A publication of ISCA*:
International Society for Computers
and Their Applications

INTERNATIONAL JOURNAL OF COMPUTERS AND THEIR APPLICATIONS

TABLE OF CONTENTS

*** "International Journal of Computers and Their Applications is Peer Reviewed".**

Volume 27, No. 4, Dec. 2020 **ISSN 1076-5204**

International Journal of Computers and Their Applications

A publication of the International Society for Computers and Their Applications

ISCA Headquarters···•···278 Mankato Ave, #220, Winona, MN 55987···•···Phone: (507) 458-4517
E-mail: isca@ipass.net • URL: http://www.isca-hq.org

Deep Learning on Image Recognition - Feature Learning by Layers

Hasham Burhani *
RBC Capital Markets, Toronto, Ontario, Canada, M5J 2J5
Trent University, Peterborough, Ontario, Canada, K9L 0G2

Wenying Feng *
Trent University, Peterborough, Ontario, Canada, K9L 0G2

Gongzhu Hu *
Central Michigan University, Mount Pleasant, Michigan, 48859, USA

Abstract

In this paper, a deep learning neural network model is developed by introducing a new activation function that offers higher gradients and faster learning rate. In addition, a sparsity function is applied to the hidden layer of the network to simplify the network structure. Moreover, a technique that swaps the activation functions of fully trained network to logistic function is introduced to enhance system performance. The algorithms were evaluated using real-world data sets. The results show the efficiency of the new model.

Key Words: Machine learning; neural network; imagine classification; sparsity function; denoising autoencoder.

1 Introduction

Along with the rapid development of computing capability, Artificial Neural Networks (ANNs) have attracted more and more interests of researchers from different areas. ANNs are biologically inspired machine learning algorithms that are capable of approximating very complex functions. Networks that are considered to meet certain requirements (networks with at least 1-hidden layer) are in fact universal function approximators [6] as they are capable of approximating any function given enough samples of the required function.

Borrowing a central idea from the human brain that deals with multiple levels of representation of the sensory input, a deep learning computational model accepts a raw input and transforms them to a better representative feature space of the problem [10]. A few of the popular algorithms are Autoencoders [11, 19], Deep Belief Networks (DBNs) [9] and Deep Convolutional Networks (DCNs) [20]. Autoencoders follow a particular neural network topology that asks the network to reconstruct its input from a hidden representation. This way, it forces the network to learn abstract features of the input and to rethink the expectation. On the other side, DBNs can be viewed as a composition of simple networks that leads to a fast, layer-by-layer training procedure. Each layer extracts features from the output of a previous layer. As a result, features from the input data can be learned hierarchically. Different from Autoencoders and DBNs, DCNs represent architectures making the explicit assumption that the inputs are images, which allows the encoding of certain properties into the architecture and therefore make the forward function more efficient to implement and vastly reduce the amount of parameters in the network.

In the area of image recognition, deep learning using neural networks is a powerful tool. For recent work, we refer to [7, 14, 21] and the references therein. Most of the work applied same activation functions and focused on the accuracy of the system. A deep learning network using Denoising Autoencoders (DAE) for pre-training was proposed in [2]. However, the preliminary testing results only used one dataset, the popular MNIST dataset [16]. In this paper, we introduce the complete design and algorithms for the *Swapped Pre-training Activation Function* (SPAF) deep learning model that has three improvements from the regular neural network model in image classifications. First, it applies a new activation function during the pre-training that offers higher gradients and faster learning rate. Second, a sparsity function [13] is applied to the hidden layer of the network to balance the prediction accuracy and the network structure. Third, a technique that swaps the activation functions of fully trained DAE to logistic functions is implemented and tested. It is shown that SPAF-networks have higher accuracy on image classifications and are faster for the training stage compared to the traditional deep learning models by a uniform activation function. Furthermore, we focus on features learned layer-by-layer based on the replacements of the core functions in the system. The SPAF-network is analyzed for the features it learns with a logistic, ReLU and a custom activation function. Comparing to the majority work that emphasizes the overall performance of a deep learning system, investigation on the behaviour of each layer of the system during the operation would provide insight of the learning procedure.

We evaluate the performance of the SPAF-network using the Chars74k dataset [5]. Similar to MNIST the Chars74k dataset

*Emails: burhani@evolutis.ca; wfeng@trentu.ca; hu1g@cmich.edu

has images of the arabic numerals for classification. It however also has the English alphabet for classification tasks. This makes this dataset much more difficult than the MNIST dataset, by bringing the total possible classes to 62 vs the 10 total categories in MNIST.

The rest of the paper is organized as following. In Section 2, we discuss the SPAF-network. Implementation algorithms are given in Section 3. Section 4 presents performance comparison using the three sub-datasets of Chars74k. Results on feature learning by layers are presented in Section 5. Finally, some conclusions, limitations and potential future work are given in Section 6.

2 The SPAF-Network Model

The idea of Swapped Pre-training Activation Function (SPAF-network) was first proposed in [2] and applied to image recognition. The model is in the structure of a neural network with sigmoidal activation neurons throughout. Training the model follows two stages: unsupervised pre-training followed by the supervised training using the weights learned from the first part. Conventionally many types of learning algorithms have been used for the unsupervised pre-training stage. In the SPAF-network, a modified Denoising Autoencoder (DAE) is used for this purpose.

The modifications applied to DAE are on the selections of two key functions. The first is in regards to the activation function used in the hidden layer. Instead of using some common activation functions such as the sigmoid function, a modified Elliott activation function in the following form is applied.

$$\phi(x) = \frac{x}{\sqrt{1+x^2}} + 0.5. \tag{1}$$

The new activation function (1) has higher gradients near zero which assures that all the neurons are well saturated in the pre-training stage [2]. Another choice for the activation function is the Rectified Linear Unit (ReLU) defined by equation (2) [12]. The ReLU activation function provides the similar gradients near zero to the modified Elliott function. However it does not produce differences for negative inputs.

$$\phi(x) = \begin{cases} x & \text{if } x \geq 0, \\ 0 & \text{if } x \leq 0. \end{cases} \tag{2}$$

The second modification is on the consideration of a cost function that produces higher sparsity for the hidden layers. Sparsity represents the complexity of the system [13]. Sparsity formally is the number of zeros in a vector. Consider two vectors $x = [1,1,0,1,0,1,1]$ and $y = [0,0,1,0,1,0,0]$. If sparsity is measured on both, y would have a higher sparsity score, since it has more zeros. A simple method of measuring sparsity would be to simply count the number of zeros. In a neural network where the goal is to measure the sparsity in the hidden layer, a sequential code that would need to check every index in the hidden layer and count the number of zeros would be required. That however may be impractical and slow for large neural networks. Various functions such as the l^1 (or $L1$) and l^2 (or $L2$) norms for vectors have been used to measure network sparsity [13]. The cost function given by equation (3) has two considerations: reducing the reconstruction error and maintaining sparse representation in the hidden layers at the mean time. The first part is easily achieved by using a mean-squared error function (MSE). Minimizing the MSE leads to minimizing the reconstruction error. The second part is achieved by using a sparsity function that was applied previously [13].

$$f_{cost} = \sum_{i=0}^{N} (\phi(x)_i - y_i)^2 + \log(\sum_{i=0}^{M} (1 + |h_i|)^2), \tag{3}$$

where x is a sample from the dataset, $\phi(x)$ is the output of the network, y is its associated expected output value, h_i is the output from the ith-neuron in the hidden layer, N and M are the dimensions of y and the hidden layer $\mathbf{h}$ respectively. The combination of the two parts for the cost function produces a balance of minimizing error with a simpler network structure.

The technique of swapping activation functions during fine-tuning was first applied in the SPAF-Network. Conventionally the weights of the supervised network in the fine-tuning stage are constructed using all of the hidden layer representations generated in the unsupervised pre-training stage. The parameters that are borrowed are the weights, the biases and the activations of each hidden layer. In the SPAF-model, only the weights and biases are kept from the unsupervised pre-training stage. The activations used in the pre-training stage are all replaced with logistic activation functions. The network resembles the topology created in the pre-training stage, with the decoding weights on the topmost layer removed. In other words, the SPAF-network follows pre-training paradigm that trains each hidden layer with the input of the preceding layer. Encoding weights are kept, and the decoding weights are removed once pre-training is complete.

The reasoning behind this specific technique is as follows: since most of the feature learning currents are in the pre-training stage, the process is accelerated by introducing an activation function with higher gradient to quickly saturating the neurons. Once the neurons are saturated, the logistic activation function is used to only fine-tune the weights in a smooth manner for classification. The goal is to reduce the classification error given the pre-trained parameters. Since the pre-trained parameters have learned the abstract features, this procedure has the advantages towards achieving a better classification result. The SPAF-Network offers higher gradients with no algorithmic or topological differences in comparison to a traditional learning model. The process keeps a constant algorithm complexity as a conventional learning model using neural networks.

3 Algorithm for Implementation

The implementation stage considers the network structure and algorithm design. The program was coded in Python to take advantage of Theano, a library that enables automatic

differentiation [1, 17]. This simplifies the backpropagation algorithm in regards to computing gradients with respect to the parameters in the network. The denoising autoencoder was designed to use the transpose of the weight matrix W for decoding. This produces better features and also simplifies the design of the autoencoder.

The model is designed as a 200-200 network: two hidden layers with 200 neurons each. It has 28×28 input neurons (corresponding to the number of pixels in each image). A subset of 50,000 from the training dataset is used to train the network, another 10,000 is kept for validation purposes. The final test set is used to measure the performance of the network. A learning rate of 0.1 is used for both the supervised and unsupervised training stages. The pre-training structure of the network is shown in Figure 1.

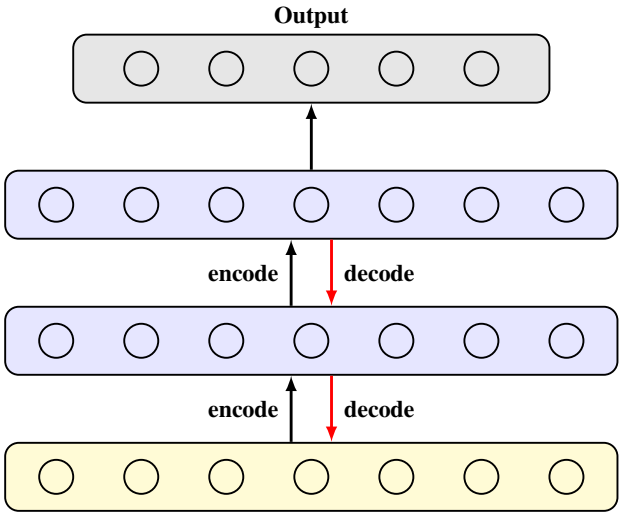


Figure 1: Pre-training structure of the SPAF-network model

The overall program can be divided into two distinct parts, the unsupervised pre-training using autoencoders, and the final fine-tuning using a regular feedforward neural network. The second part borrows parameters from the first. The final algorithms for the two modules are given in Algorithm 1 and Algorithm 2, respectively. In Algorithm 1, W is the weight matrix for decoding, γ is the pre-training learning rate, x is the input vector, y is the expected output. Similar notations are used in Algorithm 2. More details for the implementations of an autoencoder and feed-forward neural networks can be found on the Theano website [17].

4 Datasets and Testing

To test the efficiency of the newly developed algorithms, four datasets: MNIST, and the three subsets (Bmp, Hnd, and Fnt) from the Chars74k dataset [5] are applied.

As one of the most popular datasets for neural networks, the MNIST dataset represents real-world handwritten digits that were collected from the Census Bureau employees and high school students [16]. More details on the MNIST and some preliminary testing results for the SPAF-Network using MNIST was shown in [2]. In the following, we present comparisons from the standard comprehensive datasets of Chars74k, which has 74K images including the following

Algorithm 1: Pre-training algorithm

Data: $TrainingData = [(x,y)...,(x,y)]$ where x is an input vector and y is its corresponding 1-hot category vector.

begin
- $\gamma \leftarrow pretraining - learningrate$
- **for** $hiddenlayer \in HiddenLayers$ **do**
  - $W \leftarrow hiddenLayer.W$
  - $b \leftarrow hiddenLayer.b$
  - $c \leftarrow hiddenLayer.c$
  - Initialize W
  - $b \leftarrow 0$
  - $c \leftarrow 0$
  - **for** $epochs = 0; epochs < 10; epochs++$ **do**
    - **for** $miniBatch \in TrainingData$ **do**
      - $x \leftarrow sample[0]$
      - $y \leftarrow sample[1]$
      - $h \leftarrow Encode(W,b,x)$
      - $\hat{x} \leftarrow Decode(W^T,c,h)$
      - $cost \leftarrow f_{error}(\hat{x},x) + logPenalty(h)$
      - $\nabla W, \nabla b, \nabla c, \leftarrow (\nabla W, \nabla b, \nabla c) +$ $gradients(cost, w.r.t = (W,b,c))$
      - $W \leftarrow W + \gamma \nabla W$
      - $b \leftarrow b + \gamma \nabla b$
      - $c \leftarrow c + \gamma \nabla c$

(1) Bmp - images that were generated from natural images of various signs which were taken using a variety of cameras;
(2) Fnt - images of numbers and the alphabet generated from computer fonts;
(3) Hnd - images of handwritten letters generated through a tablet device.

The testing process follows two steps using the python library Pillow [3]. First, each image is processed by the sequence: (i) Reading image into program; (ii) Convert the image to greyscale; (iii) Color invert each image; (iv) Save image to disk. Second, images are converted to standard (X,Y) dataset format as input to the programs.

A total of 6 models divided into 2 groups are trained and compared for each of the three datasets. The first group is referred as SPAF-network, and the second as Sigmoidal group. The Sigmoidal group offers sigmoidal activation functions with no function swapping during fine-tuning. The SPAF-network employs the newly proposed activation function, and also incorporates function swapping to sigmoidal activations during the fine-tuning stage. Both groups are tested with three

Algorithm 2: Fine-tuning algorithm

Data: $TrainingData = [(x,y)...,(x,y)]$
Data: $ValidationData = [(x,y)...,(x,y)]$
begin
  $\gamma \leftarrow learningRate$
  $patience \leftarrow 10 \times traningBatches$
  $patienceIncrease \leftarrow 2$
  $improvementThreshold \leftarrow 0.995$
  $doneTraining \leftarrow false$
  $epch \leftarrow 0$
  $iter \leftarrow 0$
  while $epoch++ < 300$ **and** $!doneTraining$ **do**
    for $miniBatch \in TrainingData$ **do**
      $iter++$
      $\nabla\theta \leftarrow 0$
      for $sample \in miniBatch$ **do**
        $x \leftarrow sample[0]$
        $y \leftarrow sample[1]$
        $\hat{y} \leftarrow f_{net}(x,\theta)$
        $cost \leftarrow f_{cost}(\hat{y},y)$
        $\nabla\theta \leftarrow \nabla\theta + gradients(cost, w.r.t = \theta)$
      $\theta \leftarrow \theta + \gamma\nabla\theta$
  $validationScore \leftarrow TestModel(f_{net}, ValidationData)$
  if $validationscore > bestvalidationscore$ **then**
    $bestvalidationscore \leftarrow validationscore$
    if $validationscore <$
     $bestvalidationscore \times improvementthreshold$
     then
      $patience \leftarrow$
       $\max(patience, iter \times patienceincrease)$
  if $patience < iter$ **then**
    $doneTraining \leftarrow true$

variations for the cost function:

(1) No sparsity term in the cost function (the second term of equation (3));
(2) A traditional $L1$ sparsity term in the form $\sum_{i=0}^{|\mathbf{h}|}|h_i|$ to replace the second term of equation (3);
(3) The log penalty sparsity term given in equation (3).

We start with the Bmp dataset. Based on extensive tests on the network training, a learning rate of 0.2 for both pre-training and fine-tuning stages is selected since this value showed promising descend in cost. The dataset is randomly split 75/25 for training and test sets. Average classification errors from a total of 10 times running are shown in Figure 2.

The associated average activation values in each hidden layer are shown in Figure 3(a) and Figure 3(b). We can see that the L1-model offered higher sparsity at the expense of accuracy performance, and the log-penalty again decreased sparsity in both models except the second layer in the standard Sigmoidal-model.

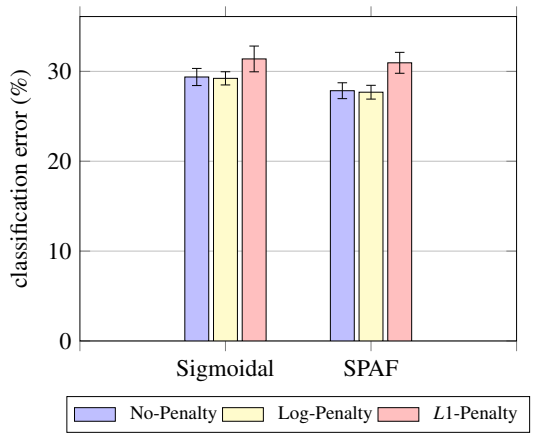
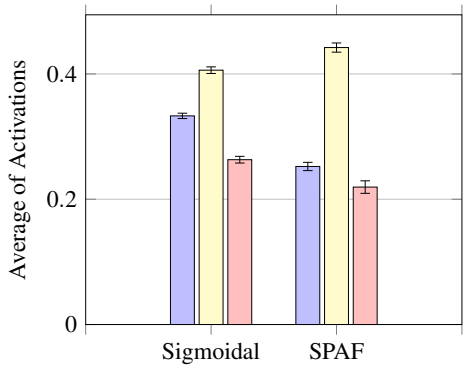
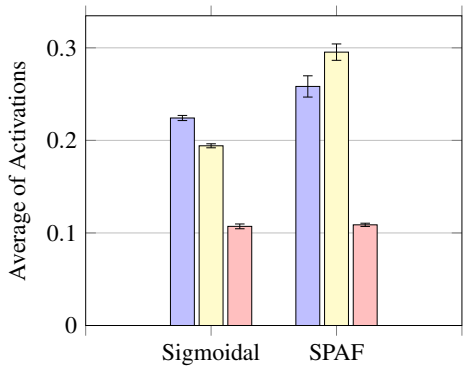


Figure 2: Classification errors of Sigmoidal-model vs SPAF on the Bmp Dataset



(a) First layer



(b) Second layer

Figure 3: Average activations of Sigmoidal activation function vs SPAF-network on the Bmp dataset

The Hnd dataset has a total of 3,410 samples collected through tablet pc from a total of 55 individuals [5]. Approximately 15 samples from each class were randomly

placed in a testing set, the remaining were placed in a training set. A total of 30 networks were trained in this manner to produce the classification error results shown in Figure 4. The results for Hnd dataset were not as pronounced as the previous two datasets discussed, both in performance and also sparsity induction. This may be related to the lack of samples for this particular dataset despite that the SPAF-network performed better on average than the standard Sigmoidal-model. However, it is noticed that the log-penalty model continued to increase sparsity in the second hidden layer of the Sigmoidal-model.

