computing (MobiHoc), 2004.

[4] Cardei, M., Thai, M.T., Y. Li, and W. Wu, "Energy-efficient target coverage in wireless sensor networks," in INFOCOM 2005, 24th Annual Joint Conference of the IEEE Computer and Communications Societies., vol. 3, March 2005.

[5] M. Thai, Y. Li, and F. Wang, "O(log n)-localized algorithms on the coverage problem in heterogeneous sensor networks," in IEEE Int'l Performance, Computing, and Communications Conference, 2007. IPCCC 2007., April 2007, pp. 85–92.

[6] S. Gao, X. Wang, and Y. Li, "p-percent coverage schedule in wireless sensor networks," in Proc. of 17th Int'l Conference on Computer Communications and Networks, 2008. ICCCN '08., Aug 2008, pp. 1–6.

[7] C. Vu, G. Chen, Y. Zhao, and Y. Li, "A universal framework for partial coverage in wireless sensor networks," in Performance Computing and Communications Conference (IPCCC), 2009 IEEE 28th Int'l, Dec 2009, pp. 1–8.

[8] Y. Li, C. Vu, C. Ai, G. Chen, and Y. Zhao, "Transforming complete coverage algorithms to partial coverage algorithms for wireless sensor networks," Parallel and Distributed Systems, IEEE Transactions on, vol. 22, no. 4, April 2011.

[9] S. Kumar, T. Lai, and A. Arora, "Barrier coverage with wireless sensors," in Proc. of the 11th Annual Int'l Conference on Mobile Computing and Networking (MobiCom), 2005.

[10] A. Saipulla, C. Westphal, B. Liu, and J. Wang, "Barrier coverage of line-based deployed wireless sensor networks," in INFOCOM 2009, IEEE, April 2009, pp. 127–135.

[11] S. Kumar, T. Lai, M. Posner, and P. Sinha, "Maximizing the lifetime of a barrier of wireless sensors," Mobile Computing, IEEE Transactions on, vol. 9, no. 8, Aug 2010.

[12] H. Yang, D. Li, Q. Zhu, W. Chen, and Y. Hong, "Minimum energy cost k-barrier coverage in wireless sensor networks," in Proc. of the 5th Int'l Conf. on Wireless Algorithms, Systems, and Ap- plications (WASA), 2010.

[13] H. Luo, H. Du, D. Kim, Q. Ye, R. Zhu, and J. Zhang, "Imperfection better than perfection: Beyond optimal lifetime barrier coverage in wireless sensor networks," in Proc. of The IEEE 10th Int'l Conference on Mobile Ad-hoc and Sensor Networks (MSN 2014), Dec 2014.

[14] D. Li, B. Xu, Y. Zhu, D. Kim, and W. Wu, "Minimum (k,w)-angle barrier coverage in wireless camera sensor networks," Int'l Journal of Sensor Networks (IJSNET), 2014.

