Graduate School ETD Form 9
(Revised 12/07)

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Vidya Bhushan Singh

Entitled
User Modeling and Optimization for Environmental Planning System Design

For the degree of Master of Science

Is approved by the final examining committee:

Snehasis Mukhopadhyay
Chair

Mihran Tuceryan

Yuni Xia

To the best of my knowledge and as understood by the student in the *Research Integrity and Copyright Disclaimer (Graduate School Form 20)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): Snehasis Mukhopadhyay

Approved by: Shiaofen Fang 02/19/2014
Head of the Graduate Program Date

USER MODELING AND OPTIMIZATION FOR ENVIRONMENTAL PLANNING SYSTEM DESIGN

A Thesis

Submitted to the Faculty

of

Purdue University

by

Vidya Bhushan Singh

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

May 2014

Purdue University

Indianapolis, Indiana

I would like to dedicate my work to
my father Vijay Bahadur Singh,
my mother Sona Mukhi Devi,
my brother and sisters.

ACKNOWLEDGMENTS

There are a number of people who helped me in performing different parts of this research and I would like to give my humble gratitude for their valuable help.

First of all I want to thank my thesis adviser and my mentor Dr. Snehasis Mukhopadhyay for providing me this opportunity to be part of this research team. Without his valuable support and guidance I would not have reached my goals.

I would also like to thank Dr. Meghna Babbar-Sebens for her valuable guidance in the development of the system and keeping me inspired for doing this research.

I want to thank Dr. Rajeev Raje for inspiring me in designing this system and his teachings about Distributed System, without which it would not have been easy to design the system and perform this research.

I would like to thank Dr. Yuni Xia for her valuable teachings about creating and managing databases, which helped me a lot in designing and managing the system. Her guidance was also very valuable.

I would like to thanks Dr. Mihran Tuceryan for his valuable teachings about operating systems, as a member of my thesis committee, his guidance was very valuable.

Adriana Debora Piemonti helped me in the research by creating different environmental models, without which, performing the simulations would not have been possible.

I want to thank the Department of Computer and Information Science and the National Science Foundation for funding the research under grant number 1332385 and providing the support for this research.

I would like to thank our collaborators Dr. Jane Luzar, Eagle Creek Watershed Alliance, Upper White River Watershed Alliance, and Indiana NRCS for their support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

ABBREVIATIONS

ANFIS	Adaptive Network based Fuzzy Inference System
ANN	Artificial Neural Network
AS	Automated Search
BMP	Best Management Practice
CBM	Case Based Memory
DPLA	Decentralized Pursuit Learning Algorithm
Dist. DPLA	Distributed DPLA
Dist. NSGAII	Distributed NSGAII
GA	Genetic Algorithm
HDM	Human Decision Maker
HPC	High Performance Computing
HS	Human Search
IGA	Interactive Genetic Algorithm
IGAMI2	Interactive Genetic Algorithm with Mixed Initiative 2
IM	Introspection Manager
LA	Learning Automata
MIM	Mixed Initiative Manager
NSGAII	Nondominated Sorting Genetic Algorithm II
OM	Optimization Manager
RLBM	Reinforcement Learning in Batch Mode
SDM	Simulated Decision Maker
SWAT	Soil and Water Assessment Tool

ABSTRACT

Singh, Vidya Bhushan M.S., Purdue University, May 2014. User Modeling and Optimization for Environmental Planning System Design. Major Professor: Snehasis Mukhopadhyay.

Environmental planning is very cumbersome work for environmentalists, government agencies like USDA and NRCS, and farmers. There are a number of conflicts and issues involved in such a decision making process. This research is based on the work to provide a common platform for environmental planning called WRESTORE (Watershed Restoration using Spatio-Temporal Optimization of Resources). We have designed a system that can be used to provide the best management practices for environmental planning. A distributed system was designed to combine high performance computing power of clusters/supercomputers in running various environmental model simulations. The system is designed to be a multi-user system just like a multi-user operating system. A number of stakeholders can log-on and run environmental model simulations simultaneously, seamlessly collaborate, and make collective judgments by visualizing their landscapes. In the research, we identified challenges in running such a system and proposed various solutions. One challenge was the lack of fast optimization algorithm. In our research, several algorithms are utilized such as Genetic Algorithm (GA) and Learning Automaton (LA). However, the criticism is that LA has a slow rate of convergence and that both LA and GA have the problem of getting stuck in local optima. We tried to solve the multi-objective problems using LA in batch mode to make the learning faster and accurate. The problems where the evaluation of the fitness functions for optimization is a bottleneck, like running environmental model simulation, evaluation of a number of such models in parallel can give considerable speed-up. In the multi-objective LA, different weight pair solu-

tions were evaluated independently. We created their parallel versions to make them practically faster in computation. Additionally, we extended the parallelism concept with the batch mode learning. Another challenge we faced was in User Modeling. There are a number of User Modeling techniques available. Selection of the best user modeling technique is a hard problem. In this research, we modeled user's preferences and search criteria using an ANN (Artificial Neural Network). Training an ANN with limited data is not always feasible. There are many situations where a simple modeling technique works better if the learning data set is small. We formulated ways to fine tune the ANN in case of limited data and also introduced the concept of Deep Learning in User Modeling for environmental planning system.