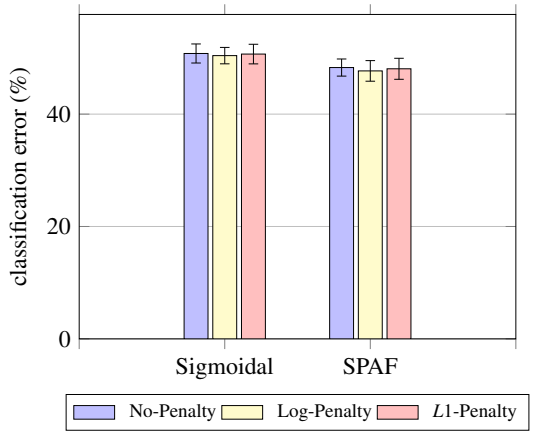


Figure 4: Classfication errors of Sigmoidal-model vs SPAF on the Hnd Dataset

The sparsity results in the first and second layers are shown in Figure 5(a) and Figure 5(b), respectively.

The Fnt dataset is the biggest. A total of 62,992 samples were collected from 254 different fonts in 4 styles [5]. Due to the higher classification complexity, two sets of experiments were conducted to test two models. First being the conventional sigmodial model with the normal logistic activation functions used in both pre-training and fine-tuning stages. The second model is the SPAF-network model. The models in the first experiment are trained for their classification performance over the number of neurons used in each hidden layer. The classification error results are plotted in Figure 6(a). In the second experiment, the models were tested for their performance given a percent of the overall dataset. All the models in this experiment utilized a network with two hidden layers of 200 neurons each. The results of the second experiment can be seen in Figure 6(b). Both models utilized the random sampling of approximately 15 samples from the dataset for the testing set.

Figure 6(a) shows that the proposed model starts at about the same result as the Sigmoidal-model at 50 neurons, but quickly outperforms it until about 700 neurons, where both models start to plateau to about the same performance average. The second experiment seen in Figure 6(b) shows that there is no real difference between the two models when we increase the number of training samples. This can be related to the fact that only 15 samples per class were used for testing, as the

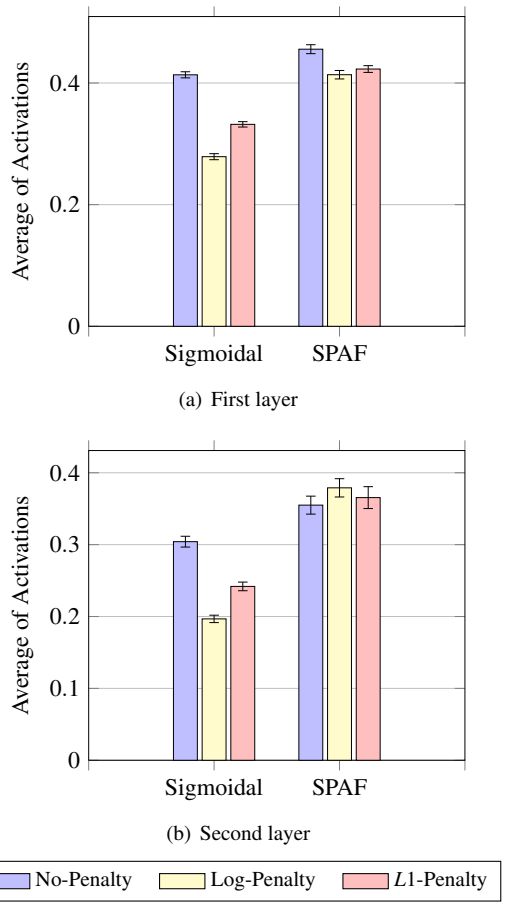


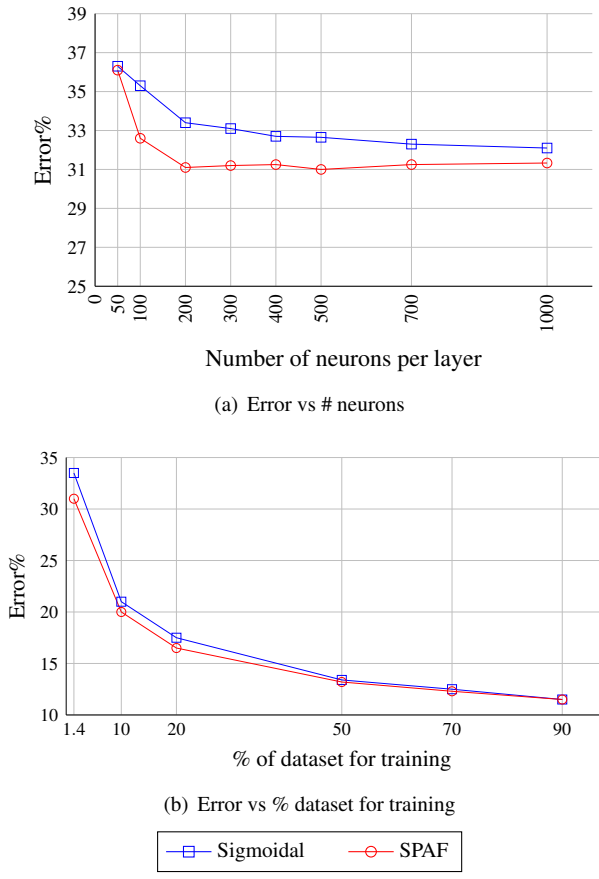
Figure 5: Average activations of Sigmoidal activation function vs SPAF-network on the Hnd dataset

two models approach the maximum possible performance, the difference in their learning ability diminishes.

In summary, in comparing the log based sparsity penalty and the $L1$ penalty on the hidden layer representation in the cost function, it is found that the log penalty offered no improvements in the first layer of the models, but showed modest improvements in the second layer of sigmoidal models. The reasoning behind it can be linked to the kind of features that are learned in layer one of the SPAF-model, that make layer 2 much more resilient to sparsity changes. The $L1$ penalty showed improvements over every layer, except for the second layer in the model with the new activation function trained on the Hnd Dataset. The log penalty did not affect the classification performance in any significant way, the $L1$ function on the other hand, increased classification error in more than one dataset as the reduction in average activation in the hidden layers was significant.

5 Feature Learning by Layers

We first explain a method of generating images from weights that are widely used in the industry [4, 8, 15, 18]. For a



(a) Error vs # neurons

(b) Error vs % dataset for training

Figure 6: Classification errors of Sigmoidal vs SPAF on the Fnt dataset

given image, weights that are positive, coming from a pixel can be thought of, as weights that 'like' the pixel. Given that a neuron has connections to every single pixel in the image, the weights can be thought of as the 'like' and 'dislike' variables of a neuron. Given a single layer, the process of converting the 'like' and 'dislike' variable intensities to pixel intensities for every neuron requires the conversion of every column in the weight matrix W to a row-major vector that follows the same technique as the original image. Given this vector, a matrix of pixel intensities can be created by first feature scaling the vector to values between $[0,1]$ using the formula $x_{scaled} = (x - X_{\min}) / (X_{\max} - X_{\min})$. This matrix can then be converted to an image using a generic library that can take pixel intensities in a matrix form. One such library that was used in this study is Pillow [3]. In this manner, values that are high mean particular pixel areas that the neuron likes in an image, and pixels that are dark are ones that the neuron dislikes. Considering the range of possible values that each scaled vector can take, the lower values get closer to 0, and the higher values get closer to 1. Absolute 0 meaning complete inhibition of that particular pixel, and 1 meaning excitation of that pixel. A collection of these can indicate the excitation of the neuron for the presence of a feature that is more complex than a pixel.

In our experiment, the first layer features were generated by using code in the Theano library. An algorithm was developed for generating matrices for layers above layer-1. The Python code of the algorithm is given in Listing 1.

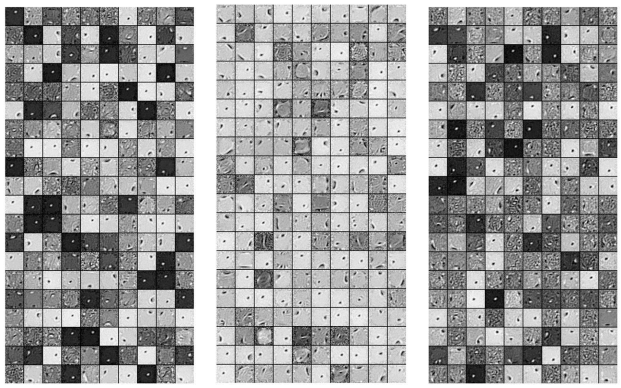
Listing 1: Python code of function for generating matrices

```
# vh is the visible-hidden matrix thereby
# representing a preceding hidden-hidden matrix.
# hh is the next level hidden-hidden matrix.
def drawWeights(vh,hh,layer) :
  new = vh.copy()
  new.fill(0)
  for master,k in enumerate(hh.T) :
    #tally up all visible unit connections
    #for this neuron.
    for idx,value in enumerate(k) :
       new[master,:]
       = new[master,:] + vh[idx,:] * value
    image = PIL.Image.fromarray(tile_raster_images(
            X=numpy.asarray(new), img_shape=(28, 28),
            tile_shape=(20, 10),tile_spacing=(1,1)))
  image.save(name+'layer-'+str(layer)+'.png')
  return new
```

We study reconstruction of images from weights using features learned across two SPAF-networks and the regular logistic DAE pre-trained based network. These three models are trained on three datasets from the previous experiments, namely MNIST, Fnt and Hnd. The same network topology of 200-200 and learning rate of 0.1 are applied. When trained on the MNIST dataset, the features learned in the first hidden layer of the three network configurations are shown in Figure 7. It is seen in the figure that the proposed function based SPAF-network produces more inhibitory and excitatory neurons than those learned by the regular Sigmoidal-network. The ReLU based SPAF network only offers excitatory and neutral neurons, this is explained by the fact that the ReLU activation only offers gradients on the positive spectrum of the inputs. Inhibitory neurons are defined as neurons, that have predominantly lower values (indicated by the black in the images) and excitatory neurons are defined by the predominantly large values (indicated by the white in the images).

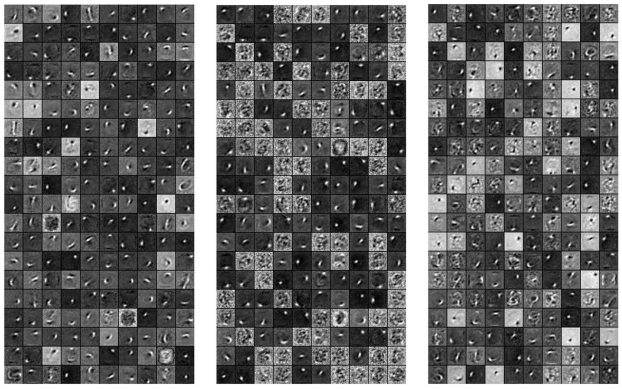
The second layer is far more interesting, as almost all the neurons produced are predominantly inhibitory neurons. The patterns still mimic the features learned in layer-1, as they are mostly basic detectors for a cluster of neighboring pixels. The ReLU based SPAF model fails at producing any discernible features for half of its neurons, and the rest of the neurons that do learn features are like those learned in the regular sigmoidal network; predominantly simple. The first block in question in Figure 8, is the representations of the new SPAF model. As can be seen almost all of the neurons with the exception of 2 (6th from top right, and 2nd from bottom left) produce complex features that mimic line and curve detection.

For the Fnt dataset, the Features learned of the first and second layers are shown in Figure 9 and Figure 10, respectively.



(a) Activation (1) (b) ReLU (c) Logistic

Figure 7: Features of the **first layer** of (a) SPAF-network with new activation function, (b) SPAF-network with ReLU, (c) Regularly DAE pre-trained network with logistic function, trained on the MNIST dataset

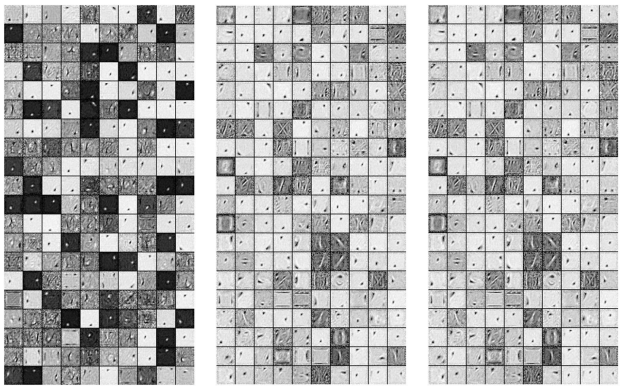


(a) Activation (1) (b) ReLU (c) Logistic

Figure 8: Features of the **second layer** of the three networks trained on the MNIST dataset

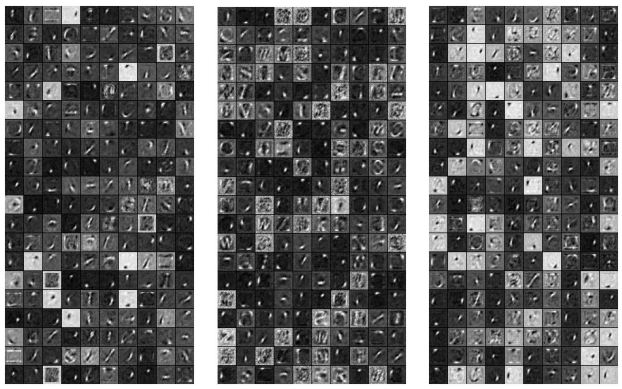
The ReLU based SPAF network again produces predominantly excitatory neurons with a minority being neutral. The SPAF network again produced more excitatory and inhibitory neurons than the regular sigmoidal model. The results of the second layer are quite interesting, as some of the neurons can be seen to have copied individual images. This is not necessarily great for classification, as these particular neurons have over-fit to one particular sample. The other two models continued to produce good features in its second layer.

The Hnd based model learned very few interesting features in all of the models. The fewest can be seen in the regular sigmoidal model. The trend of ReLU producing excitatory neurons in layer-1 continued here as well. Overall in the second layer, while the SPAF model and regular sigmoidal-network produced interesting features, there was however not much complexity when compared to previous datasets in the type of features learned. This can be attributed to the limited



(a) Activation (1) (b) ReLU (c) Logistic

Figure 9: Features of the **first layer** of the three networks trained on the FNT dataset



(a) Activation (1) (b) ReLU (c) Logistic

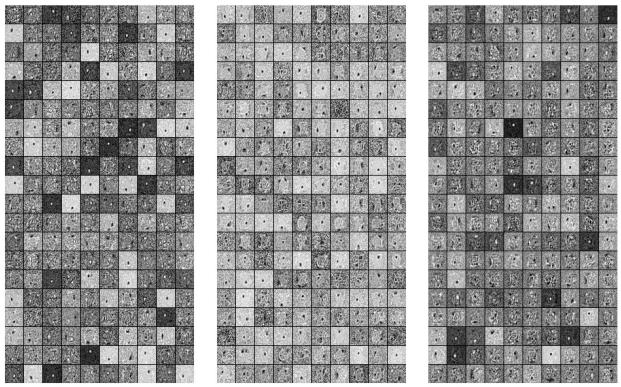
Figure 10: Features of the **second layer** of the three networks trained on the FNT dataset

number of samples used for each individual class, in fact by using 50% of the total 3410 samples available in the dataset, each individual class had roughly 27 samples in the training dataset. The respective features from layer-1 and layer-2 are shown in Figure 11 and Figure 12.

Lastly, what is important to take away from these results is that each network learned unique features, especially in layer-2 of the networks.

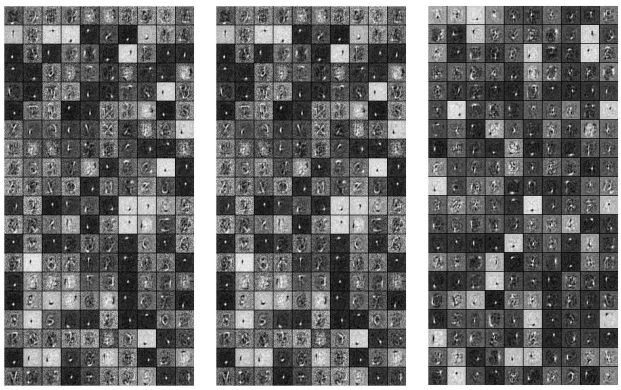
6 Conclusions

In this paper, we explored the pre-training stage in supervised networks whose weights are initialized using unsupervised autoencoders. We introduced a new activation function, a log based sparsity penalty on the hidden layer representation in the cost function, and the idea of swapping of activation functions from the unsupervised to supervised training stages. The experiment results have shown that the new model offers fuller features and better average error rates.



(a) Activation (1) (b) ReLU (c) Logistic

Figure 11: Features of the **first layer** of the three networks trained on the Hnd dataset



(a) Activation (1) (b) ReLU (c) Logistic

Figure 12: Features of the **second layer** of the three networks trained on the Hnd dataset

A limitation of the work is on the selection of constant parameters such as the learning rate and the parameters of the new activation function. A comparison over all the models with multiple reasonable learning rates can offer more insight over the differences in the learning abilities of the various models. For the sparsity function, it is also interesting to test others such as the l^2 function and possible combination of two functions [13] for an optimal network structure. Another consideration is the max number of allowed epochs for training in both the unsupervised and supervised stages. Varying the number of epochs and studying the effects of the length of training time over the features learned as well as the final performance will shed more light into the effects of swapping activation functions as well as varying penalty terms.

As potential future work, other activation functions could be considered during activation swapping from pre-training to fine-tuning stages. For instance, this work tested a new activation function that closely resembled a *tanh* but with a *y*-axis translation and a higher gradient. In this form as was seen, both excitatory and inhibitory features are produced in the hidden layer. Other activation functions may produce different proportions of excitatory, inhibitory and neutral neurons. The ReLU activation function for instance, only provides gradients for positive output, hence pushing the weights towards positive weights; this gives us a potential model that only ever produces excitatory neurons. Other activation function, that may focus more on producing negative values will likewise produce predominantly inhibitory neurons. It would be interesting to compare and contrast the performance of both these models. That can further be extended by creating a hybrid model that randomly selects 50% of the neurons from each model to create a hybrid layer with features created in models that produce excitatory neurons and inhibitory neurons.

Acknowledgement

The authors thank the anonymous referees for their detailed comments and suggestions that helped to improve the quality of the paper. The project was supported by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio. "Theano: New Features and Speed Improvements". *CoRR: Computing Research Repository*, abs/1211.5590, 2012.