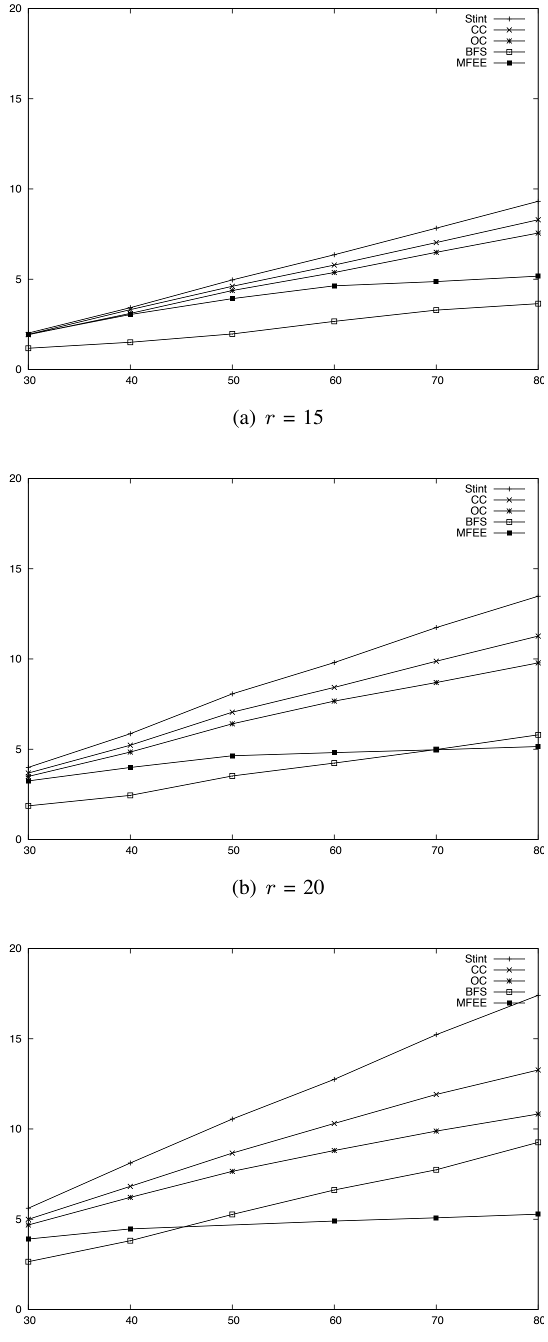
[15] L. Guo, D. Kim, D. Li, W. Chen, and A. Tokuta, "Constructing belt-barrier providing quality of monitoring with minimum camera sensors," in Computer Communication and Networks (ICCCN), 2014 23rd Int'l Conf. on, Aug 2014.

[16] B. Xu, D. Kim, D. Li, J. Lee, H. Jiang, and A. Tokuta, "Fortifying barrier-coverage of wireless sensor network with mobile sensor nodes," in Proc. of the 9th Int'l Conference on Wireless Algorithms, Systems, and Applications (WASA 2014), Jun 2014.

[17] D. Kim, J. Kim, D. L. abd S. S. Kwon, and A. Tokuta, "On sleep-wakeup scheduling of non-penetrable barrier-coverage of wireless sensors," in Proc. of the IEEE Global Communications Conference (GLOBECOM 2012), Dec 2012, pp. 321–327.

[18] D. Kim, H. Kim, D. Li, S.-S. Kwon, A. O. Tokuta, and J. A. Cobb, "Maximum lifetime dependable barrier-coverage in wireless sensor networks," Ad Hoc Networks, vol. 36, no. 1, Jan 2016.

[19] J. A. Cobb, "Improving the lifetime of non-penetrable barrier coverage in sensor networks," in IEEE 14^{th} International Workshop on Assurance in Distributed Systems and Networks (ADSN), Jun 2015, pp. pp. 1–10.