1 INTRODUCTION

This thesis is based on the work we did for designing an Environmental Planning System. The Environmental planning is a decision making mechanism that considers several parameters as social, political, economic, and governance factors to plan various practices to protect the interests of both the natural environment and the public by making compromising but optimal decisions. The system being built is a decision support system that can be used to make environmental planning decisions.

We are working on this environmental watershed optimization problem to provide the best farming practices so as to minimize the soil erosion, fertilizer loss, and maintain water quality of the region while maximizing the profit of farmers. We are extending the work of [1] by implementing a similar concept to provide the Best Management Practices (BMP) to the Eagle Creek Watershed, located North West of Indianapolis, shown in Fig. 1.1. The entire Eagle Creek is divided into total 2,953 potential wetlands. A distributed hydrology model was built using SWAT (Soil and Water Assessment Tool) [2] and [3]. The total wetlands are divided into 130 aggregated wetlands as shown in Fig.5.1. To perform the optimization efficiently, we have developed a system that uses the fast computing power of clusters/supercomputers and involves human users to visualize the problems and help the overall search process.

The decision support system is needed because the environmental planning is a difficult work especially due to a number of conflicting issues between different stakeholders. Due to lack of proper planning and management system, the effect on the natural environment is very severe. We frequently have the problem of flooding or drought in many regions, due to which the climate of the region is unpredictable. The flooding of the region reduces the fertility of the soil, decrease the quality of drinking water and many other problems while drought causes decrease in the harvest of crops

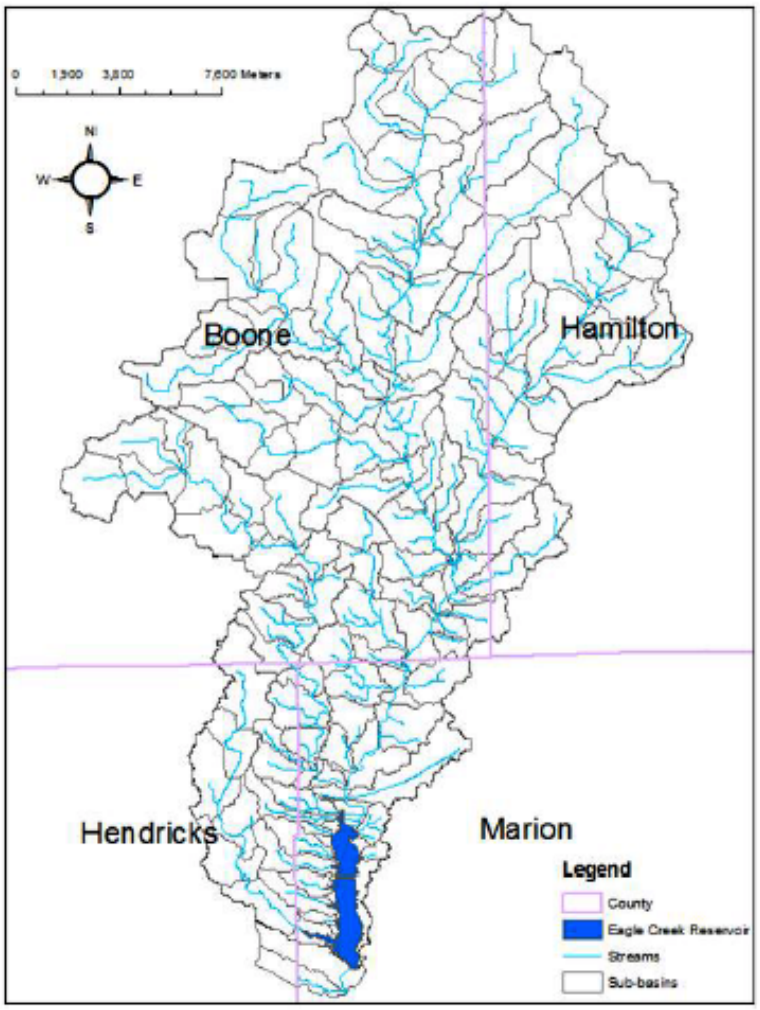

Photo credit: *Ref. Omkar [4]*

Figure 1.1. Counties, Reservoir, Streams of Eagle Creek Watershed.

that impact the economy. One possible way of solving this situation is to reduce the amount of flood damage by storing the excess floodwater.