[2] B. Burhani, W. Feng, and G. Hu. "Denoising Autoencoder in Neural Networks with Modified Elloitt Activation Function and Sparsity-Favoring Cost Function". *Proceedings of the 2015 3rd International Conference on Applied Computing and Information Technology/2nd International Conference on Computational Science and Intelligence (ACIT-CSI)*, pp. 343–348, 2015.

[3] A. Clark and Contributors. The Friendly Python Imaging Library, https://pillow.readthedocs.io/en/stable, 2010-2020 revision.

[4] M. A. Côté and H. Larochelle. "An Infinite Restricted Boltzmann Machine. *Neural Computation*, 28(7):1265–1288, 2016.

[5] T. E. de Campos, B. Rakesh Babu, and M. Varma. "Character Recognition in Natural Images". In *Proceedings of the 4th International Conference on Computer Vision Theory and Applications*, pp. 273–280, 2009.

[6] E. Hartman, J. D. Keeler, and J. M. & Kowalski. "Layered Neural Networks with Gaussian Hidden Units as Universal Approximation". *Neural Computation*, 2:210–215, 1989.

[7] D. Heaven. "Deep Trouble for Deep Learning". *Nature*, 574:163–166, 2019.

[8] G. Hinton, S. Osindero, and Y. W. Teh. "A Fast Learning Algorithm for Deep Belief Nets". *Neural Computation*, 18(7):1527–1554, 2006.

[9] G. E. Hinton. "Deep Belief Networks". *Scholarpedia*, 5(5):5947, 2009, doi:10.4249/scholarpedia.5947.

[10] G. E. Hinton. "Learning Multiple Layers of Representation". 11, pp. 428–434, 2007.

[11] G. E. Hinton and R. S. Zemel. "Autoencoders, Minimum Description Length, and Helmholtz Free Energy". *Advances in Neural Information Processing Systems*, pp. 3–10, 1994.

[12] H. Hu. "vReLU Activation Functions for Artificial Neural Networks". In *14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery, ICNC-FSKD*, pp. 856–860, 2018.

[13] N. Hurley and S. Rickard. "Comparing Measures of Sparsity". *IEEE Transactions on information Theory*, 55(10):4723–4741, 2009.

[14] M. T. Islam, B. M. N. Karim Siddique, Rahman S., and Jabid T. "Image Recognition with Deep Learning". In *2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*. IEEE, pp. 106–110, 2018.

[15] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin. "Exploring Strategies for Training Deep Neural Networks". *Journal Machine Learning Research*, 10:1–40, 2009.

[16] Y. Lecun, C. Cortes, and C. J. C. Burges. The MNIST Database of Handwritten Digits, http://yann.lecun.com/exdb/mnist, 2009.

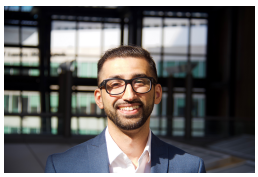
[17] Theano Development Team. "Theano: A python Framework for Fast Computation of Mathematical Expressions". *Computing Research Repository (CoRR)*, abs/1605.02688, 2016.

[18] P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol. "Extracting and Composing Robust Features with Denoising Autoencoders". In *Proceedings of the 25th International Conference on Machine Learning*. ACM, pp. 1096–1103.

[19] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. A. Manzagol. "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion". *The Journal of Machine Learning Research*, 11:3371–3408, 2010.

[20] M. D. Zeiler and R. Fergus. "Visualizing and Understanding Convolutional Networks". *LNCS*, 8689:818–833, 2014.

[21] X. Zhu, X. Luo, J. Zhao, D. Hou, Z. Han, and Y. Wang. "Research on Deep Feature Learning and Condition Recognition Method for Bearing Vibration". *Applied Acoustics*, 168, pp. 1–13, 2020.



Hasham Burhani is a Lead AI Scientist for RBC Capital Markets where he works on Deep Reinforcement Learning Models for trading execution. His research interests lie in deep reinforcement learning and more specifically multi-agent reinforcement learning techniques.

Wenying Feng is a Full Professor at the Department of Computer Science and the Department of Mathematics, Trent University, Canada. She is also an Adjunct Professor at the School of Computing, Queen's University, Canada. Dr. Feng's research areas inlcude nonlinear differential equations, nonlinear analysis, machine learning algorithms, mathematical and computational modelling. She has presented as a keynote speaker, served as program chairs and organized special sessions for international conferences.

Gongzhu Hu received B.S. in Numeric Analysis from Tsinghua University, M.S. in Computer Science from the University of Wisconsin-Madison, and Ph.D. in Computer Science from Michigan State University. He joined the Computer Science Department at Central Michigan University in 1987 and is a professor of the department. He was the department chair from 1994 to 2007. His research interests include databases, data mining, distributed systems, and formal methods. He has published over 150 papers in refereed journals and conference proceedings. Dr. Hu served as the conference chair or program chair of several international conferences as well as a member of the editorial board of several journals. He is a member of ACIS, ACM, ISCA, and a senior member of IEEE.

Image Processing for High-Throughput Phenotyping of Seeds using 3D Graphics

Venkat Margapuri*, Chaney Courtney*, and Mitchell Neilsen*
Kansas State University, Manhattan, KS, USA

Abstract

High-throughput phenotyping of seeds is the assessment of seed morphometry to aid in the prediction of yield, tolerance, resistance, and development of seeds in various environmental conditions. The paper focuses on the application of 3D graphics to image processing as a means to conduct seed phenotyping better. The paper proposes two algorithms - similar in the outcome, but different in implementation. The algorithms perform image processing on a variety of seeds such as wheat, soy, sorghum, rough rice, white rice, and canola to arrive at their morphometric estimations. In the area of static image processing, addressed are at least three common yet significant problems of seed clusters on images, skewed images, and poor image quality. As a means to address the problems, we propose the use of low-cost physical components. The algorithms provide the estimated count, area, perimeter, length, and width of seeds within an image.

Key Words: High-throughput phenotyping, 3D graphics, Watershed algorithm, image processing, seed morphometry.

1 Introduction

At the current growth rate of 1.3%, the population of the world is estimated to be at 9.3 billion by 2050. However, the world cereal yield and agricultural production have decreased since 1961 [7]. A suitable means to tackle this challenge is the application of phenotyping techniques to seeds. Seed Phenotyping is the comprehensive assessment of complex seed traits such as growth, development, tolerance, resistance, ecology, yield, and the measurement of parameters that form more complex traits [6]. High-throughput phenotyping increases the accuracy of measurements while reducing costs. Automation, remote sensing, data integration, and experimental design help reduce the costs of measurements. Several studies [3, 8, 10, 15] emphasize the importance of morphometry in predicting the behavior of performance in different environmental settings.

One of the foundational ideas for the work is the application of 3D graphics to high-throughput phenotyping. Common problems faced in the area of image processing include clusters work proposes a technique that uses low-cost physical on images, skewness of images, and poor image quality. The components, which improves the accuracy of morphometric estimations. The proposed components are as follows:

1. 3D printed stand to hold the image capture device. For this experiment, the image capture device is a phone; hence, the 3D printed stand has a groove fit to hold a phone.
2. 3D printed mesh with the idea that seeds lay within the hexagons of the mesh.
3. A lightbox to ensure that the images lay on a common background at the time of image capture leading to good quality images.

Furthermore, the work provides an algorithm that does not use the components described above. Instead, the alternate approach leverages the Watershed algorithm for image segmentation.

In the remainder of the paper, Section 2 specifies related work, Section 3 discusses the proposed algorithms for static image processing, and Section 4 concludes the paper.

2 Related Work

One of the building blocks for the algorithms is the Watershed algorithm, as discussed by F. Meyer and S. Beucher in the work, 'The Morphological Approach to Segmentation: The Watershed Transformation' [2]. The work discusses in detail the intricacies of the algorithm, including the tools, transformations, uses, and the application of Watershed to images.

Some of the Android applications which were similar and acted as useful validation/ comparison metrics were Smart Grain [13], Grain Scan [14], Leaf IT [11], and Seed Counter [4]. While Leaf IT was built for the morphometric estimation of leaves, all of the other applications above were developed for morphometric estimations of seeds.

3 Image Processing Algorithms

The following sub-sections present the developed image processing algorithms. The algorithms are developed using OpenCV-Python and applied to six different types of seeds: wheat, soy, rough rice, white rice, sorghum, and canola.

* Department of Computer Science. Email: marven@ksu.edu, chaneylc@ksu.edu, and neilsen@ksu.edu.

3.1 Mesh Algorithm

The Mesh algorithm refers to the static image processing algorithm involving the 3D stand, 3D mesh, and lightbox, as shown in Figure 1. Each of the components has a specific use case. The algorithm solves multiple problems in the following ways:

1. **3D Stand:** The 3D stand solves the problem of skew that most image processing algorithms encounter. The stand ensures that the image capture device such as a phone is always orthogonal to the surface. This angle ensures the image is not skewed. The stand used for the experiment is printed using a white filament with a height of 110 mm (11 cm).
2. **3D Mesh:** The 3D mesh addresses the problem of the object touching on images. The hexagonal boundaries act as barriers ensuring that the seeds do not touch each other at all times. The mesh, printed using a white filament contains hexagons of side 5 mm.
3. **Lightbox:** The lightbox ensures that the images which are captured have a bright background. A bright background makes it easier to identify the objects on the image and eliminate any noise such as tiny dirt particles that may be present on the image.

The algorithm can be generalized as a two-step process, described as follows:

1. Detect the mesh on the image and estimate the morphometry. The hexagons within the mesh are saved as ground truth values.
2. Detect the seeds on the image and use the estimated morphometry of the mesh to infer on the seeds.

3.1.1.1 Mesh Detection and Estimation using Pixel Color. A key point worth reiterating here is that the filament used to build the mesh and the light emitted by the lightbox are white. From Figure 2, the mesh is merely a shade darker to the light emitted by the lightbox.

1 For reproducibility, the mesh should have pixel values between 170 and 255 for each channel.
2 Perform a median blur and apply canny edge detection to detect the edges on the image.
3 Detect the contours of the edges identified by canny edge detection.
4 (Observation/ Assumption) Find all contours which occupy an area greater than an individual hexagon of the mesh in pixels.
5 Compute the median area occupied by the mesh in pixels and consider it as the area occupied by each hexagon in the mesh. Likewise, compute the median perimeter of the mesh.

A relationship between the number of pixels to the area in metric units is now established. The area of a hexagon in mm^2 is given by $(3 * 1.732 * a * a)/2$ where a is the side of the hexagon in mm, and the perimeter is given by $6 * a$ where a is the side of the hexagon in mm. Figure 3 shows the different steps involved in the process of mesh detection.

3.1.1.2 Mesh Identification and Estimation using HSV Thresholding

1. Convert the source image from BGR to HSV color scheme.
2. Create a mask which identifies H, S and V values of a desired range.
3. Perform a bitwise-AND operation on the mask and the source image.
4. Apply a Median blur followed by Canny edge detection.

Once the canny edges of the image are identified, the process to estimate the mesh is the same as that described in steps 4 and 5 of Section 3.1.1.1. Figure 4 shows the detection of the mesh from the input image.

3.1.2 Seed Detection and Estimation

1. Apply binary thresholding on the source image to filter out all of the white pixels and retain only the black pixels, i.e., pixels that represent seeds on the image.
2. Median blur the image and perform canny edge detection on the image.
3. Detect seeds on the image and compute the area occupied by each of the seed contours.
4. (**Seed Count**) The number of contours deemed seeds is the estimated number of seeds on the image.
5. (**Area and Perimeter Estimation**) The area of seeds in metric units is estimated by taking the median area occupied by the mesh's hexagons in pixels and metric units as ground truth values, and cross-multiplying, i.e., $seed\ area\ in\ mm^2 = (seed\ area\ in\ pixels * hexagon\ area\ in\ mm)/(median\ hexagon\ area\ in\ pixels)$. Likewise, a simple cross-multiplication is applied to obtain the perimeter of the seed in mm.

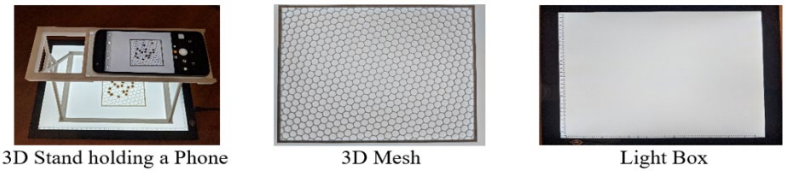


Figure 1: Proposed components

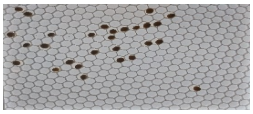


Figure 2: Soy seeds in a mesh

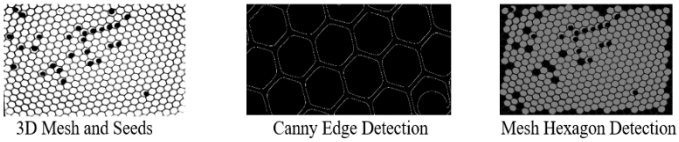


Figure 3: Mesh detection using pixel color

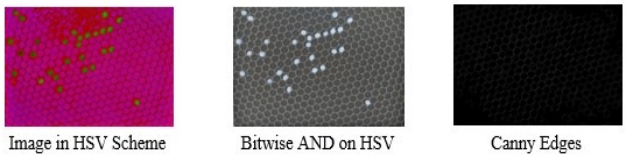


Figure 4: Mesh detection using HSV thresholding

6. (**Length and Width Estimation**) The lengths and widths of each contour are estimated using minimum area rotated rectangles. Rectangles are fit to the contour samples. The longest axis returned is considered the length, while the shorter axis is the width. Besides, connecting the left-most to the right-most and top-most to the bottom-most points, and computing the length of the connecting lines is a means to estimate the length and width of the seeds.

Figure 5 shows the seed counts, estimated seed areas in mm^2, and length-width estimations in mm for a soy seed sample.

As mentioned, the experiment is performed on six different types of seeds. The algorithm performs well consistently on five of the six types of seeds in question failing on white rice. The reason is that the light emitted by the lightbox is the same color as the seed of white rice. As a result, the algorithm fails to distinguish between the mesh and seeds. While the proposed algorithm still holds, a different experimental setup involving a contrasting background is essential for white rice. An implementation of the algorithm in Python-OpenCV is available at MeshAlgorithm.

3.1.3 Morphometry Estimations on Different Mesh Types

The algorithm is tested on three different geometries of meshes i.e., hexagon, triangle and rhombus with different infill rates and boundary thickness. Table 1 shows characteristics of the meshes used as part of the experiment.

As the morphometry of the seeds is estimated on different mesh types, the observation is that the estimations are similar, although not exactly the same on all of the mesh types. The difference in the estimations is better explained in section 3.4. In order to demonstrate the performance of the algorithm, a sample of seven soy seeds is considered and estimated using different mesh types. The seeds laid on different meshes is shown in Figure 6 and the readings are as shown in Table 2.

3.2 No-Mesh Algorithm

The No-Mesh algorithm is a static image processing algorithm that does not involve the 3D printed mesh, unlike the algorithm stated in section 3.1. Instead, the algorithm leverages the popular Watershed algorithm to segment seed clusters on images. The other components, 3D stand, and the lightbox are optional but recommended to achieve the proper performance of the algorithm. The algorithm is explained as a three-step process, described as follows:

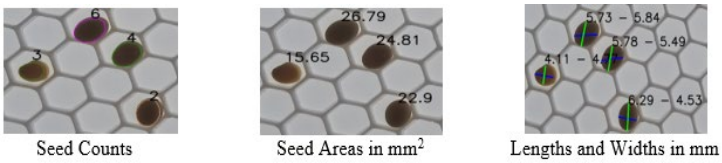


Figure 5: Estimated seed morphometry (image zoomed in)

Table 1: Geometry and characteristics of the meshes

Mesh Geometry	Side (mm) – Area(mm^2)	Infill Rates	Thickness
Hexagon	4 – 41.57, 5 - 64.95,6.5 – 109.77	18%, 15%, 12%, 10%,8%	5 mm, 1.5 mm, 0.5 mm
Rhombus	7.7 – 60.5, 6.3 – 40.5, 10.6 – 112.5	15%, 12%, 10%	5 mm, 0.5 mm
Triangle	13 – 84.5, 10 – 50	10%, 12%	5 mm, 0.5 mm

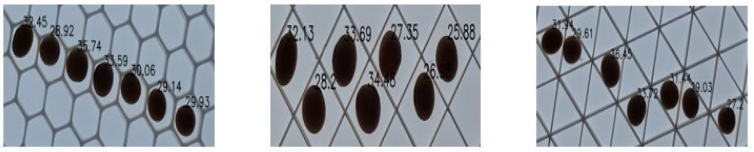


Figure 6: Seven seed soy sample area estimations

Table 2: Seed area estimations for seven seed sample of soy

Mesh	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7
Hexagonal	32.45	28.92	35.74	33.59	30.06	29.14	29.93
Rhombus	32.13	28.2	33.69	34.45	27.35	26.1	25.88
Triangular	31.91	29.61	36.45	33.72	31.44	29.03	27.2

1. Detect segmented seeds on the image i.e., seeds not in contact with other seeds on the image.
2. Apply the Watershed algorithm on the image and segment the seeds to perform morphometric estimations.
3. Estimate the morphometry of seeds using a ground truth object.

3.2.1 Detection of Segmented Seeds

The detection of segmented seeds makes use of the interquartile range, convexity defects, and the convex hull. The **Interquartile Range (IQR)** is a measure of the middle 50% of the values in a dataset. It is a proven mathematical technique that aids in the detection of outliers. In Figure 7, the IQR is $Q3 - Q1$. Q1 represents a quarter of the way through the list, and Q3 represents three-quarters of the way through the list. The IQR rule is that all values that are not between $Q1 - 1.5\ x\ IQR$ and $Q3 + 1.5\ x\ IQR$ are outliers.

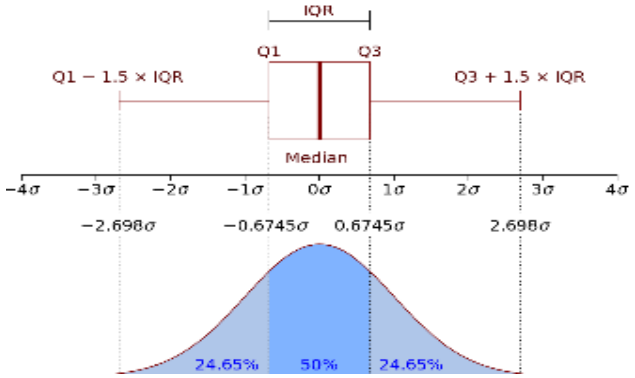


Figure 7: IQR calculation

Convex hull and convexity defects: Firstly, a convex object is one where none of the interior angles is greater than 180 degrees. The convex hull is a tight-fitting boundary around the object. As shown in Figure 8, the image on the left is convex. It has a convex hull that wraps entirely around the boundary, whereas the image on the right is not convex. The convex hull does not wrap around correctly. The imperfections in the convex hull are known as convexity defects.