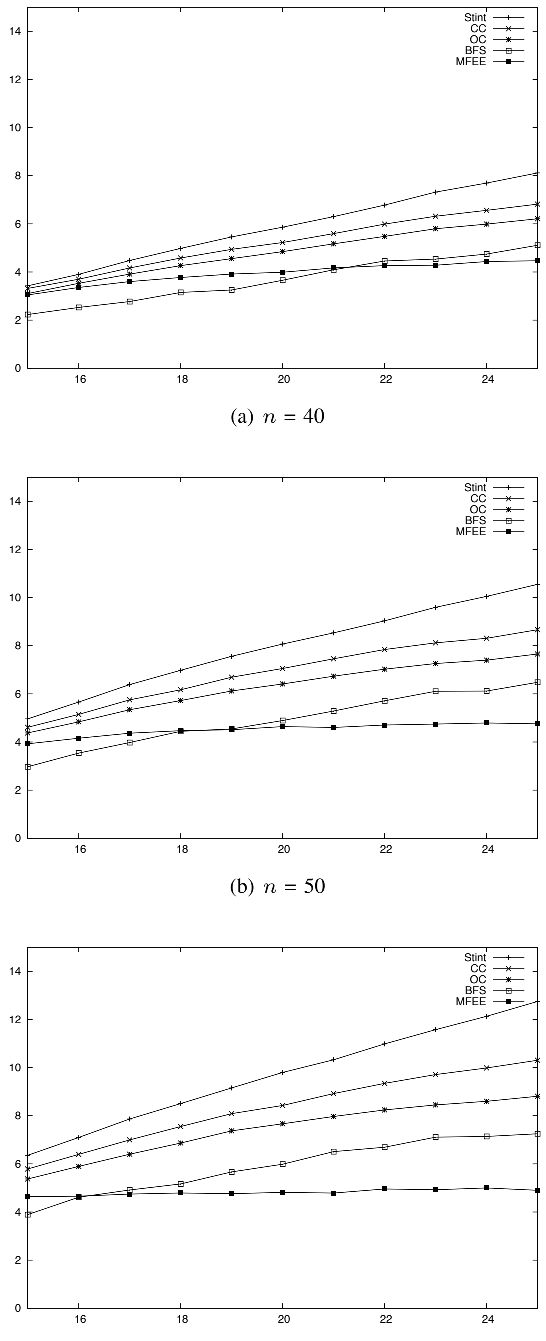


(a) r = 15

(b) r = 20

(c) r = 25

Figure 6. Number of sensors (horizontal) vs. number of barriers (vertical) in 100×100 region



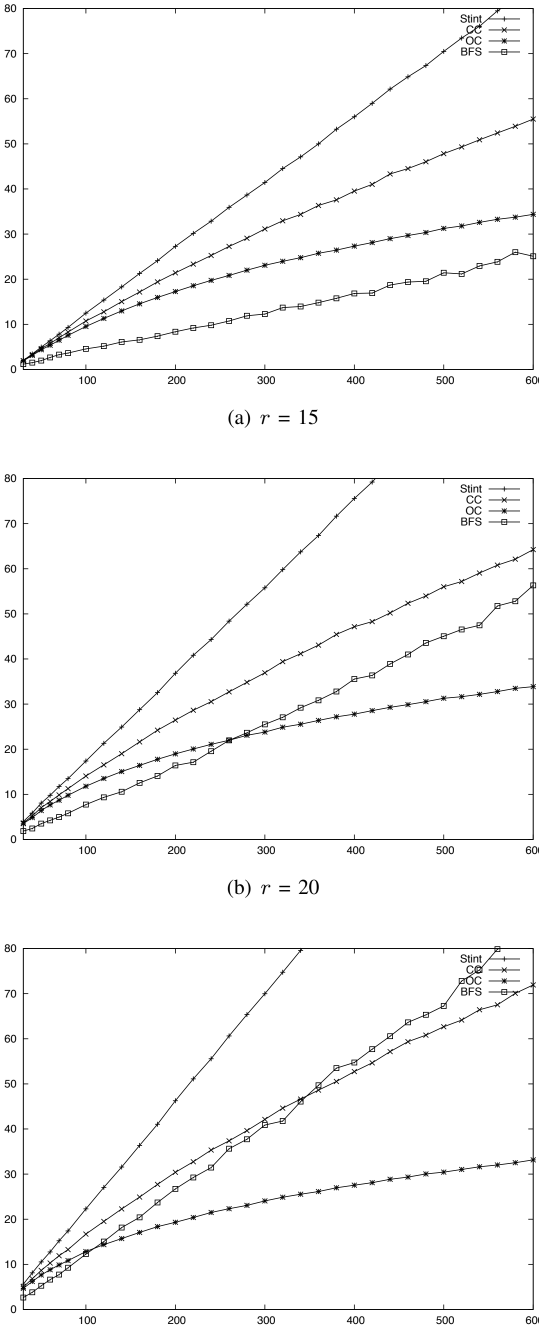
(a) n = 40

(b) n = 50

(c) n = 60

Figure 7. Transmission range (horizontal) vs. number of barriers (vertical) in 100×100 region

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-499-2



(a) r = 15

(b) r = 20

(c) r = 25

Figure 8. Number of sensors (horizontal) vs. number of barriers (vertical) in 100×100 region

GitStud: A Web-based Application for Sharing Project Files

Mohammad Bsoul, Emad E. Abdallah, Qais Saif, Ibrahim Albarghouthi, Mohammed Albeddawi, Essam Qaddurah
Faculty of Prince Al-Hussein Bin Abdullah II for Information Technology
The Hashemite University email:mbsoul@hu.edu.jo
Zarqa, Jordan

***Abstract*—The aim of this paper is implementing a web-based application for facilitating the communication of the members of the same project (assignment) group. Most of the group work needs to be organized and shared between teammates to enable team working and collaboration, which makes it easier and faster to accomplish tasks. Our contribution to this field is a web-based application that grants access to a repository with complete history of all files regarding a specific project. This web-based application named GitStud is expected to make it easier for students who work in groups to organize their work and keep track of all files that are being committed up-to-date. GitStud is based on sharing directories, which makes storing files from local to global. We used Git version control software to implement our web-based application. Git version control software is considered an open source, distributed version control system that can handle projects of different sizes.**

Keywords–Git; web-based application; Students; Instructors; Projects.

I. Introduction

The students who work on the same project might face difficulties in communicating with each other. Team work needs cooperation between teammates, and this could be done better by sharing the project files with other teammates. The process of sharing files requires a shared repository which provides everyone with an access to the project files.

The files related to a project should be viewed by all team members to see and edit as needed. Git version control software [1] allows the user to track the history of a collection of files and includes the functionality to revert a file back to its previous version. Each version captures a snapshot of the file at a certain time. The collection of files belonging to the same project files and their complete history are stored in a repository. This way, losing files or any conflict in the development process will be avoided. There is a number of hosting services Git repositories such as Assembla, Beanstalk, Bitbucket, CloudForge, Codebase, Fog Creek Kiln, GitHub, GitLab, Planio, Perforce, RhodeCode, and Unfuddle [2].Git development began in 2005. Git was originally designed as a low-level version control system engine on top of which others could write front ends. Then, the core Git project has become a complete version control software that can be used directly. However, the existing version control software is not dedicated for educational purposes where the intended users are students and instructors. This kind of software is needed to facilitate the communication between students of the same group and between students and instructors.

In this paper, we implemented a web-based application named GitStud and which is based on Git. GitStud provides a web-based graphical interface, which is much easier to work with than Git which is strictly a command-line tool. GitStud is expected to help improve collaboration and eases the communication process between the project members.

The rest of this paper is structured as follows. Related works are presented in Section II. Section III describes the new web-based application. In Section IV, we show some screen results from the web-based application. Finally, Section V concludes the paper and describes future work.

II. Related works

In this section, we will discuss the differences between our implemented GitStud and other version control software (including Git that we used in our implemented GitStud).

Bazaar [3] can be used by either a developer that is working on many branches which have local content or by groups sharing work through a network. Bazaar is free software written in Python programming language.

Darcs [4] is a distributed version control system that has many features such as the ability to select which changes to accept, and interaction with local or remote repositories.

Mercurial [5] is a cross-platform and distributed revision control tool that can be used by software developers and It is written in Python programming language. Mercurial is a command line program.

Revision Control System [6] is a software that automates the storing, retrieval, logging, identification, and merging of changes. The Revision Control System is useful for text that is modified frequently.

Apache Subversion [7] is a software for maintaining current and historical versions of files.

StarTeam [8] is employed in software development, especially when a project involves many teams in various places.

Code Co-op [9]is a system that employs peer-to-peer architecture to share projects and to control modifications to files. Code Co-op replicates its database on every device in the project.

PTC Integrity [10] uses a client/server model and allows software developers to track their work.

Git has many features over other version control software. One of its features is its branching model. In this model, it is allowed to have multiple local branches that are entirely independent of each other. The creation, merging, and deletion of these lines of development only needs seconds. Additionally, Git treats the stored data as a stream of snapshots. As a result, when a project is committed, or saved in Git, it takes a snapshot of what all the files look like at that moment and adds a reference to the snapshot. If there is no change on the file, Git does not store the file again, but only adds a link to the previous identical file.

None of the above version control software is dedicated for educational purposes. Therefore, there is a need for a software that facilitates the communication between students in the same project. Additionally, this software is needed to facilitate the communication between students and instructors who create assignments for these students.