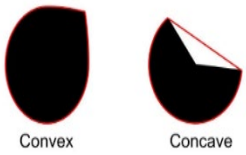


Figure 8: Convexity hull and convex defects

1. Convert the image to grayscale and gaussian blur the image using a 3 x 3 kernel.
2. To threshold the seeds from the background, apply inverted binary Otsu thresholding on the image.
3. Detect the contours of the seeds along with the area occupied by each of the contours. Also, find the convex hull of each of the seed contours.
4. Detect a segmented seed by evaluating the convexity of a contour, i.e., $if\ length(contour)/\ length(convexhull)\ <=$ 3, the contour is identified as a potential segmented seed.
5. Find the area occupied by each of the contours of the potential segmented seeds.
6. Remove the outliers from the potential segmented seeds by applying the IQR rule for outliers, stated above.
7. **(Case for Touching Seeds)** An edge case is that a few false positives might satisfy the step-four check. To eliminate them, filter any contours where the center is relatively white, i.e., with an intensity of about 170 or higher.
8. After segmentation, compute the morphometries, including the farthest point from the convex hull to the seed. Results are utilized further in Section 3.2.2.3. The steps in the process of segmented seed detection are as shown in Figure 9.

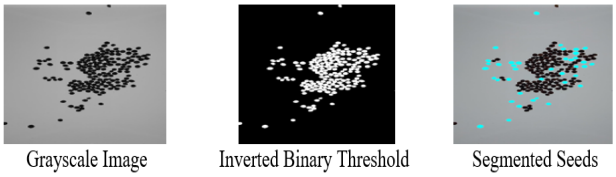


Figure 9: Detection of segmented seeds

3.2.2 Application of Watershed Algorithm

The Watershed algorithm's application relies on the concepts of morphological operations, erosion, dilation, closing, and opening, shown in Figure 10.

The principle behind morphological operations is the idea that 'on a grayscale image, black (0) and white (255) pixels can be identified with good precision'.

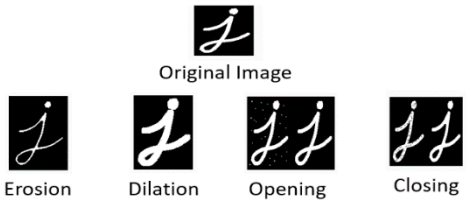


Figure 10: Morphological operations

Erosion: Erosion is a means to pare the image. A kernel of any size is convolved across the image. If one of the pixels along the kernel is black, all pixels on the image with respect to the kernel, are set to black (0).

Dilation: Dilation is a means to enhance the size of the image. A kernel is convolved across the image. If one of the pixels on the kernel is white, all pixels on the kernel are set to white (255).

Opening: Opening is a means to eliminate noise from an image. It is an erosion followed by a dilation.

Closing: Closing is a means to fill out the gaps in the image. It is a dilation followed by an erosion.

3.2.2.1 Preprocessing and Application of Watershed Algorithm

1. Convert the image to grayscale and apply binary thresholding to the image.
2. Perform a morphological opening to remove noise and a morphological closing to remove any holes in the image.
3. **(Background Identification)** Sure background area on the image is identified by performing a dilation.
4. **(Foreground Identification)** Compute the distance transform on the image and apply a threshold on the distance transform. The threshold values depend on the image.
5. **(Unknown/ Border)** The regions which are neither foreground nor background are the unknown/ border areas. They are computed by subtracting the foreground area from the background area.
6. **(Label Assignment)** Assign labels to the identified foreground, background, and unknown areas. If using OpenCV, the cv.ConnectedComponents() function can be used for this purpose. It assigns 0 to the background and random numbers starting from 1 to the objects in the foreground. However, the Watershed algorithm assumes all regions with a value of 0 as unknown areas. To avoid this scenario, increment all values by one, so the background has a value of 1 and assign 0 to all pixels belonging to the unknown area.
7. Apply a gaussian blur to the image to reduce the inner contour noise and avoid over-segmentation.
8. Apply the Watershed algorithm to the blurred image and capture the output markers.

The output of the watershed algorithm at each step of the process is as shown in Figure 11.

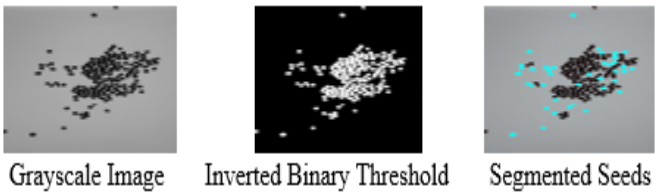


Figure 11: Preprocessing and application of watershed

3.2.2.2 Image Segmentation Post-Watershed Application

1. Draw the boundaries of the watershed segments on the markers produced by applying the Watershed algorithm (Remember: The Watershed algorithm indicates boundaries by -1).
2. Change the boundary markings from -1 to background, i.e., 0.
3. Invert the background and the foreground, i.e., set all 0's to 255's and the others to 0's, to identify boundaries in white since markers is a grayscale image.
4. Find contours on markers that also return the hierarchy of the contours.

Note: Contour hierarchy is an OpenCV concept. The hierarchy is a data structure that allows contours to access four different relative values, next contour, previous contour, first child, and parent contour.

5. Use the hierarchy to find all contours which are not considered noise. Identify child contours larger than a quarter of the average segmented seed contour area and mark them as 'parent'.

3.2.2.3 Counting Seeds. Seeds in images are often overlapping or clustered; therefore, counting the number of seeds on the image is a tricky task. The algorithm, however, provides two types of counting techniques, namely, area-based count and contour-based count.

Area-Based Count: The area-based counting technique uses

the average segmented seed areas to estimate the number of seeds in a given contour containing a cluster of seeds. For example, if the average pixel area of segmented seed contours is 10000 and the contour with a cluster of seeds has an area of 50000, the area based counting technique estimates the number of seeds in the cluster to be five.

Contour-based Count: The contour-based counting technique uses the results computed in section 3.2.1, i.e., the mean (SS_Mean), median (SS_Median), and standard deviation (SS_SD) of the segmented seeds. The count also makes use of the contour area (PC_CA), and the fixed-point depths (PC_FPD) of the contours identified as 'parent' contours in section 3.2.2.2. The estimation of the seed count is defined as the following:

$$If\ Mean\ (PC\_FPD) > SS\_Median + 3 * SS\_SD:$$

$$count = rounded\ value\ of\ (PC\_CA/SS\_Mean)$$

$$Else: count = 1$$

The algorithm is used to count the number of seeds on images of different seeds. The results are shown below in Figure 12.

As observed from the results, the algorithm performs well on the images of sorghum, soy, and wheat, where the range of estimated count is within the actual number of seeds on the image. However, on the seeds of canola and rough rice, the algorithm overcounts and undercounts, respectively. The results for white rice are not presented because the algorithm fails to detect the seeds precisely. Note that a change in the arrangement, be it the position or the orientation of seeds on the image, leads to the detection of different contours and might influence the seed count since it is entirely dependent upon the detected contours.

3.3 Seed Morphometry Estimations

Unlike the Mesh algorithm, the No-Mesh algorithm has no ground truth value. Hence, the idea is to capture the image with an object of known morphometry and base the estimations on the ground truth. For this experiment, the penny, a US coin, is used.

For the experiment, a total of four pennies are placed in each of the four corners of the lightbox when the image is captured. The pennies on the image are identified based on the size of the contour, i.e., the four most massive contours found on the image are assumed to be those of coins and circularity computed as $4 * pi * (area/perimeter^2)$ where area and perimeter are pertinent to the contour in question.

A sample of seven soy seeds whose areas (mm^2) are estimated using pennies is as shown in Figure 13.

3.4 Analysis of Outcome Impacting Parameters

Analysis on the impacts of parameters such as height of image capture, size of image and seed orientation is performed to recommend the best parameters for image capture. The observations are as follows.

3.4.1 Impact/ Correlation of Height and Image Size

In order to study the impact of height and image size, 3 random samples of 7 seeds are taken. 3 different heights of 11cm, 17cm and 21cm are the choice for image capture. The size of the images is varied between dimensions of *3456 x 4608* px and *1920 x 1080* px. For this experiment, the heights of image capture are varied while image size is kept constant. The results observed for all 3 samples of seeds are similar. The results of one of the samples are shown in Tables 3 and 4 for reference.

From the readings in Table 1, it can be inferred that the impact of height is minimal when the size of the image is *3456 x 4608* px.

From the readings of the values in Table 2, it is inferred that the size of the image and the height of image capture have an inverse relationship i.e., the height of the image has minimal impact when the size of the image is large. Figure 14, which contains seeds and pennies on the image together help to reason the behavior. Green contours are supposed to be drawn only around seeds but when the height of image capture is 21 cm, the pennies also appear small enough that they are also considered seeds

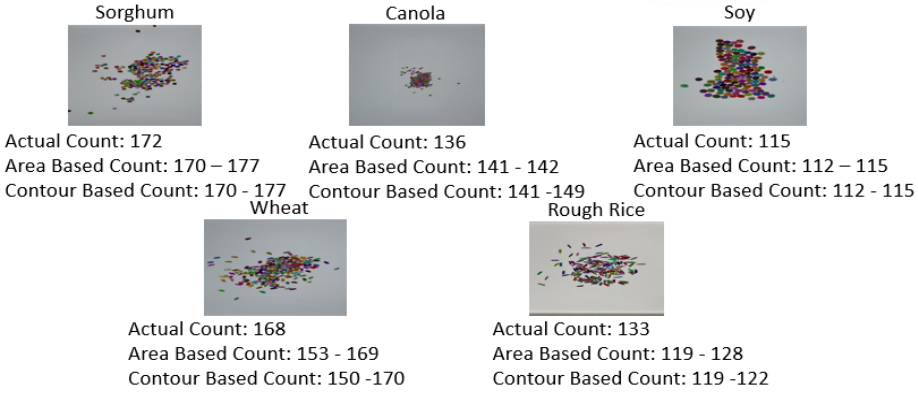


Figure 12: Seed counts using no-mesh algorithm

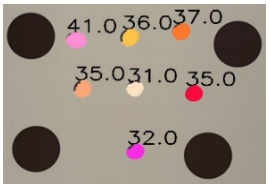


Figure 13: Estimated seed areas in mm^2 using pennies

3.4.1 Impact of Seed Orientation

The impact of seed orientation is investigated by using a sample of seven seeds laid on a mesh. The hypothesis behind investigating orientation is that the algorithm gives out different results depending upon the orientation of the seeds since it affects the detected contours which is ultimately the basis for the estimations. Turns out that the hypothesis is true and is explained further from observing Figure 15 which shows a sample of seven seeds each laid in a different orientation.

As observed from the results in Table 5, the lengths and the widths of the seeds differ depending upon the orientation of the seeds. This could also impact the area estimations if they were based on the length and width of the seeds.

3.4.2 Impact of Thickness of the Mesh

The thickness of the mesh has a direct correlation to the portion of the seed that is overlaid by the mesh. The more the overlay, the higher the deviation of the estimated morphometry to the actual. Figure 16 shows an example where a sample of seven canola seeds are laid on a thicker hexagonal mesh (3 mm) and a thinner rhombus mesh (0.5 mm).

Table 3: Seed areas with varying image heights and image size of *3456 x 4608* px

Image Height	**Seed 1**	**Seed 2**	**Seed 3**	**Seed 4**	**Seed 5**	**Seed 6**	**Seed 7**
11cm	32.99	32.49	29.48	34.53	28.07	28.43	27.71
17cm	33.04	32.84	29.71	32.18	27.98	28.16	27.97
21cm	33.13	33.03	29.85	31.98	27.79	28.07	28.27

Table 4: Seed areas with varying image heights and image size of *1920 x 1080* px

Image Height	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7
11cm	44.0	42.0	37.0	42.0	35.0	37.0	37.0
17cm	41.0	36.0	37.0	35.0	31.0	35.0	32.0
21cm	0	0	0	0	0	0	0

Figure 14: Impact of height of image capture on estimation

Figure 15: Impact of height of seed orientation on estimation

Table 5: Lengths and widths of seeds in different orientations (length – width in mm)

Orientation	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Seed 7
#1	7.44 - 5.96	6.85-6.17	6.38-6.15	6.62-6.16	6.06-5.9	6.14-5.87	6.36-5.68
#2	7.19-6.15	6.76-6.1	6.39-6.18	6.56-6.23	6.04-5.85	6.11-5.85	6.56-5.53

Figure 16: Impact of mesh thickness on estimation

The readings in Table 6 show that the estimates for seed areas are significantly different when using meshes of different thickness. The thicker mesh overlays more and hides more part of the seed compared to the thinner mesh which does not overlay as much. As a result, the contours detected are not the same even with all other parameters being the same, causing the results between the two images to be deviant. For instance, consider the seed circled in red in Figure 16. The detected contours for the seed when placed on the thicker and lighter meshes are as shown in Figure 17. As observed, the thicker mesh results in a part of the seed being hidden yielding a partial contour. The impact of mesh thickness is felt more on seeds that are relatively smaller in size such as canola.

3.5 Comparison to Other Applications

The areas estimated by the proposed algorithms are compared against the estimates of other applications. Seed Counter, LeafIT, SmartGrain, and a manual estimation performed using a vernier caliper are compared. A sample of seven soy seeds is used for the comparison. The reason for the soy seed sample is that some of the applications in consideration are built to estimate specific types of seeds and do not necessarily pick-up seeds smaller or larger than a specific size. Soy is one of the seeds which is estimated by all of the applications in question and the results of estimation are shown in Figure 18.

From the results:

1. Manual estimations using a vernier caliper are consistently lower than those provided by any of the applications and proposed algorithms.
2. The results provided by the Mesh algorithm are most similar to those of the results provided by the android app, Seed Counter with the most considerable difference in estimates being two mm^2 for seeds one, four, and seven.

Table 6: Seed areas using meshes of different thickness (area in mm^2)

Thickness	Seed 1	Seed 2		Seed 3	Seed 4	Seed 5	Seed 6	Seed 7
3mm	1.57	1.82		1.33	2.29	1.5	1.36	1.15
0.5mm	2.75	2.55		2.32	3.29	2.16	2.55	2.43

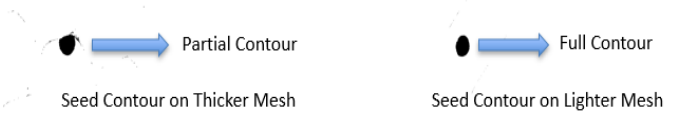


Figure 17: Detected Seed Contour on Thicker and Lighter Meshes

3. The results provided by the No-Mesh Algorithm are most similar to that of LeafIT's, with the most considerable difference in estimations being three mm2. Not surprisingly, both algorithms use ground truth reference objects. The difference, though, is that LeafIT estimates morphometry of leaves, one at a time. In contrast, the No-Mesh algorithm estimates morphometry of seeds, multiple at a time.

4 Future Work and Conclusion

The Mesh algorithm is an excellent addition to the current image processing techniques using 3D graphics. With encouraging results in the initial phase, the focus in the next phase is to solidify the algorithm by testing the feasibility and accuracy of the algorithm on various kinds of meshes and further study the impact of factors such as the height of image capture, image size, camera distortion and seed orientation on the algorithm. Current plans to implement the No-Mesh algorithm as part of an Android app are in progress, and an initial version of the implementation is currently being tested.

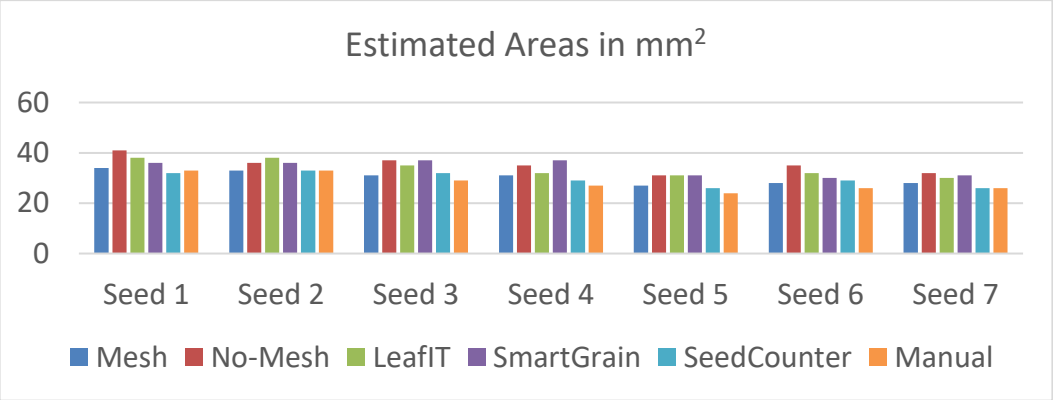


Figure 18: Estimated seed Areas in mm^2 using pennies

References

[1] S. Amaravadi, "*Mobile Applications for High-Throughput Seed Characterization*, Master's thesis, Kansas State University", 2018.

[2] S. Beucher and F. Meyer, "The Morphological Approach to Segmentation: the Watershed Transformation", *Mathematical Morphology in Image Processing*, 34:433-481, 1993.

[3] Z. Kehel, M. Sanchez-Garcia, A. El Baouchi, H. Aberkane, A. Tsivelikas, C, Chen, and A. Amri, "Predictive Characterization for Seed Morphometric Traits for Genebank Accessions Using Genomic Selection", *Frontiers in Ecology and Evolution*, 8:32, 2020.

[4] E. Komyshev, M. Genaev, and D. Afonnikov, "Evaluation of the SeedCounter, a Mobile Application for Grain Phenotyping". *Frontiers in Plant Science*, *7*:1990, 2017.

[5] A. S. Kornilov and I. V. Safonov, "An Overview of Watershed Algorithm Implementations in Open-Source Libraries". *Journal of Imaging*, 4(10), 123, 2018.

[6] L. Li, Q. Zhang, and D. Huang, "A Review of Imaging Techniques for Plant Phenotyp-ing", *Sensors*, 14(11):20078-20111, 2014.

[7] O. E. Maisonet-Guzman, "Food Security and Population Growth in the 21st Century", Retr. From http://www.e-ir.info/2011/07/18/food-security-and-popula tion-growth-in-the-21st-century, 2011.

[8] R. R. Mir, M. Reynolds, F. Pinto, M. A. Khan, M. and M. A. Bhat, "High-Throughput Phenotyping for Crop Improvement in the Genomics Era", *Plant Science*, 282:60-72, 2019.

[9] M. L. Neilsen, C. Courtney, S. Amaravadi, Z. Xiong, J. Poland, and T. Rife, "A Dynamic, Real-Time Algorithm for Seed Counting", *Proc. Of the 26th International Conference on Software Engineering and Data Engineering*, October, 2017.