The aim of our web-based application called GitStud is implementing a web-based application using Git and deploy it to ease the communication between the project members.

III. GitStud web-based application

This paper is about implementing a web-based application using Git. Its target is making the communication process easier between the students of the same project group (teammates). GitStud provides a user interface that is used by the group members to work on the files of the project (assignment). The provided user interface includes all the functionalities needed to work on the project posted by the instructor on GitStud.

GitStud offers a time saving solution for students to overcome team-working problems. GitStud supports the communication between students of the same group and between students and instructors by providing a global repository that is based on Git service.

GitStud requirements are:

- Web browser.
- Latest version of Java.
- Git software.
- Internet connection.

Figure 1 shows the workflow diagram of GitStud. One of the student (named Super Student) in the project group has first to login to his account in order to create a repository for the project. This created repository will contain the files that belong to the project. Then, all the students in this project group can pull the files in this repository in order to work on them and then push them back into the repository after they finish. Upon completion of the project, their work can be submitted to the instructor who posted the project (assignment) for evaluation.

Figure 2 shows the use case diagram, and Tables I to IX show detailed descriptions of use cases.

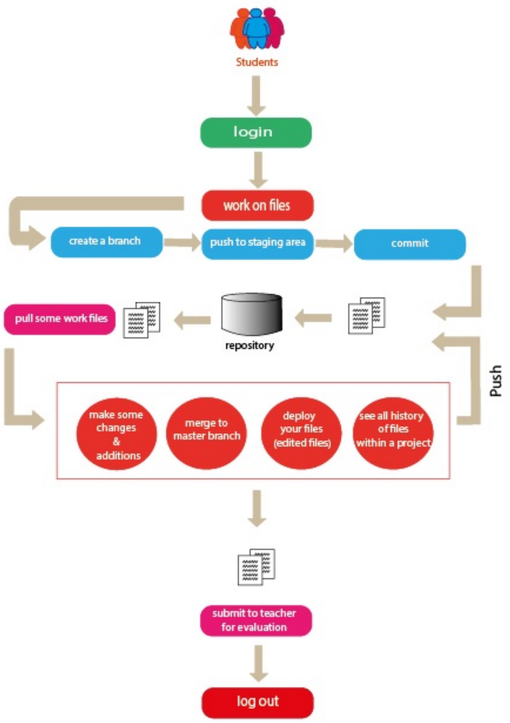

Students
login
work on files
create a branch
push to staging area
commit
pull some work files
repository
Push
make some changes & additions
merge to master branch
deploy your files (edited files)
see all history of files within a project
submit to teacher for evaluation
log out

Figure 1. Workflow diagram.

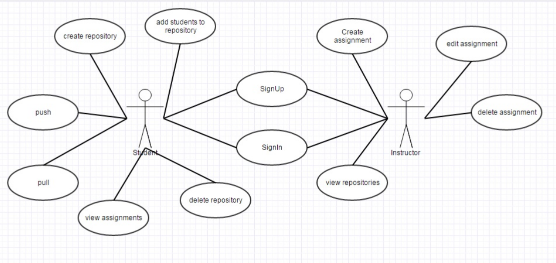


Figure 2. Use case diagram.

TABLE I. Sign up use case.

Name	Sign up.
Brief description	This use case is the basic step to enter the system. It ensures security and authentication for users.
Actors	Students, instructors.
Pre-conditions	The system should be up and running and the sign up form must be opened.
Main flow of events	A user opens GitStud and clicks the sign up button to open the sign up form. Then, the user chooses if he is a student or instructor. Next, the user provides the needed information such as the user name, e-mail, password and student ID/instructor ID and click on sign up button to confirm his information.
Alternative flow of events	A user opens GitStud and clicks the sign up button to open the sign up form. Then, the user provides the needed information. The information that the user entered appears that it already exists in the system such as the user name, e-mail and the student ID/instructor ID. If username, e-mail, or student ID/instructor ID exists, the user is prompted that he has an existing account.
Post-conditions	If the sign up process went successfully the system shows a message to the user to inform him that the sign up process was successful. Otherwise, the user will be informed to enter the required missing information or to correct the incorrect information.