[10] M. L. Neilsen, S. D. Gangadhara, and T. Rife, "Extending Watershed Segmentation Algorithms for High Throughput Phenotyping", *Proceedings of the 29th International Conf. on Computer Applications in Industry and Engineering,* Denver, CO, September 2016.

[11] J. Schrader, G. Pillar, and H. Kreft, "Leaf-IT: An Android Application for Measuring Leaf Area", *Ecology and Evolution*, 7(22), 9731-9738, 2017.

[12] S. Suzuki, "Topological Structural Analysis of Digitized Binary Images by Border Following", *Computer Vision, Graphics, and Image Processing*, 30(1):32-46, 1985.

[13] T. Tanabata, T. Shibaya, K. Hori, K. Ebana, and M. Yano, "SmartGrain: High-Throughput Phenotyping Software for Measuring Seed Shape Through Image Analysis". *Plant Physiology*, 160(4):1871-1880, 2012.

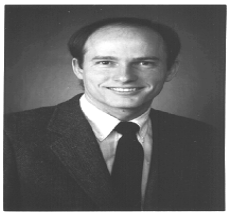
[14] A. P. Whan, A. B. Smith, C. R. Cavanagh, J. P. F. Ral, L. M. Shaw, C. A. Howitt, and L. Bischof, "GrainScan: A Low Cost, Fast Method for Grain Size and Colour Measurements", *Plant Methods*, 10(1):23, 2014.

[15] V. Wiley and T Lucas, "Computer Vision and Image Processing: A Paper Review", *International Journal of Artificial Intelligence Research*, *2*(1):29-36, 2018.

[16] S. Yuheng and Y. Hao, "Image Segmentation Algorithms Overview". *arXiv preprint arXiv:1707.02051*, 2017.

Venkat Margapuri is a researcher working on his PhD. in Computer Science at Kansas State University. His interests are in the areas of machine learning, scientific computing, robotics and data science.

Chaney Courtney is a Computational Scientist with a demonstrated history of working in the higher education. He is skilled in distributed algorithms, android development, deep learning, programming languages, computer vision, and leadership. He is a research professional with a PhD in Computer Science from Kansas State University.



Mitchell L. Neilsen is a Professor in the Department of Computer Science at Kansas State University. His research interests include scientific computing, cyber-physical systems, computer vision and distributed algorithms.

Generalizing Chinese Remainder Theorem Based Fault Tolerant Non-DHT Hierarchical Structured Peer-to-Peer Network

Koushik Maddali*, Swathi Kaluvakuri*, Indranil Roy
Southern Illinois University Carbondale, Carbondale, IL, USA

Ziping Liu†
Southeast Missouri State University, Cape Girardeau, MO, USA

Bidyut Gupta*
Southern Illinois University Carbondale, Carbondale, IL, USA

Narayan Debnath‡
Eastern International University, VIETNAM

Abstract

In this work, we have considered generalizing a fault tolerant, non-DHT-based, low diameter hierarchical structured P2P network where neighborhood relationships among peers are designed applying modular arithmetic, called the Chinese Remainder Theorem (CRT). This is an interest based two level architecture. At each level of the overlay hierarchy, all the participating peers are directly connected to one another making it a complete graph. Hence, each level network has a diameter of 1 overlay hop. Such low diameters play an important role in designing very efficient data lookup algorithms where both intra and inter-group data lookup alogrithms achieve a time-complexity of $O(1)$. In addition, use of the same mathematical approach to denote a resource type and the corresponding group-head is also a reason for the search process to be simple and efficient.

Key Words: Structured P2P network, chinese remainder theorem, network diameter, data lookup, generalization, interest based, fault tolerance.

1 Introduction

Peer-to-Peer (P2P) overlay networks are prevalent in distributed systems. Unstructured and structured constitute two classes of P2P networks. In unstructured systems [2, 5] peers are organized into arbitrary topology. Flooding is commonly used for data look up. Churn is the problem where peers frequently join and leave the system. Churn is handled effectively in unstructured systems. However, it compromises with the efficiency of data query and the coveted flexibility. On the other hand, properly designed structured architectures can offer efficient, flexible, and robust service [12, 16, 19, 21, 25]. Such overlay networks use distributed hash tables (DHTs) to achieve efficient data insertion, lookup etc. However, maintaining DHTs is a complex task and it needs huge effort to manage the problem of churn [4, 24]. Therefore, the major challenge for architecture design is easing churn management while still providing an efficient data query service. There are notable approaches that have considered hybrid systems [22] with the aim of incorporating the advantages of both structured and unstructured architectures. However, these works have their own pros and cons [1, 7, 15].

1.1 Our Contribution

In this work, we have considered P2P systems based on common interest [9, 10, 14, 22]. We have conceived a non-DHT based hierarchical P2P architecture in which each level of the network hierarchy is structured, and diameter of each network is 1 overlay hop. Preliminary findings have been reported in [6]. To the best of our knowledge, it is the first time that a successful attempt has been made to design structured hierarchical P2P networks with its entire constituent subnetworks possessing the diameter of 1 overlay hop only. Note that low diameters play significant role in designing very efficient data lookup algorithms. The proposed architecture uses a mathematical model based on the Chinese Remainder Theorem (CRT) to define the neighborhood relations among the peers to obtain small diameters.

In this work, we present generalization of interest based hierarchical peer-to-peer architecture and various fault tolerant cases involved. In addition, Intra-group as well as inter-group data look up algorithms with $O(1)$ time complexity considering the generalization of the architecture is also defined.

The paper is organized as follows. In Section 2, we state in

* School of Computing. Emails: koushik@siu.edu, swathi.kaluvakuri @siu.edu, indranil.roy@siu.edu, bidyut@cs.siu.edu.
† Dept. of Computer Science. Email: zliu@semo.edu.
‡ School of Computing and Information Technology. Email: Narayan.debnath@eiu.edu.

detail the proposed architecture and the mathematical foundation used in the design phase. In Section 3, we present two cases of fault tolerance. In Section 4, we present the generalization of the architecture. Finally, In Section 5, analytical comparisons with some notable works based on data complexity as well as the common interest nature has been presented.

2 Preliminaries

In this section, we present a structured architecture for interest-based peer-to-peer system [10, 11, 13, 18] and the required mathematical basis supporting the architecture. As mentioned earlier that some related preliminary findings have been reported in [6]. We use the following notations along with their interpretations while we define the architecture. We define a resource as a tuple $<R_i, V>$, where R_i denotes the type of a resource and V is the value of the resource. A resource can have many values. For example, let R_i denote the resource type 'songs' and V' denote a particular singer. Thus $<R_i, V'>$ represents songs (some or all) sung by a particular singer V'. In the proposed model for interest-based P2P systems, we assume that no two peers with the same resource type R_i can have the same tuple; that is, two peers with the same resource type R_i must have tuples $<R_i, V>$ and $<R_i, V">$ such that $V' \neq V"$. We define the following. Let S be the set of all peers in a peer-to-peer system. Then $S = \{P^{Ri}\}$, $0 \leq i \leq r-1$. Here P^{Ri} denotes the subset consisting of all peers with the same resource type R_i and no two peers in P^{Ri} have the same value for R_i and the number of distinct resource types present in the system is r. Also for each subset P^{Ri}, P_i is the first peer among the peers in P^{Ri} to join the system. However, this constraint can easily be relaxed.

We now propose the following architecture suitable for interest-based peer-to-peer system. We assume that no peer can have more than one resource type.

2.1 Two Level Hierarchy

We propose a two-level overlay architecture and at each level, structured networks of peers exist. It is explained in detail below. At level 1, we have a network of peers such that peers are directly connected (logically) to each other. In graph theoretic term, the network at level 1 is a complete graph. Hence, the network diameter is 1 overlay hop. The periphery of this network appears as a ring network and we name it as transit ring network. This network consists of the peers P_i ($0 \leq i \leq r-1$). Therefore, number of peers on the ring is r and we have assumed that this number represents the number of distinct resource types of the P2P system. Each of these n peers is termed *group-head*. The periphery of this network as well as the direct links connecting any two peers in this network can be used for efficient data lookup. In this architecture, each group-head has a global resource table (GRT) that has every group-head's logical as well as IP addresses.

At level-2, there are n numbers of completely connected networks of peers. Each such network, say N_i is formed by the peers of the subset P^{Ri}, ($0 \leq i \leq n-1$), such that all peers (ϵP^{Ri}) are directly connected (logically) to each other, resulting in the network diameter of *1 overlay hop*. Each such N_i also called group (in short as G_i) is connected to the transit ring network via the peer P_i, the group-head of G_i. The architecture is shown in Figure 1.

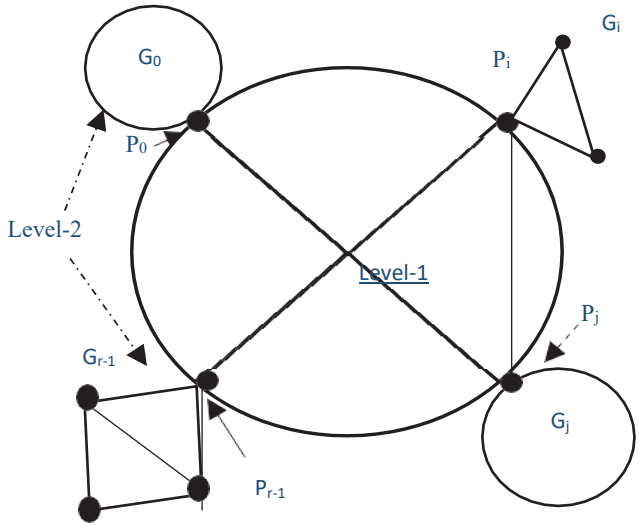


Figure 1: A two-level structured P2P architecture with r distinct resource types

In any structured P2P system, the mathematical model used to build the architecture defines neighborhood relations among peers. The mathematical model is intimately related to the efficiency of different data lookup schemes used in a given structured P2P system.

We now state a brief sketch of the mathematical model used in this approach to realize the architecture [14]. The authors first determine a simultaneous solution (a positive integer) of a given system of linear congruencies and then determine some more solutions as needed to form the architecture, which are congruent to the simultaneous solution. For this, the authors have used the Chinese Remainder Theorem (CRT). Each such solution will become the logical address of a group head uniquely [14, 15]. At the same time, it requires to determine separately the solutions of each linear congruence as needed and these solutions will represent the logical addresses of the peers present in a group [11]. The following interesting structural facts are revealed.

Observation 1. Any insertion of a group head P_l always takes place between the current last group head P_{l-1} and the first group head P_0 along the transit ring network.

Lemma 1. Diameter of the Level-1 network is 1 overlay hop.
Lemma 2. Diameter of a Level-2 group is 1 overlay hop.

Theorem 1. Diameter of the hierarchical two-level structured architecture is 3 overlay hops.

Remark 1. There are infinite number of solutions which are congruent to the one mutually congruent solution of any Linear

Diophantine Equation (LDE) considered in CRT, hence, size of a cluster at Level-2 can be made very large (theoretically unlimited), yet the diameter remains 1.

Observation 2. Each group head has two different logical addresses; one from Level-1 assignment and one from Level 2 assignment.

Observation 3. Different group heads may get identical Level 2 assigned addresses. It will not affect any intra-cluster lookup query in a cluster, as this address is local to this group only.

3 Fault Tolerance

Fault tolerance of a network is defined as the ability of the network to continue its operation without any interruption in the event of an unexpected error or problem. The main objective of our fault tolerant architecture is to prevent the problem of single point failure, to ensure the network is up and running without any interruptions. Cases of peers joining the system have already been discussed in [8]. In this section, we talk about the how our P2P architecture handles this situation in cases of peer crashing. The situation of a group head failure is considered for fault tolerance:

3.1 Group Head Failure

Whenever there is a situation of group head Pk failure, a new group head P* is selected from the peers of the level-2 common resource group. This comes with a small overhead. All the pointer values present in the group head Pk are to be copied onto a peer P* $\in G_k$ periodically. As all the peers in the group are directly connected, a single hop is all it takes to reach one another.

Due to this, any non-group head peer can play the role of P*. In this work, we follow the sequence of peer joining as the basis for selection of P*. Additionally, an update from Pk to P* is triggered whenever the group head Pk detects a change in the transit network. To guard against any loss of information due to the group head failure, snapshot of its request queue is sent to P* each time the content of the queue is updated. The procedure of setting up a new group head is described in Figure 2 and it leads to the following important observation. Effect of a group head failure is restricted only to its common interest group.

When a group member is faulty, value retention is not possible as having multiple copies of the value is an overhead and not practical in the proposed architecture. The case of group member failure is discussed below.

3.2 Group Member Failure

If any peer on a level two architecture fails, as all the participating peers of each group are directly connected to each other, when a peer P $\in G_k$ fails, each neighboring peer in the group G_k will detect the failure through periodic hello packet exchanges and then delete the entry for P from its list of neighbors. The very nature of the peer relationship achieved using our work [6], the connectivity among the rest of the peers remains intact.

4 Generalization of the Architecture

here are multiple scenarios of data insertions through which generalization can be discussed. Let us consider a situation where in a group G_k, the group-head P_k or a peer P $\in G_k$ wants data insertion in the system of another existing resource type R_a; note that R_a exists in the group G_a and P_k / P already possesses R_k. The two cases of data insertions to be discussed in this work are:

1. Peers with multiple existing resource types
2. Existing peers declaring new resource types

4.1 Peers with Multiple Existing Resource Types:

Following is how generalization is achieved in this case. Peer P_k / P will become a member of group G_a as well. That is, the IP address of P_k / P will be known to the members of both group G_k and group G_a. Logically it means that in the overlay network, P_k / P will be directly connected to all members of both G_k and G_a. Following algorithm implements the above-mentioned insertion.

Step 1:	If the group head P_k of the group G_k fails, its neighbors on the level-1 transit network P_{k+1} and P_{k-1} as well as those in the level-2 common interest network learn about the failure through periodic hello packet exchanges.
Step 2:	To make sure the GRTs remain unchanged, P^* is now designated as the group head and its logical address is changed from $(x_k + 1.m_k)$ to x_k. As P* is already aware of the logical address of P_{k+1} and P_{k-1} due to the periodic information exchanges between itself and the group head, P^* takes the role of P_k effectively.
Step 3:	This group head change is now broadcasted in the group G_k.
Step 4:	Successor of P^*, $P^{**} = (x_k + 2.m_k)$ is chosen to be the new P^* and periodic information exchanges get initiated between the new group head and P^{**}.

Figure 2: Algorithm 1. Fault tolerance in the case of group head failure

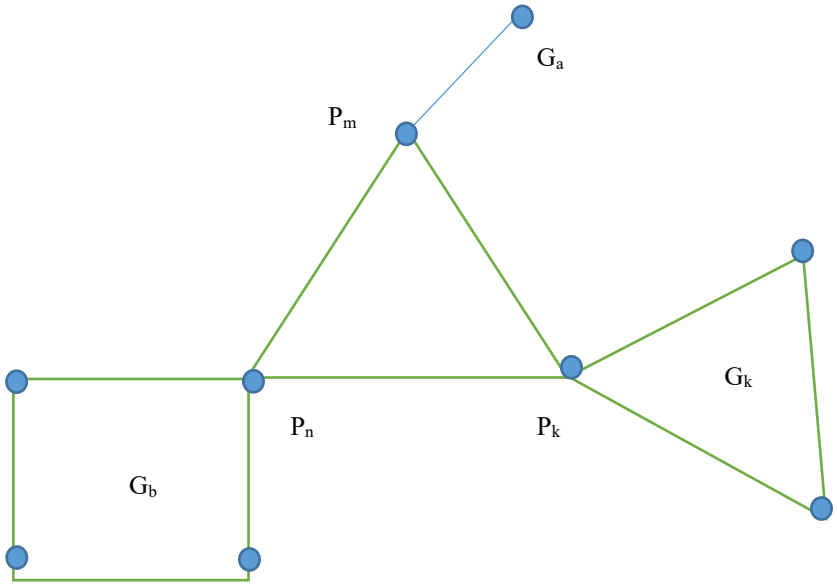


Figure 3: A structured P2P architecture with 3 distinct resource types

1. A request for data insertion of resource type R_a is forwarded from P_k / P to the group head P_a. //no of hops is 1 if origin is P_k, 2 if origin is P
2. P_a assigns the next available address which is not yet assigned in the group G_a to P_k / P.

   // Next available address is derived using $(x_a + j\ m_a)$
3. P_a broadcasts the address of P_k / P in G_a and each group member updates their respective neighbor list.
4. P_a then unicasts a copy of its neighbor list to P_k / P

Figure 4: Algorithm 2. Data insertion in the case of peers with multiple existing resource types

4.2 Existing Peers Declaring New Resource Types:

To start with, it is assumed that the P2P system has S number of distinct resource types, viz., R_0, R_1, R_2 … R_{s-1}. Without any loss of generality, let peer P_k / P in group G_k wants a data insertion for a new resource type R_s. Then following the way, the transit ring is constructed, peer P_k / P will become the group-head of the newly created groups possessing resource type R_s. As the recent group-head P_s, location of P_k / P on the ring is now between P_{s-1} and P_0. Therefore, if it is P_k, peer P_k will appear (logically) twice as group-heads on the ring for group G_k and group G_s. If it is peer P, it will appear once as the group-head of group G_s and once as a member of group G_k. For G_s, P_k / P bears an address from (X + i*m). Now, P_k / P will ask all the group-heads to update their global resource tables by including R_s along with the IP address of P_k / P.

4.3 Data Lookup Considering Generalization of the Architecture

It is considered that a peer P_k is also the group-head Ps of group G_s. The proposed approach works as well if P_k possesses any number of distinct resource types. It is assumed that the system has r distinct resource types.

4.4 Intra Group Lookup

Without any loss of generality, let us consider data lookup in group G_i by a peer P_a possessing $< R_i, V_a >$ and requesting the resource $<R_i, V_b>$. The algorithm for intra-group data lookup appears in the below above. The time complexity achieved here is O (1).

4.5 Inter Group Lookup

In the proposed architecture, any communication between a node $P_i \in G_i$ and $P_j \in G_j$ takes place only via the respective group-heads P_i and P_j.