TABLE II. SIGN IN USE CASE.

Name	Sign in.
Brief description	This step allows the user to access the system in order to use the features and functions of GitSud.
Actors	Students, instructors.
Pre-conditions	The user must have an existing account on GitStud to be able to access the system.
Main flow of events	A user enters his username and password in the sign in form. After successful sign in, the user will have access to GitStud's dashboard.
Alternative flow of events	A user enters incorrect username and password in the sign in form. In this case, the system notifies the user that his user name or password is incorrect.
Post-conditions	After signing in, GitStud's dashboard will appear and the user will have access to functions of the system.

TABLE III. CREATE REPOSITORY USE CASE.

Name	Create Repository.
Brief description	The creation of the repository allows students to have a shared data store to add their work files to.
Actors	Students.
Pre-conditions	Student must have an account to see this function.
Main flow of events	A student sign in. Then, he student chooses the create repository feature. Next, the student names the new repository. Then, the student sees the available courses. Next, the student chooses one of the courses in order to see the assignments for this course. If the student chooses an assignment, an empty repository is created for it. This student will become the super student for this repository and will have full control over it.
Alternative flow of events	A student chooses to create a new repository. This student has another repository with the same name. The student is prompted to rename the new repository. The student has to rename the new repository.
Post-conditions	The new empty repository is created and ready for use.

TABLE IV. PULL USE CASE.

Name	Pull.
Brief description	A pull request updates current local directory to up-to-date files in the repository.
Actors	Students.
Pre-conditions	A student must share a repository and the repository has work files to be pulled. The student must first clone the repository to a local directory. The student ensures that the local directory is not up-to-date.
Main flow of events	A student clicks on the pull button from the desktop application.
Alternative flow of events	A student tries to pull some files before cloning the repository. The student is asked to clone the repository first to be able to pull latest files.
Post-conditions	New work files are added to his local directory.

TABLE V. PUSH USE CASE.

Name	Push.
Brief description	The push feature adds modified files to the repository.
Actors	Students.
Pre-conditions	A student must share a repository. The student must first clone the repository to a local directory. Next, the student ensures that the local directory is modified. Then, the student has to commit changes.
Main flow of events	A student clicks the push button from the desktop application.
Alternative flow of events	A student tries to push some files before cloning the repository. In this case, the student is asked to clone repository first. The student must commit before push operation.
Post-conditions	The files are added to the repository with information about the commit activity.

TABLE VI. COMMIT USE CASE.

Name	Commit.
Brief description	The commit adds files that will be pushed to a repository.
Actors	Students.
Pre-conditions	A student has modified files and wants to add them to the repository.
Main flow of events	A student works on files on the local directory. Then, the student selects the files that are ready for the push process and adds a message with the commit.
Alternative flow of events	The student did not select the files to be committed. In this case, the student is asked to select the files first.
Post-conditions	The files are added to the staging area and are ready to be pushed.

TABLE VII. VIEW HISTORY USE CASE.

Name	View history.
Brief description	This feature allows user to trace full history of commits.
Actors	Students, instructors.
Pre-conditions	The user must have an access to the repository.
Main flow of events	The User selects the repository that he wants to view its history. The user clicks on view history.
Alternative flow of events	——————————
Post-conditions	The User can move between commits.

TABLE VIII. CREATE ASSIGNMENT (PROJECT) USE CASE.

Name	Create assignment.
Brief description	An instructor adds an assignment for a specific course so students can work on.
Actors	Instructors.
Pre-conditions	The instructor must have added at least one course.
Main flow of events	The instructor chooses a course then fills the assignment form. Next, he clicks on the upload button to upload the required assignment.
Alternative flow of events	The instructor did not fill the required fields. The instructor is notified to fill the empty fields.
Post-conditions	The students in the course are informed about the assignment.

TABLE IX. EDIT ASSIGNMENT (PROJECT) USE CASE.

Name	Edit assignment.
Brief description	An instructor modifies an assignment.
Actors	Instructors.
Pre-conditions	The instructor must have added at least one assignment to the course.
Main flow of events	The instructor chooses a certain assignment. Then, the instructor clicks on edit assignment button. Next, the instructor modifies the assignment and clicks on save.
Alternative flow of events	The instructor modifies a field by leaving it empty. The instructor is notified that he left a field empty.
Post-conditions	The students of the assignment are notified of the change made to the assignment.

Figure 3 shows the class diagram and the variables and methods of each entity (class). Super student entity represents the student (one of the project members) who is responsible for repository creation, deletion, and management.

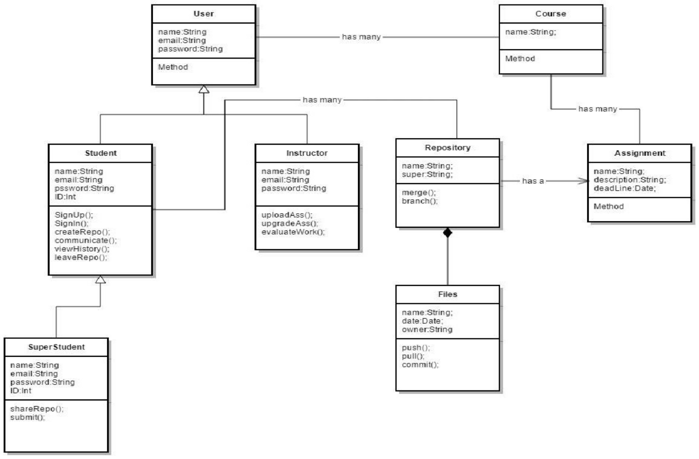
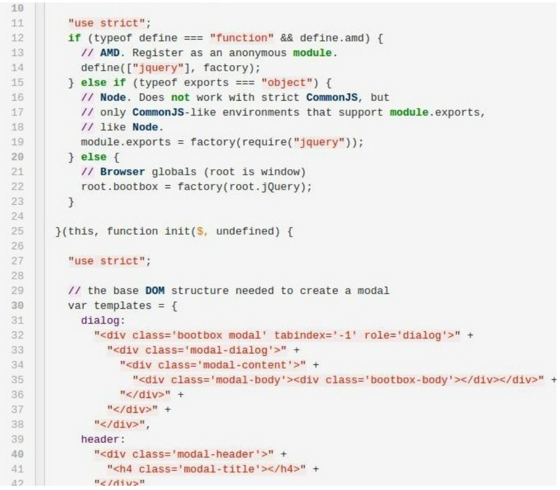


Figure 3. Class diagram.

IV. RESULT SCREENS

In this section, we show some screen results from our implemented web-based application.

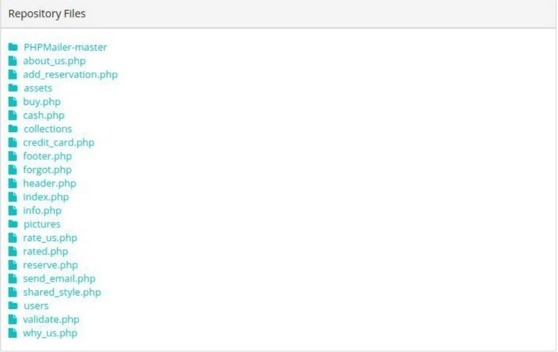
Figure 4 shows the content of one of the files belonging to the repository of an assignment (project) group.



```
  "use strict";
  if (typeof define === "function" && define.amd) {
    // AMD. Register as an anonymous module.
    define(["jquery"], factory);
  } else if (typeof exports === "object") {
    // Node. Does not work with strict CommonJS, but
    // only CommonJS-like environments that support module.exports,
    // like Node.
    module.exports = factory(require("jquery"));
  } else {
    // Browser globals (root is window)
    root.bootbox = factory(root.jQuery);
  }

}(this, function init($, undefined) {

  "use strict";

  // the base DOM structure needed to create a modal
  var templates = {
    dialog:
      "<div class='bootbox modal' tabindex='-1' role='dialog'>" +
        "<div class='modal-dialog'>" +
          "<div class='modal-content'>" +
            "<div class='modal-body'><div class='bootbox-body'></div></div>" +
          "</div>" +
        "</div>" +
      "</div>",
    header:
      "<div class='modal-header'>" +
        "<h4 class='modal-title'></h4>" +
      "</div>"
```

Figure 4. In file screen.