Without any loss of generality let a peer $P_i \in$ Gi request for a resource $< R_j, V_j>$ where R_j denotes a resource type and V_j denotes an instance of R_j. As is evident from the architecture that peer P_i knows that R_j does not belong to G_i. We denote the

1. Node P_a broadcasts in Gi for $<R_i, V_b>$
   // One hop communication as it's a complete graph
2. if P_b with $<R_i,V_b>$ then
3. Pb unicasts $<R_i, V_b>$ to node P_a
4. else
5. Search for $<R_i,V_b>$ fails
6. end

Figure 5: Algorithm 3. Intra group lookup considering generalization of the architecture

1 Node P_i (ϵ G_i) unicasts request for $< R_j, V_j>$ to group-head P_i
2 P_i determines IP(P_j) from GRT
3 P_i unicasts the query to P_j // P_i is directly linked with P_j
4 **if** P_j possesses $< R_j, V_j>$ **then**
5 P_j unicasts $< R_j, V_j>$ to P_i
6 P_i unicasts $< R_j, V_j>$ to P_a
7 **else**
8 P_j broadcasts the request for $<R_j, V_j>$ in group G_j
// one-hop communication since G_j is a complete graph
9 if P_b (ϵ G_j) with $<R_j, V_j>$ **then**
10 P_b unicasts $< R_j, V_j>$ to P_j
11 P_j unicasts $< R_j, V_j>$ to P_i
12 P_i unicasts $<R_j, V_j>$ to P_a
13 else
14 **end**

Figure 6. Algorithm 4. Inter group lookup considering generalization of the architecture

IP address of a node P_m as IP(P_m). The algorithm appears in the above figure. The time complexity of the algorithm is $O(1)$. It needs a maximum of 3 overlay hops irrespective of in which groups the requesters, and the corresponding responders reside

5 Comparison

5.1 Data Lookup Complexity

In [19] search along the chord is not followed, because it is very inefficient in a large peer to peer system since the mean number of hops required per search is N / 2, where N is the total number of peers in the system. In our work, the maximum number of hops required over both the levels per search is 3.

It is also apparent from the fact that in Chord [19] and in other structured P-2-P systems [16, 25] the complexity involved in data lookup is a function of the no. of peers N in the system. The point to mention is that use of the same code to denote a resource type R_i and the corresponding group-head P_i has made the search process simple and efficient. So, our work achieves a time complexity of $O(1)$. In Table 1, we have presented data lookup complexity of our approach as well as those of some important existing DHT based systems. Observe that from the viewpoint of data lookup complexity, our proposed architecture offers better performance.[23]

5.2 Discussion on Some Noted Interest Based Works

Here we have considered four noteworthy interest based P2P systems [3, 9, 17, 20] and briefly state their main features from the viewpoint of their proposed architecture. Then we state the same of our work and justify why our design is superior to these works. All these four works have incorporated the idea of peer heterogeneity in their design. In doing so the work in [3] has

Table 1: Data lookup complexity comparison

	CAN[15]	**Chord [5]**	**Pastry [4]**	**Our Work**
Architecture	DHT-based	DHT-based	DHT-based	CRT-based
Lookup Protocol	{Key, value} pairs to map a point P in the coordinate space using uniform hash function.	Matching key and NodeID.	Matching key and prefix in NodeID.	Inter-Group: Routing through Group-heads; Intra-group: Inside a Graph.
Parameters	*N*-number of peers in network; *d*-number of dimensions.	*N*-number of peers in network.	N-number of peers in network; *b*-number of bits *($B = 2^b$)* used for the base of the chosen identifier.	*r* - Number of distinct resource types; N-number of peers in network. *$r << N$*
Lookup Performance	$O(d\ N^{1/d})$	$O(\log N)$	$O(\log_B N)$	Intra-group: $O(1)$ Inter-group: $O(1)$

used the existing idea of super peer. Besides, gossiping has been used for group formation with peers of common interest. The work in [20] uses the idea of popular peer which is quite like the idea of super peer. The base architecture is an unstructured network. In [9] authors have considered super peer. It is a hybrid architecture that uses both chord and unstructured network.

In [17] gossiping has been used for group formation. Besides, at the time of joining a new peer searches from a list of known peers for a particular peer which has most links among all in the list and then gets connected to it. In our proposed work, we do not use gossiping for group formation; we do not consider either super peer or popular peer or even a joining peer does not look for the best peer to be connected to as in [17]. We now justify why it is so in our work. First the existing idea of gossiping is not at all an efficient way to form groups of peers of common interests. Second, when new peers join, there is always some probability that the new one will be better than an existing super peer or a popular peer. So, again a new one may have to be selected. It wastes time, particularly when several peers join at the same time or peers join frequently. The joining of a new peer [17] incurs unnecessary waste of time to search for the best peer to connect to. In addition, some of the above works have considered unstructured network. So, it may involve the typical searching problem inherent to unstructured networks.

We have used a very well-established number theory based mathematical tool, viz. Chinese Remainder Theorem (CRT). It assigns overlay addresses to peers in a way that any group with peers of common interest becomes a complete graph consisting of any number of peers and the group-heads form a complete graph too. Thus, we avoid using the inefficient gossiping method for group formation. Ours is a non-DHT based structured architecture that offers search latency of $O(1)$ both for intra and inter group communication. Overlay diameter of a group is just 1 and that of the whole architecture is just 3. To the best of our knowledge, there does not exist any structured P2P architecture with such lowest possible diameters. Besides, our process of churn handling is one of the simplest known ones. Since a group is a complete network, any new join or a leave does not change its diameter; thereby restructuring of a group is very simple. Besides, position of a new group is always between the first one and the existing last one. Realistically, we believe that our work is a challenge to any existing DHT based architecture from the viewpoints of search latency and churn handling.

Finally, during design phase we have not considered peer heterogeneity; it has made our design process very simple. So, there is no need to select any super or a popular peer. However, we have already reported our capacity constrained communication protocols [8] in our CRT based architecture; thereby, we have incorporated peer heterogeneity only during communication. That is one of the main features of our design.

6 Conclusion

In this paper, we have extended our non-DHT based structured P2P architecture to incorporate the generic idea that a peer can possess multiple resource types. We have applied some properties of modular arithmetic, specifically Chinese Remainder theorem (CRT), to design a scalable, hierarchical structured overlay P2P system, which provides highly efficient data lookup algorithms. One noteworthy point is that complexity involved in data lookup is a function of the number of distinct resource types n unlike in DHT-based systems. Another point to mention is that use of the same mathematical approach to denote a resource type R_i and the corresponding group-head P_i has made the search process simple and efficient. This work is a part of an ongoing research project with the goal of designing P2P federation consisting of small P2P systems so that bandwidth cannot be an issue.

References

[1] C. K. S. Banerjee and B. Bhattacharjee, "Scalable Application Layer Multicast", *Proc. ACM SIGCOMM'02,* pp. 205-217, Aug. 2002.

[2] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making Gnutella-Like P2P Systems Scalable, *Proc. ACM SIGCOMM*, Karlsruhe, Germany, pp.407-418, August 25-29 2003.

[3] Wen-Tsuen Chen, Chi-Hong Chao and Jeng-Long Chiang, "An Interested-Based Architecture for Peer-to-Peer Network Systems", 20th International Conference on Advanced Information Networking and Applications (AINA'06), Vienna, 1:707-712, 2006, doi: 10.1109/AINA.2006.93, 2006.

[4] E. Cohen, A. Fiat, and H. Kaplan, "Associative Search in Peer-to-Peer Networks: Harnessing Latent Semantics, 2:1261-1271, 2003.

[5] P. Ganesan, Q. Sun, and H. Garcia-Molina, "Yappers: A Peer-to-Peer Lookup Service over Arbitrary topology", *Proc. IEEE Infocom 2003*, San Francisco, USA, pp. 1250-1260, March 30-April 1 2003.

[6] Bidyut Gupta, Nick Rahimi, Henry Hexmoor, and Koushik Maddali. "Design of a New Hierarchical Structured Peer-to-Peer Network Based On Chinese Remainder Theorem", *Proc. The 33rd International Conference on Computers and Their Applications (CATA)*. Pp. 69-74, 2018.

[7] Bidyut Gupta, Nick Rahimi, Henry Hexmoor, Shahram Rahimi, Koushik Maddali, and Gongzhu Hu, "Design of Very Efficient Lookup Algorithms for a Low Diameter Hierarchical Structured Peer-to-Peer Network", *Proc. IEEE 16th Int. Conf. Industrial Informatics (IEEE INDIN)*, Porto, Portugal, pp. 861-868, July 2018.

[8] B. Gupta, N. Rahimi, H. Hexmoor, S. Rahimi, K. Maddali and G. Hu, "Design of Very Efficient Lookup Algorithms for a Low Diameter Hierarchical Structured Peer-to-Peer Network", IEEE 16th International Conference on Industrial Informatics (INDIN), Porto, pp. 861-868, 2018, doi: 10.1109/INDIN.2018.8471936, 2018.

[9] Mujtaba Khambatti, Kyung Ryu, and Partha Dasgupta. "Structuring Peer-to-Peer Networks Using Interest-Based Communities", Lecture Notes in Computer Science, 1st International Workshop, DBISP2P 2003, Berlin, September 2003.

[10] S. K. A. Khan and L. N. Tokarchuk, "Interest-Based Self Organization in Group-Structured P2P Networks", 6th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, pp. 1-5, 2009, doi: 10.1109/CCNC.2009.4784959, 2009.

[11] M. Kleis, E. K. Lua,, and X. Zhou, "Hierarchical Peer-to-Peer Networks using Lightweight SuperPeer Topologies", *Proc. IEEE Symp. Computers and Communications*, pp. 143-148, 2005.

[12] D. Korzun and A. Gurtov, "Hierarchical Architectures in Structured Peer-to-Peer Overlay Networks", Peer-to-Peer Networking and Applications, Springer, pp. 1-37, March 2013.

[13] Andrea Passarella, "A Survey on Content-Centric Technologies for the Current Internet: CDN and P2P Solutions", *Computer Communications*, 35:1-32, 2012.

[14] Z. Peng, Z. Duan, J.Jun Qi, Y. Cao, and E. Lv, "HP2P: A Hybrid Hierarchical P2P Network", *Proc. Intl. Conf. Digital Society*, pp. 18-28, 2007.

[15] N. Rahimi, K. Sinha, B. Gupta, and S. Rahimi, "LDEPTH: A Low Diameter Hierarchical P2P Network Architecture", *Proc. 2016 IEEE Int. Conf. on Industrial Informatics (INDIN 2016)*, Poitiers, France, pp. 832-837, July 2016.

[16] A. Rowstron and P. Druschel, "Pastry: Scalable, Distributed Object Location and Routing for Large Scale Peer-to-Peer Systems", *Proc. FIP/ACM Intl. Conf. Distributed Systems Platforms (Middleware)*, pp. 329-350, 2001.

[17] I. Roy, K. Maddali, S. Kaluvakuri, B. Rekabdar, Z. Liu, B. Gupta, and N. Debnath, "Efficient Any Source Overlay Multicast in CRT-Based P2P Networks - A Capacity-Constrained Approach", IEEE 17th International Conference on Industrial Informatics (INDIN), Helsinki, Finland, pp. 1351-1357, 2019, doi: 10.1109/INDIN41052.2019.8972151, 2019.

[18] K. Shuang, P Zhang, and S. Su, "Comb: A Resilient and Efficient Two-Hop Lookup Service for Distributed Communication System", *Security and Communication Networks*, 8(10):1890-1903, 2015.

[19] R. I. Stocia, R. Morris, D. Liben-Nowell, D. R. Karger, M. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications, *IEEE/ACM Tran. Networking*, 11(1):17-32, Feb. 2003.

[20] Z. Tu, W. Jiang, and J. Jia, "Hierarchical Hybrid DVE-P2P Networking Based on Interests Clustering", International Conference on Virtual Reality and Visualization (ICVRV), Zhengzhou, China, pp. 378-381, 2017, doi: 10.1109/ICVRV.2017.00087, 2017.

[21] M. Xu, S. Zhou, and J. Guan, "A New and Effective Hierarchical Overlay Structure for Peer-to-Peer Networks", *Computer Communications*, 34:862-874, 2011.
[22] M. Yang and Y. Yang, "An Efficient Hybrid Peer-to-Peer System for Distributed Data Sharing", *IEEE Trans. Computers*, 59(9):1158-1171, (Sep. 2010).
[23] R. Zhang and Y. C. Hu, "Borg: A Hybrid Protocol for Scalable Application-Level Multicast in Peer-to-Peer Networks", *Proc. Int'l. Workshop Network and Operating System Support for Digital Audio and Video (NOSSDAV'03)*, pp. 172-179, 2003.
[24] R. Zhang and Y.C. Hu, "Assisted Peer–to-Peer Search with Partial Indexing", *IEEE Trans. Parallel and Distributed Systems*, 18(8):1146-1158, 2007.
[25] B. Y. Zhao, L. Huang, S. C. Rhea, J. Stribling, A. Zoseph, and J. D. Kubiatowicz, "Tapestry: A Global-Scale Overlay for Rapid Service Deployment", *IEEE J-SAC*, 22(1):41-53, Jan. 2004.

Koushik Maddali (photo not available) is a Ph.D. candidate in Department of Computer Science at Southern Illinois University Carbondale. He received his MS from the same university and his BS from Jawaharlal Nehru Technological University, India. His research interests include Peer to Peer Networking, BlockChain and worked on a Virtual Terminal project of Cisco from 2017-2018.

Swathi Kaluvakuri (photo not available) is a Ph.D. candidate from Southern Illinois University Carbondale – School of Computing. She graduated from Jawaharlal Nehru Technological University with a Bachelor of Technology degree in Computer Science major. She holds a keen interest in the areas of Peer-to-Peer Networking and BlockChain and worked as a Software Engineer, Technical Product Support and IBM AS400 developer for NetCracker Pvt Ltd from 2012-2014.

Indranil Roy (photo not available) is currently a PhD student in Computer Science Department of Southern Illinois University, Carbondale. He has completed his B.E in Electronics & Communication from RCCIIT, Kolkata, in the year 2016. He received his M.S in Computer Science degree from Southern Illinois University, Carbondale. His main research interests include blockchain along with interest-based P2P architecture.

Ziping Liu (photo not available) is currently a Professor of Computer Science at Southeast Missouri State University. Her research interests include wireless ad-hoc network/sensor network communication and security, distributed computing and cloud Computing, machine learning, mobile and full-stack application development, and secured software design

Bidyut Gupta (photo not available) received his M. Tech. degree in electronics engineering and Ph.D. degree in Computer Science from Calcutta University, Calcutta, India. At present, he is a professor at the School of Computing (formerly Computer Science Department), Southern Illinois University, Carbondale, Illinois, USA. His current research interest includes design of architecture and communication protocols for structured peer-to-peer overlay networks, security in overlay networks, and block chain. He is a senior member of IEEE and ISCA.

Narayan Debnath (photo not available) earned a Doctor of Science (D.Sc.) degree in Computer Science and also a Doctor of Philosophy (Ph.D.) degree in Physics. Narayan C. Debnath is currently the Founding Dean of the School of Computing and Information Technology at Eastern International University, Vietnam. He is also serving as the Head of the Department of Software Engineering at Eastern International University, Vietnam. Dr. Debnath has been the Director of the International Society for Computers and their Applications (ISCA) since 2014. Formerly, Dr. Debnath served as a Full Professor of Computer Science at Winona State University, Minnesota, USA for 28 years (1989-2017). Dr. Debnath has been an active member of the ACM, IEEE Computer Society, Arab Computer Society, and a senior member of the ISCA.

Novel Design of Load-Balanced and Fault-Tolerant Multicasting Protocols for PIM-SM

Indranil Roy*, Swathi Kaluvakuri*, Koushik Maddali*
Southern Illinois University Carbondale, Carbondale, IL USA

Ziping Liu†
Southeast Missouri State University, Cape Girardeau, MO USA

Bidyut Gupta*
Southern Illinois University Carbondale, Carbondale, IL USA

Narayan Debnath‡
Eastern International University, VIETNAM

Abstract

Shared tree multicast uses a single core to handle entire multicast traffic load in a domain. In this paper, we present a new load-balanced multicast approach with multiple cores to reduce the traffic load. In addition, we have considered fault tolerant multicasting in presence of multiple core failures. To achieve this, we partition all multicast addresses (class D addresses) in p partitions and select p.q cores with q cores per partition. Different multicast groups using addresses belonging to the same partition will use the q cores designated for this partition for load balanced multicore multicasting. This partitioning of multicast addresses is a very recent idea and assigning cores with the respective partitions is a practical approach. This partitioning of addresses is a one-time job because these addresses have nothing to do with network topology. In addition, for selecting the p.q cores, there is no need to know the topology. We have considered a method that will consider the information present in the routers' routing tables to select the cores and this approach does not require the knowledge of the topology at all. Each time, routers' routing tables are updated, a new set of cores can be selected based on only the routers' routing information. Existing works in this direction have to have the knowledge of the topology without which these works cannot select any cores. Each time the topology changes due to some link/router failures, these works have to first learn about the changed topology before the selection of the cores. That makes these approaches impractical even though some of these works are theoretically elegant.

Key Words: Static partition, core selection, pseudo diameter, load sharing, fault-tolerance.

* School of Computing. Email: indranil.roy@siu.edu, swathi.kaluvakuri@siu.edu, koushik@siu.edu, bidyut@cs.siu.edu.
† Dept. of Computer Science. Email: zliu@semo.edu.
‡ School of Computing and Information Technology. Email: Narayan.debnath@eiu.edu.

1 Introduction

Multicast communication protocols [4] are classified into two categories, namely, source-based trees [1, 17, 22] and core based trees (CBT) [3]. A problem associated with source-based-tree routing is that a router has to keep the pair information (source, group) and it is a one tree per source. In reality the Internet is a complex heterogeneous environment, which potentially has to support many thousands of active groups, each of which may be sparsely distributed; this technique clearly does not scale. Shared tree based approaches like CBT [3, 15] and protocol independent multicasting – sparse mode (PIM-SM) [6] offer an improvement in scalability by a factor of the number of active sources.

In a core-based tree/shared tree [3] the tree branches are made up of other routers, so-called non-core routers, which form a shortest path between a member-host's directly attached router and the core. A core need not be topologically centered, since multicasts vary in nature and therefore, the form of a core-based tree also can vary. CBT is attractive compared to source based tree because of its key architectural features like scaling, tree creation, and unicast routing separation. The major concerns of shared tree approach are; core selection and core as a single point failure. Core selection [3, 11, 15] is the problem of appropriate placement of a core or cores in a network for the purpose of improving the performance of the tree(s) constructed around these core(s). Core selection in static networks depends on knowledge of entire network topology. It involves all routers in the network. There exist several important works [12, 18, 20-21] which take into account network topology while selecting a core.

1.1 Related Works

In case of single core-tree based multicast, the core has to handle all traffic load. It degrades the performance. To overcome the problem, shared tree multicast using multiple cores is the only solution. It distributes the total traffic load on the cores resulting in improved load balancing and thereby causing

improved multicast performance. There exist in the literature some important contributions in this area of multicore-based multicasting [7, 10, 19, 23]. The goal of these works is to achieve load balancing. In [10], Jia et al., have presented a Multiple Shared-Trees (MST) based approach to multicast routing. In their approach, the tree roots are formed into an anycast group so that the multicast packets can be anycast to the nearest node at one of the shared trees. However, load balancing is at the level of each source choosing the best core closest to it rather than attempting to utilize all the candidate cores simultaneously. This may lead to congestion in a core if multiple sources choose that core based on their shortest delay metric. In all these works, a core is assigned to a multicast group during a multicast session.