In Figure 5, it can be seen the files in one of the assignment repositories.



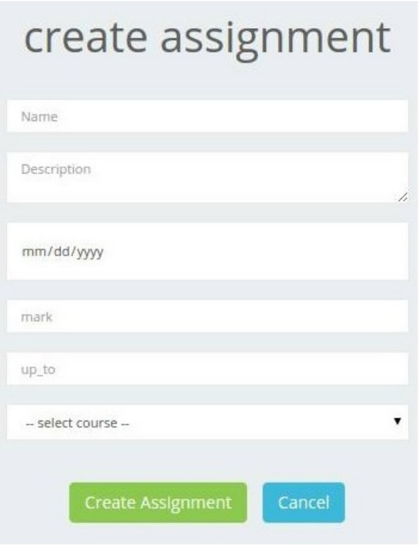
Repository Files

- PHPMailer-master
- about_us.php
- add_reservation.php
- assets
- buy.php
- cash.php
- collections
- credit_card.php
- footer.php
- forgot.php
- header.php
- index.php
- info.php
- pictures
- rate_us.php
- rated.php
- reserve.php
- send_email.php
- shared_style.php
- users
- validate.php
- why_us.php

Figure 5. Repository files screen.

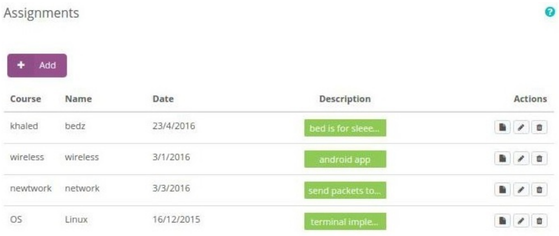
Figure 6 shows the screen that appears to the instructor when he wants to create a new assignment.

In Figure 7, it is shown the assignments created by one of the instructors.



create assignment

Figure 6. Create assignment screen.



Assignments

Course	Name	Date	Description	Actions
khaled	bedz	23/4/2016	bed is for sleee...	
wireless	wireless	3/1/2016	android app	
newtwork	network	3/3/2016	send packets to...	
OS	Linux	16/12/2015	terminal imple...	

Figure 7. Instructor assignments screen.

V. CONCLUSION

In this paper, we have implemented a web-based application named GitStud using Git.This web-based application has been designed to make it easier for students who work in groups to organize their work and keep track of all files that are being committed up-to-date.

GitStud allows students to track the history of a collection of files and includes the functionality to revert the collection of files back to the previous version. In this way, losing files or any conflict in work will be avoided. It is also expected to help improve collaboration and ease the communication process between the students of the same project group.

In future work, we plan to add e-learning feature where the students can watch the lectures posted by instructors. Additionally, we plan to support more languages. Finally, we plan to have user testing for GitStud to prove that it makes it easier for students to work collaboratively, enhance communication between them in team-work situations and work together on files.

REFERENCES

[1] https://git-scm.com, [retrieved: July, 2016].

[2] T. Gunther, "12 git hosting services compared," http://www.git-tower.com/blog/git-hosting-services-compared/, 2014, [retrieved: July, 2016].

[3] http://bazaar.canonical.com/en/, [retrieved: July, 2016].

[4] http://darcs.net/, [retrieved: July, 2016].

[5] https://www.mercurial-scm.org/, [retrieved: July, 2016].

[6] http://www.gnu.org/software/rcs/rcs.html, [retrieved: July, 2016].

[7] https://subversion.apache.org/, [retrieved: July, 2016].

[8] http://www.borland.com/en-GB/Products/Change-Management/StarTeam, [retrieved: July, 2016].

[9] http://www.relisoft.com/co_op/index.htm, [retrieved: July, 2016].

[10] http://www.ptc.com/application-lifecycle-management/integrity, [retrieved: July, 2016].