In [19], a unique tree consisting of multiple cores is maintained with one of the cores being the root. The objective of the work is to develop a loop free multi-core based tree by assigning level numbers to the cores and the nodes to join the tree to help maintain the tree structure. The cores need to coordinate with one another for their operations.

Zappala et.al., [23] have considered two different approaches for multicore load shared multicasting. The first one is senders-to-many scheme; it partitions the receivers of a group among the trees rooted at different cores so that each receiver is exactly on one core tree at a time. Therefore, one core tree spans some of the group members only. Even though it offers less routing state, yet it has the complex task to take care of newly arriving group members, i.e., partition them appropriately to the different core trees. In their second scheme, each core tree spans over the entire receiver group. To transmit data, different senders in a multicast group can use different trees with respect to the proximity of the source to the tree; it helps in balancing the traffic load and improve performance. The trees are maintained independently unlike the work in [19]. The core distribution method follows the hash based scheme of [5]. Note that a different kind of load sharing (not load balancing) approach exists for splitting the load over Equal Cost Multipath (ECMP). Multicast traffic from different sources or from different sources and groups are load split across equal-cost paths provided such equal cost paths exist and are recognized as well [16]. The idea of core-based multicasting has been extended to allow migration of cores [7]. When the performance at an alternate core is 'reasonably' better than that at the current core, migration takes place to the alternate core. It helps in controlling multicast latency and load sharing.

1.2 Motivation and Our Contribution

In this paper, we have considered shared tree multicast with multiple cores in PIM-SM domain. The motivation of the work is two-fold: to offer a practical approach to achieve improved multicast performance via load sharing and to achieve fault tolerant multicasting in presence of multiple core failures. Sometimes the term 'sharing' is preferred over 'balancing' because the objective of the work is to engage all cores whenever possible and more so because it is neither possible to know a priori the duration of different multicast sessions running at a given time, nor it is possible to know a priori the number of possible senders per multicast group.

There exist several works of core selection in the literature [10, 12, 18-19, 23]. No doubt, that it is a well-studied area. All these works must have the knowledge of the topology of the networks to determine the best core or a set of best possible cores. If there is a change, say some link or router failures, a topology will change from its earlier version. As a result, the designers have to start all over again to get to know the new topology before determining a best core or a set of good cores. While some of these works are theoretically elegant, yet the whole idea to know the topology each time there is a change due to link/router failures, appears quite impractical.

It has led us to consider an extension of the following recently reported practical approach [2] in which all possible multicast addresses (i.e., class D addresses) are partitioned a priori in, say, p partitions of equal size. Any first hop router can do it immediately after the network is booted and it will do it only once because it is independent of network topology. For PIM-SM, to build the routing tables of the routers, whatever unicast routing protocol is used, routing information needs to be exchanged periodically, or if some router experiences a change in its neighborhood relations, it may initiate the process. In [9], pseudo diameter of a router R has been defined as the largest cost present in its routing table; it means that using this cost router R can send any information to anywhere in the network. Thus, pseudo diameter actually offers a good idea about the physical location of the router in the network. Therefore, considering all routers whichever has the lowest pseudo diameter (the lowest cost) can be considered as the best core (approximately located at a central position in the network). There may be more than one such core. It has been shown earlier [14] that information about the network topology is not needed when core selection is done based on pseudo diameter idea. The work in [2] selects p number of cores starting from the router with the lowest pseudo diameter. Therefore, working principle of the multicore multicast protocol is that assign all multicast communications with addresses belonging to a given partition to one such core. Designers can select the value of p. Hence, there is no need to know the topology of a network to find the best core or a set of best possible cores. The next time when, the routing information will be exchanged, the routers can easily determine a new set of such cores without requiring to learn if there has been a change in the network topology. In short, cost-information in the routing tables is used to determine a best core or a set of best possible cores without having to know the topology of a given network. It makes the process of core selection quite practical. According to the authors [2] there does not exist any such multicast architecture for PIM SM that uses static partitioning of all possible multicast addresses for multicore multicast.

We have extended the above multicore multicast approach to include both load balanced multicast and its fault tolerant version for shared tree (PIM-SM) architecture. To achieve it, we partition each sub-partition p_i of p partitions further into identical sized q distinct address ranges, say range1(i), range2(i), …, range q(i). Next, we select a total of p.q cores using the pseudo diameter idea. The purpose for this is that for each sub-partition p_i, we will keep aside q distinct cores, say core1(i), core2(i), …, core q(i). The designers can select the values of p and q. The

working principle of our proposed load balanced approach is: for multicast to address range, say range1(i), assign core1(i), for address range range2(i), assign core2(i), and so on. Note that in the proposed protocol, we have used slightly different notations for the address ranges and the q cores. Later, we have extended the protocol to include fault tolerant load balance multicasting in presence of multiple core failures.

Organization of the paper is as follows. In Section 2, we state some important results related to pseudo diameter-based core selection process and discuss its superiority compared to some existing eminent works. In Section 3, we present the load balanced multicast protocol and in Section 4, we present the fault tolerant protocol.

2 Preliminaries

Two widely used unicast routing protocols are distance vector routing (DVR) and link state routing (LSR). In the former one, routers do not have the knowledge of network topology, whereas in the latter routers have this knowledge. The concept of pseudo diameter is independent of the underlying unicast routing protocol. We denote the unicast routing table by UCT_i for some router R_i; it can be either the DVR table or the LSR table of the router depending on the unicast protocol used. Pseudo diameter of a router R_i denoted as $P_d(R_i)$ is defined as follows [9, 13-14].

$$P_d(R_i) = \max \{c_{i,j}\}, \text{ where } c_{i,j} = \text{cost}(r_i, r_j), \ [1 \le j \le n, j \ne i]$$

and $c_{i,j} \in UCT_i$ and n = number of routers in the network

In words, pseudo diameter of router R_i denoted as $P_d(R_i)$, is the maximum value among the costs (as present in its routing table UCTi) to reach from R_i all other routers in a network. The implication of pseudo-diameter is that any other router is reachable from router R_i within the distance (i.e., no. of hops/delay etc.) equal to the pseudo diameter $P_d(R_i)$ of router R_i, It thus directly relates to the physical location of router R_i. Pseudo diameter is not the actual diameter of the network, because it depends on the location of router R_i in the network. So different routers in the network may have different values for their respective pseudo diameters. Therefore, pseudo diameter P_d is always less than or equal to the actual diameter of a network. Hence is the name pseudo diameter.

As an example [7], consider the network shown in Figure 1. Without any loss of generality, we have considered DVR protocol as the underlying unicast protocol used in the network. Note that the diameter of the network is 90. From router A's DVR table (Figure 2), it is seen that its pseudo diameter is 90, which is incidentally equal to the network diameter, whereas for router C it is 70 as is seen from C's DVR table in Figure 2. It means that if C is the source of communication, the maximum cost to reach any other router will be 70, which is less than the network diameter of 90. Observe that in this network router E has the minimum pseudo diameter, viz. 60.

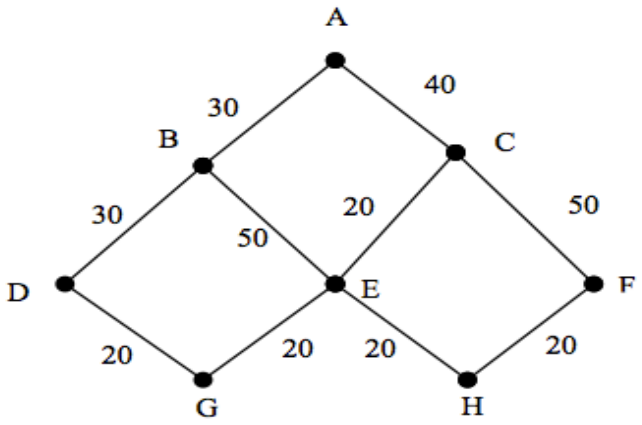


Figure 1: An 8-router network

2.1 Performance

The multicore selection scheme needs only one broadcast independent of the number of cores to be used for multicasting. Therefore, the message complexity is $O(n^2)$, where n is the total number of nodes in the network. However, in this approach a router does not need to have the complete topological information. Note that to incorporate the effect of any changes in the network, e.g., router failure, this core selection process will run whenever routing information is exchanged (usually periodically). Besides, the core selection scheme is independent of any multicast groups unlike most existing works because it is a static core selection approach. A detailed discussion of the performance of the presented core selection method has appeared in [8]. The core selection scheme has been compared with some important existing core selection algorithms, mainly Maximum Path Count (MPC) [12], Delay Variant Multicast Algorithm (DVMA) [18], Minimum Average Distance (MAD) method, and Opt Tree method [12]. Results of the comparisons for randomly generated networks of different sizes are shown in the following figures. The superiority of our approach from the viewpoint of message complexity to these other eminent approaches is evident from the results. Note that unlike in Opt Tree method, in our approach a router does not need to have the complete topological information. Experimental setup has used NS2 simulator, BRITE topology generator, and Waxman's probability model for interconnection of the nodes.

3 Static Partition-Based Load-Balanced Multicore Multicasting

In the proposed approach, we first select statically the candidate cores; followed by the proposed partitioning scheme of all possible multicast addresses. All first-hop routers perform this partitioning immediately after the network is booted. They do it independently. During a multicast session, a first-hop router connected to a multicast source uses the knowledge of the

A		
Dest.	Next	Delay
A	A	0
B	B	30
C	C	40
D	B	60
E	C	60
F	C	90
G	C	80
H	C	80

B		
Dest.	Next	Delay
A	A	30
B	B	0
C	A	70
D	D	30
E	E	50
F	E	90
G	D	50
H	E	70

C		
Dest.	Next	Delay
A	A	40
B	A	70
C	C	0
D	E	60
E	E	20
F	F	50
G	E	40
H	E	40

D		
Dest.	Next	Delay
A	B	60
B	B	30
C	G	60
D	D	0
E	G	40
F	G	80
G	G	20
H	G	60

E		
Dest.	Next	Delay
A	C	60
B	B	50
C	C	20
D	G	40
E	E	0
F	H	40
G	G	20
H	H	20

F		
Dest.	Next	Delay
A	C	90
B	H	90
C	C	50
D	H	80
E	H	40
F	F	0
G	H	60
H	H	20

G		
Dest.	Next	Delay
A	E	80
B	D	50
C	E	40
D	D	20
E	E	20
F	E	60
G	G	0
H	E	40

H		
Dest.	Next	Delay
A	E	80
B	E	70
C	E	40
D	E	60
E	E	20
F	F	20
G	E	40
H	H	0

Figure 2: DVR tables of the routers

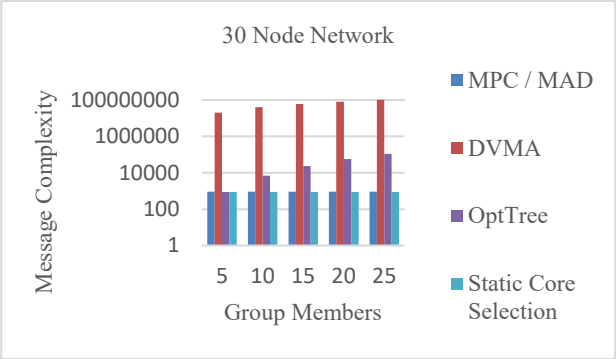


Figure 3: 30-router network

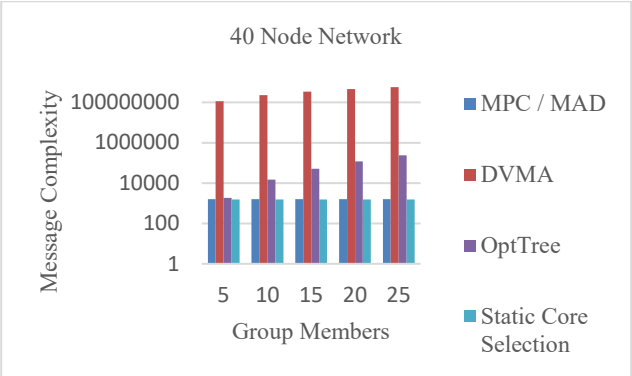


Figure 4: 40-router network

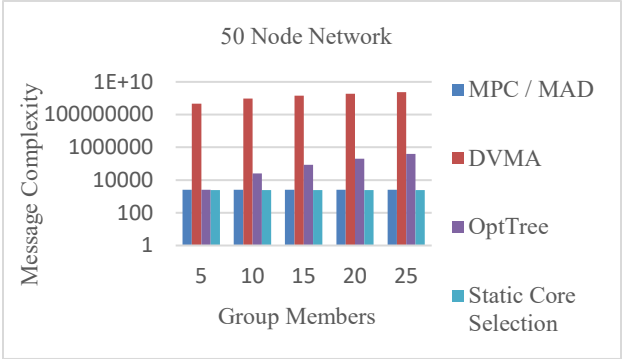


Figure 5: 50-router network

partitions and the cores to accomplish its responsibility following the principle of PIM-SM.

3.1 Selection of Candidate Cores – A Distributed Approach

We exclude any first hop router from a possible set of candidate cores. The reason for such exclusion is that a first hop router's main responsibility is to connect LAN users to the rest of the network domain. We denote a router R_x as R_x^f if it is a first hop router; otherwise by just R_x. We present a modification of the schemes proposed in [2, 7] to select candidate cores to be used for multicore load balanced multicasting. Cores are selected statically using the routing information of all routers (except all first hop routers) in the underlying domain. This core selection is independent of any multicast group.

Candidate cores selection process by each first hop router

1. Each non-first hop router R_i determines its $P_d(R_i)$ from its unicast routing table.
2. It broadcasts its $P_d(R_i)$ in the network using pseudo diameter-based broadcasting [14].
3. Every first hop router R_f receives all $P_d(R_i)$ from every non-first hop router R_i.
4. Router R_f creates a list of r routers out of all non-first hop routers, sorted in ascending order based on their P_d values.
   // these r routers are used for multicore load balanced multicast.
5. Each first hop router R_f partitions the list into p equal parts, i.e. $r = p.q$.
   // each partition contains q cores
6. R_f identifies the 1^{st} core in the i^{th} partition, p_i as the primary core, C^i of partition p_i for $1 \le i \le p$.
   // P_d of C^i is minimum among the q cores in partition p_i.
7. R_f creates a list L of the p primary cores corresponding to the p partitions.

Observation 1. Every first hop router creates identical sorted list of the r cores.
Observation 2. Message complexity of the core selection process is $O(n^2)$.
Observation 3. In the core selection process, a router does not need to know the topology.

3.2 Responsibility of Each First Hop Router R_f in the Network Immediately after the Core Selection Process is Over

Partitioning of Multicast Addresses:

1. Each first hop router R_f divides all group addresses into p distinct ranges (partitions) with starting address of the i^{th} partition as SA_i, $1 \le i \le p$.
2. Router R_f creates a vector (VSA) consisting of the starting addresses of the partitions.
   It is: $VSA = < SA_1, ..., SA_i, ..., SA_p >$ *sorted in ascending order of their magnitudes.*
3. Router R_f uses the list L to build a core map table (CMT) of size p such that its i^{th} entry is
   $CMT(i) = < SA_i$, *IP address of core* $C^i >$.

Observation 4. Every first hop router creates identical VSA.
Observation 5. Partitioning of multicast addresses is independent of any network topology.

Each primary core C^i maintains the following date structure:

1. Primary core C^i divides the address range SA_i to SA_{i+1} into q distinct ranges.
2. It builds the following array of q tuples $<(sa_1, C_1^i), (sa_2, C_2^i), ..., (sa_q, C_q^i)>$

where a core C_j^i $(1 \le j \le q)$ belongs to partition p_i and is responsible for multicasting packets to addresses in the range $[sa_j$ *and* $< sa_{j+1}]$.

3.3 Load Balanced multicast Session in a partition p_i

Responsibility of a primary core C^i:

1. Receives the *join request* from R_f
2. if $sa_j \leq G_m < sa_{j+1}$

   Primary core C_i selects core C_j^i for multicast;

   C^i unicasts C_j^i to R_f

   // C^i may itself be C_i^i.

Responsibility of a first hop router R_f during a multicast session in a partition p_i:

1. First hop router R_f receives a join request with group address G_m from a source connected to it.
2. R_f identifies the largest SA_i in the vector VSA such that $SA_i \leq G_m$.
3. R_f identifies the primary core C^i for partition p_i to be used from the entry $CMT\ (i) = < SA_i$, *IP address of core* $C_i >$ in the core map table (*CMT*).
4. R_f unicasts *'join request'* to *core* C^i.
5. R_f receives the message *join core* C_j^i / C^i from primary core C^i.

   // Core C_j^i belongs to partition p_i.
6. R_f joins core C_j^i / C^i.
7. R_f unicasts 'multicast packets' to C_j^i / C^i.

   // R_f acts as new source for multicast group G_m.

Responsibility of Core C_j^i / C^i :

1. Core C_j^i / C^i multicasts the packet received from R_f for group G_m in the C_j^i / C^i – based tree

3.4. Some Further Discussion on the Performance

Any network consists of very large number of routers with asymmetric link connections. All existing eminent works have to determine the network topology for selecting the best core. Note that network topology changes even if a single link or router fails. Therefore, in that event, all existing works have to restart gathering the knowledge of all links and routers to build the changed topology again and then apply their methods to determine again a best core. This is clearly a very much impractical approach since it is quite time consuming to gather the required knowledge and somewhat a complicated way to determine the best core.

It is known that in any network, routers' routing tables are updated periodically or this updating process is initiated when a failure (link/router) is detected. In our work, we take advantage of the information present in the routing tables without any attempt to determine the current topology. We simply determine pseudo diameter of a router from the cost information present in its routing table. It is just the largest cost present in the router table. So, this determination is very trivial. Each router just broadcasts its pseudo diameter to all other routers. In effect, each router will have the same global information about all these pseudo diameters and eventually will come to the same conclusion without any further communication among them about the best core and for fault tolerant purpose the next few (based on the degree of fault-tolerance) best cores. Thus,a need to figure out the topology is totally absent in our work. Clearly it is highly efficient and very much practical, in addition simple to implement as well. We believe that efficiency of a load balanced multicast algorithm is intimately related to the way multiple cores are selected.

In Section 2, we have presented simulation results to determine cores [8] and have shown the superiority of the pseudo diameter-based core selection method over the most eminent and theoretically elegant existing works in this direction. To the best of our knowledge there does not exist any other theoretical work that can be considered as eminent as the ones which we have considered in our simulation.

Next, we have considered the following problem: given C number of cores, how to use them for simultaneous multicore multicasting without the requirement of having any knowledge of network topology. We have offered a practical and easy to implement method. It works as follows: all multicast addresses (e.g., all Class D addresses) are partitioned equally into C partitions with a possible exception that the last one may have a smaller number of multicast addresses. Then assign one core

uniquely to a partition such that all multicast communications which involve addresses in this partition will use this core only. Fact is, this assignment is done even before the network starts operating. This is an elegant practical approach and to the best of our knowledge there does not exist any such approach by others in the literature.

4 Fault Tolerance

Earlier we have assumed that the selected list of cores, L will have p partitions and each partition p_i will have q number of cores sorted in ascending order of their pseudo diameters. These q cores in a partition p_i will share responsibility of multicasting involving group addresses that will be either equal or greater than SA_i, but definitely less than SA_{i+1}. Note that the primary core of partition p_i is C^i, which we denote also as C_0^i in the following discussion. Let us denote these q cores as $C_0^i, C_1^i, C_2^i, \ldots, C_{(q-1)}^i$.

We now consider the following problem: what happens if any or all of these q cores become faulty at the same time. We offer the following solution. Instead of selecting r number of cores, we select $2r$ number of cores and each partition p_i will consist of $2q$ number of cores sorted in ascending order of their respective pseudo diameters. So the cores in partition p_i will now be: $C_0^i, C_1^i, C_2^i, \ldots, C_{(q-1)}^i, C_q^i, C_{(q+1)}^i, C_2^i, \ldots, C_{(2q-1)}^i$.

We divide these cores in groups of two. Thus, the cores in p_i will now appear as $[C_0^i, C_1^i], [C_2^i, C_3^i], \ldots, [C_{(q-2)}^i, C_{(q-1)}^i], \ldots\ldots, [C_{(2q-2)}^i, C_{(2q-1)}^i]$. In each group, the first core will multicast if it is not faulty, otherwise the second core will multicast. Therefore, the second core in a group is the standby core for the first one. *Therefore, our fault model is: in a group, only one core can be faulty; considering all groups in a partition at most* ***q*** *number of cores can be faulty and considering all partitions at most* ***p.q*** *cores can be faulty at any given time.*

Observe that according to the positions in the sorted list all odd-numbered cores share multicasting in absence of any core failures. Every odd-numbered core will have its standby as the even numbered core that immediately follows it in the sorted list of the cores; this standby core selection utilizes proximity between these two cores from the viewpoint of their pseudo diameter values.

One can extend the fault model to any level. For example, if we select 3q cores per partition, each group inside a partition will have three cores. The first core will multicast if it is fault free; otherwise the second core will multicast if this one is not faulty; otherwise the third one will multicast. *Therefore, the fault model is now at most 2q cores per partition can be faulty at the same time with a maximum of two faulty cores in a group.* In this way, one can extend the fault model further to enhance the reliability of multicast communication from the viewpoint of core failures. It is up to the designers of the protocol to choose the degree of fault tolerance. Observe that enhancing the degree of fault tolerance does not involve any extra cost during core selection process. We now state the modifications of the fault free protocol. We assume that in a group, only one core can be faulty; considering all groups in a partition, at most q number of cores can be faulty. In the following fault tolerant protocol, as mentioned above C^i and C_0^i denote primary core of a partition p_i interchangeably.

4.1 Candidate Cores Selection Process by Each First Hop Router

1. Each non-first hop router R_i determines its $P_d(R_i)$ from its unicast routing table.
2. It broadcasts its $P_d(R_i)$ in the network using pseudo diameter-based broadcasting [14].
3. Every first hop router R_f receives all $P_d(R_i)$ from every non-first hop router R_i.
4. Router R_f creates a list of r' routers out of all non-first hop routers, sorted in ascending order based on their P_d values.

   // these r' routers are used for multicore load balanced multicast.
5. Each first hop router R_f partitions the list into p equal parts, with 2q routers in each part, i.e. $r' = p.2q$.
6. R_f identifies the 1^{st} core in the i^{th} partition p_i as the primary core, C^i of partition p_i for $1 \le i \le p$.

   // P_d of C^i is minimum among the 2q cores in partition p_i.
7. R_f creates a list L' of the p primary cores corresponding to the p partitions along with their respective standby cores.

4.2 Static Partitioning of Multicast Addresses

1. Each first hop router R_f divides all group addresses into p distinct ranges (partitions) with starting address of the i^{th} partition as SA_i, $1 \leq i \leq p$.
2. Router R_f creates a vector (VSA) consisting of the starting addresses of the partitions.
   It is: $VSA = < SA_1, ..., SA_i, ..., SA_p >$ sorted in ascending order of their magnitudes.
3. Router R_f uses the list L' and the vector VSA to build a core map table (CMT) of size p such that its i^{th} entry is $CMT\ (i) = <$ *IP address of core* C^i, *IP address of standby core of* $C^i >$.

4.3 Load Balanced Multicast Session in a Partition p_i

Each primary core C^i maintains the following date structure:

1. Primary core C^i divides the address range SA_i to SA_{i+1} into q distinct ranges with starting addresses as $sa_1, sa_2, ..., sa_q$ sorted in ascending order
2. It builds the following array of q tuples $< [sa_1, C_0^i, C_1^i], [sa_2, C_2^i, C_3^i], ..., [sa_q, C_{(2q-2)}^i, C_{(2q-1)}^i]>$

Note that a core C_j^i $(1 \leq j \leq 2q-1)$ is responsible for multicasting packets to addresses in the range $[sa_j$ and $< sa_{j+1}]$ *if it is not faulty; otherwise its standby core* $C_{(j+1)}^i$ *will multicast.*

Responsibility of a first hop router R_f during a multicast session in a partition p_i :

1. First hop router R_f receives a join request with group address G_m from a source connected to it.
2. R_f identifies the largest SA_i in the vector VSA such that $SA_i \leq G_m$.
3. R_f identifies the primary core C^i for partition p_i to be used from the entry $CMT\ (i) = <$ *IP address of* C^i, *standby core of* $C^i >$ in the core map table (CMT).
4. R_f unicasts *'join request'* to *core* C^i *and to standby core* C_1^i.
   //to guard against the possibility of any one of these two cores being faulty
5. R_f receives the message *join core* C_j^i / $C_{(j+1)}^i$ from primary core C^i.
6. R_f joins core C_j^i / $C_{(j+1)}^i$.
7. R_f unicasts 'multicast packets' to C_j^i / $C_{(j+1)}^i$.
   // R_f acts as new source for multicast group G_m.

4.4 Responsibility of Primary Core C^i / its Standby Core C_1^i

1. Receives the *join request* from R_f
2. R_f determines $sa_j \leq G_m < sa_{j+}$

if primary core C^i is not faulty

if C_j^i is not faulty

primary core C^i unicasts to R_f the *IP address* of C_j^i

// first core in the j^{th} tuple is selected for multicast

else primary core C^i unicasts to R_f the *IP address* of standby core $C_{(j+1)}^i$

// standby core in the j^{th} tuple is selected for multicast

else

if C_j^i is not faulty

standby core C_1^i unicasts to R_f the *IP address* of C_j^i

else standby core C_1^i unicasts to R_f the *IP address* of standby core $C_{(j+1)}^i$

// $C^i(C_0^i)$ may itself be C_j^i ; C_1^i may itself be $C_{(j+1)}^i$

4.5 Responsibility of the Selected Core

1. The selected core multicasts the packet received from R_f for group G_m in the *selected core – based tree.*

5 Conclusion

In this paper, a new load-balanced multicast approach and its fault tolerant version have been presented. The noteworthy points of the approach are its multiple core selection method, partitioning of all possible multicast addresses, and a simple mapping of multicast group addresses to the selected cores based on the partitioning. For core selection, knowledge about the complete topology is not required unlike the existing works. Only the pseudo diameters are determined by the routers from their respective routing tables and are used to select the cores with complexity $O(n^2)$ [14]. In addition, partitioning of all possible multicast addresses is done by the first hop routers only when the network is booted and this partitioning is independent of any network topology. Because of these features the presented approaches appear quite practical. To the best of our knowledge, there does not exist any such multicast architecture for PIM SM except the one in [2] that uses static partitioning of all possible multicast addresses for multicore multicasting / load sharing.

References

[1] Andrew Adams, Jonathan Nicholas, and William Siadak, "Protocol Independent Multicast - Dense Mode (PIM-DM)", Internet Engineering Task Force (IETF), RFC-3973, January 2005.

[2] Ashraf Alyanbaawi, Indranil Roy, Swathi Kaluvakuri, Koushik Maddali, Bidyut Gupta, and Narayan Debnath, "A Novel Multicore Multicasting Scheme for PIM-SM, EMENA-ISTL", 2019; LAIS 7 (Learning and Analytics in Intelligent Systems 7, Springer, pp. 631- 638, 2020.

[3] Tony A. Ballardie, "Core Based Tree Multicast Routing Architecture", Internet Engineering Task Force (IETF), RFC 2201, September 1997.

[4] Stephen E. Deering and David R. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs", *ACM Transactions on Computer Systems (TOCS)*, 8(2):85-110, May 1990.

[5] Deborah Estrin, Mark Handley, Ahmed Helmy, and Polly Huang, "A Dynamic Bootstrap Mechanism for Rendezvous-Based Multicast Routing", *Proc. IEEE INFOCOM*, pp. 1090-1098, 1999.

[6] Bill Fenner, Mark Handley, Hugh Holbrook, and Isidor Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM)", Internet Engineering Task Force (IETF), RFC-4601, August 2006.

[7] B. Gupta, A. Alyanbaawi, N. Rahimi, K. Sinha, and Z. Liu, "Novel Low Latency Load Shared Multicore Multicasting Schemes ─ An Extension to Core Migration", *IJCA*, 25(3):132-141, Sept. 2018.

[8] B. Gupta, A. Alyanbaawi, S. Rahimi, N. Rahimi, and K. Sinha, "An Efficient Approach for Load- Shared and Fault-Tolerant Multicore Shared Tree Multicasting", *Proc. IEEE INDIN*, Emden, Germany, pp. 937-943, July 2017.

[9] Bidyut Gupta, Sindoora Koneru, and Shahram Rahimi, "Multicast Routing Protocol for Computer Networks", US Patent No. 9,461,832 B2, Oct. 2016.

[10] W. Jia, W. Tu, W. Zhao, and G. Xu, "Multi-Shared-Trees Based Multicast Routing Control Protocol Using Anycast Selection", *The International Journal of Parallel,*

Emergent and Distributed Systems, Taylor & Francis, 20(4):69-84, March 2005.
[11] Weijia Jia, Wei Zhao, Dong Xuan, and Gaochao Xu, "An Efficient Fault-Tolerant Multicast Routing Protocol with Core-Based Tree Techniques", *IEEE Trans. on Parallel and Distributed Systems*, 10(10):984-1000, October1999.
[12] A. Karaman and H. Hassanein, "Core Selection Algorithms in Multicast Routing – Comparative and Complexity Analysis", *J. Computer Communications*, 29(8):998-1014, May 2006.
[13] Sindoora Koneru, Bidyut Gupta, Shahram Rahimi, and Ziping Liu, "Hierarchical Pruning to Improve Bandwidth Utilization of RPF-Based Broadcasting", *IEEE Symposium on Computers and Communications (ISCC)*, Split, Croatia, pp. 96-100, July 2013.
[14] Sindoora Koneru, Bidyut Gupta, Shahram Rahimi, Ziping Liu, and Narayan Debnath, "A Highly Efficient RPF-Based Broadcast Protocol Using a New Two-Level Pruning Mechanism", *Journal of Computational Science (JOCS)*, 5(3):645-652, March 2014. SpringerLink, Berlin, Heidelberg, 2345:1045-1056, 2002.
[15] Hwa-Chun Lin and Shou-Chuan Lai, "A Simple and Effective Core Placement Method for the Core-Based Tree Multicast Routing Architecture", *Proc. IEEE Int. Conf. Performance, Computing, and Communications*, pp. 215-219, February 2000.
[16] "Load Splitting IP: Multicast Traffic over ECMP. Cisco", mcast/configuration/guide/mctlsplt.html, Mar 2015.
[17] Tom Pusateri, "Distance Vector Multicast Routing Protocol", Juniper Networks, Internet Engineering Task Force (IETF), draft-ietf-idmr-dvmrp-v3-11.txt, October 2003.
[18] George N. Rouskas and Ilia Baldine, "Multicast Routing with End-to-End Delay and Delay Variation Constraints, *IEEE Journal on Selected Areas in Communications*, 15(3):346-356, April 1997.
[19] C. Shields, J. J. Garcia-Luna-Acevez, "The Ordered Core-Based Tree Protocol", *Proc. IEEE INFOCOM*, pp. 884-891, 1997.
[20] Young-Chul Shim and Shin-Kyu Kang, "New Center Location Algorithms for Shared Multicast Trees", Lecture Notes in Computer Science, SpringerLink, Berlin, Heidelberg, 2345:1045-1056, 2002.
[21] David G. Thaler and Chinya V. Ravishankar, "Distributed Center-Location Algorithms", *IEEE Journal on Selected Areas in Communication,* 15(3):291- 303, April 1997.
[22] David Waitzman, Craig Partridge, and Stephen E. Deering, "Distance Vector Multicast Routing Protocol (DVMRP)", Internet Engineering Task Force (IETF), RFC 1075, November 1988.
[23] D. Zappala, A. Fabbri, and V. Lo, "An Evaluation of Shared Multicast Trees with Multiple Active Cores", *Journal of Telecommunication Systems*, pp. 461-479, March 2002.

Indranil Roy is currently a PhD student in Computer Science department of Southern Illinois University, Carbondale. He has completed his Bachelors of Technology in Electronics & Communication from RCCIIT, Kolkata, in the year 2016. He received his M.S in Computer Science degree from Southern Illinois University, Carbondale. His main research interests include blockchain along with interest-based P2P architecture.

Swathi Kaluvakuri (photo not available) is a Ph.D. candidate from Southern Illinois University Carbondale – School of Computing. She graduated from Jawaharlal Nehru Technological Unversity with a Bachelor of Technology degree in Computer Science. She holds a keen interest in the areas of Peer-to-Peer Networking and BlockChain and worked as a Software Engineer, Technical Product Support and IBM AS400 developer for NetCracker Pvt Ltd from 2012-2014.

Koushik Maddali (photo not available) is a Ph.D. candidate in Department of Computer Science at Southern Illinois University Carbondale. He received his MS from the same university and his BS from Jawaharlal Nehru Technological University, India. His research interests include Peer to Peer Networking, BlockChain and worked on a Virtual Terminal project of Cisco from 2017-2018.

Ziping Liu is currently a Professor of Computer Science at Southeast Missouri State University. Her research interests include wireless ad-hoc network/sensor network communication and security, distributed computing and cloud Computing, machine learning, mobile and full-stack application development, and secured software design

Bidyut Gupta (photo not available) received his M. Tech. degree in electronics engineering and Ph.D. degree in Computer Science from Calcutta University, Calcutta, India. At present, he is a professor at the School of Computing (formerly Computer Science Department), Southern Illinois University, Carbondale, Illinois, USA. His current research interest includes design of architecture and communication protocols for structured peer-to-peer overlay networks, security in overlay networks, and block chain. He is a senior member of IEEE and ISCA.

Narayan Debnath (photo not available) earned a Doctor of Science (D.Sc.) degree in Computer Science and also a Doctor of Philosophy (Ph.D.) degree in Physics. Narayan C. Debnath is currently the Founding Dean of the School of Computing and Information Technology at Eastern International University, Vietnam. He is also serving as the Head of the Department of Software Engineering at Eastern International University, Vietnam. Dr. Debnath has been the Director of the International Society for Computers and their Applications (ISCA) since 2014. Formerly, Dr. Debnath served as a Full Professor of Computer Science at Winona State University, Minnesota, USA for 28 years (1989-2017). Dr. Debnath has been an active member of the ACM, IEEE Computer Society, Arab Computer Society, and a senior member of the ISCA.

Index

Authors

A-B

C

D-G

H

I-L

M-O

Key Words

Journal Submission

The International Journal of Computers and Their Applications is published four times a year with the purpose of providing a forum for state-of-the-art developments and research in the theory and design of computers, as well as current innovative activities in the applications of computers. In contrast to other journals, this journal focuses on emerging computer technologies with emphasis on the applicability to real world problems. Current areas of particular interest include, but are not limited to: architecture, networks, intelligent systems, parallel and distributed computing, software and information engineering, and computer applications (e.g., engineering, medicine, business, education, etc.). All papers are subject to peer review before selection.

A. Procedure for Submission of a Technical Paper for Consideration

1. Email your manuscript to the Editor-in-Chief, Dr. Ziping Liu at: zliu@semo.edu.
2. Illustrations should be high quality (originals unnecessary).
3. Enclose a separate page (or include in the email message) the preferred author and address for correspondence. Also, please include email, telephone, and fax information should further contact be needed.
4. **Note**: Papers shorter than 10 pages long will be returned.

B. Manuscript Style:

1. **WORD DOCUMENT**: The text should be **double-spaced** (12 point or larger), **single column** and **single-sided** on 8.5 X 11 inch pages. Or it can be single spaced double column.

   LaTex DOCUMENT: The text is to be a double column (10 point font) in pdf format.
2. An informative abstract of 100-250 words should be provided.
3. At least 5 keywords following the abstract describing the paper topics.
4. References (alphabetized by first author) should appear at the end of the paper, as follows: author(s), first initials followed by last name, title in quotation marks, periodical, volume, inclusive page numbers, month and year.
5. The figures are to be integrated in the text after referenced in the text.

C. Submission of Accepted Manuscripts

1. The final complete paper (with abstract, figures, tables, and keywords) satisfying Section B above in **MS Word format** should be submitted to the Editor-in-Chief. If one wished to use LaTex, please see the corresponding LaTex template.
2. The submission may be on a CD/DVD or as an email attachment(s). **The following electronic files should be included:**
   - Paper text (required).
   - Bios (required for each author).
   - Author Photos are to be integrated into the text.
   - Figures, Tables, and Illustrations. These should be integrated into the paper text file.
3. Reminder: The authors photos and short bios should be integrated into the text at the end of the paper. All figures, tables, and illustrations should be integrated into the text after being mentioned in the text.
4. The final paper should be submitted in (a) pdf AND (b) either Word or LaTex. For those authors using LaTex, please follow the guidelines and template.
5. Authors are asked to sign an ISCA copyright form (http://www.isca-hq.org/j-copyright.htm), indicating that they are transferring the copyright to ISCA or declaring the work to be government-sponsored work in the public domain. Also, letters of permission for inclusion of non-original materials are required.

Publication Charges

After a manuscript has been accepted for publication, the contact author will be invoiced a publication charge of **$500.00 USD** to cover part of the cost of publication. For ISCA members, publication charges are **$400.00 USD** publication charges are required.

Revised 2020