So the farmers, the environmentalists, the government agencies like USDA, and NRCS need a decision support system that they can use to visualize the landscape by implementing certain practices and make a collective decision. e.g., a network of wetlands can be designed that improves the upland storage in the watershed.

Now there are many ways a particular practice can be implemented, and there are an infinite number of possible solutions. The problem is how to decide what particular solution to choose and how to minimize the number of possible solutions. The Human guided search based interactive optimization is one of the techniques that can be used.

Human guided search is a method of solving problems whose search space is very large. The search space may be so broad that neither a human nor the most powerful supercomputer alone can solve the problem because it is an NP-Hard problem. The human brain is very good in visualizing data for analysis [5]. If the NP-Hard problem involves visualizing the solutions, then a human can participate in the search process in and during the advisory role [6]. The expectation is that the human will help the search to proceed in the correct direction and decrease the total search space that will help the computer to solve the problem in a reduced amount of time.

In interactive optimization, a human user's preference is considered to compute the design alternatives so that the optimization algorithm will generate the optimal solution of design. It may not be the best overall optimal design for the user, but because it uses the user specified criterion, the total search space will be decreased as in a greedy choice. This can be visualized as user trimming the search space based on his/her preference criterion as given in Trade-off Plane Cutting Algorithm [7].

But there is a limitation beyond which a human can not participate in the search completely, due to human fatigue [8]. So some mechanism is needed to relieve the human in a long search process. One way of doing this is by learning the choices made by the human using some machine learning algorithms, such as an ANN, fuzzy

logic, or other linear/non-linear modeling techniques. That will create a surrogate function [8] which we called Simulated Decision Maker (SDM). The SDM is used to search a large set of population of designs and provide feedback on behalf of the user. There are many machine learning techniques that can be used to model the user's preferences, but finding the most suitable machine learning algorithm in itself is a hard problem especially if the domain's (e.g., environmental planning) data is highly stochastic. Although the user's preference criterion is consistent for a short duration in which the domain's search data is not very stochastic, over a long time scale it may exhibit randomness and non-stationary behavior. These preferences are modeled using a robust, noise-tolerant machine learning techniques such as ANN and fuzzy logic, in order to use them in an efficient search and optimization.

The same approach can also be extended to search with multiple human users, in a collaborative problem-solving approach. The preference considerations of multiple users should decrease the overall search space based on overlapping preference criterion as in the collaborative filtering based technique [9]. The preferences vary based on users, as a landowner, stakeholders, or government agencies like USDA and NRCS, and farmers. Finding their priorities and constraints universally is not an easy task.

In this research, we designed a distributed system that involves multiple computers, and clusters to run various environmental model simulations in an efficient manner. We needed a distributed system because running a large number of environmental model simulations and managing data of multiple users is not possible for ordinary computers. Different components of the distributed system were designed based on the need and availability of resources. More description of the distributed system is given in the upcoming sections.

During the design of the system, we encountered several challenges. Out of those, two big challenge viz. a faster optimization algorithm [10] and user modeling [11] are discussed, and some solutions are proposed in this thesis.

2 RELATED WORK

There are many environmental planning systems built in the past for various different kinds of optimization.

The interactive genetic algorithm (IGA) has been used in Ground Water Monitoring problems [1], in which the mixed initiative approach was used to solve the multiobjective problem. Their concept was, in a search there are many qualitative criteria that cannot be expressed using any mathematical model. The real human is much better in understanding the qualitative criteria. The human expert can participate in the search process using interactive search algorithms like Genetic Algorithm (GA).

Human Guided Search is being performed by [12] in which the computer finds the local minimal, using simple hill climbing search, while the human directs the search using visualization so that the search can escape the local minimal.

Reinforcement Learning is used by [13] and [14] as Learning Automata (LA) learning technique. LA takes a lot of time to converge to a good solution, so multiple studies have investigated faster learning approaches via parallelization [13]. The parallelization techniques were done for common pay-off games, parameterized learning automata and pattern classification problems. The motivation [13] was N independent Agents acting simultaneously should speed-up a process roughly by a factor of N, similar to a population as in GA. But this parallelization was tested for direct learning algorithm such as LRI Algorithm. In our work, the N LA's take M independent actions at a time. Combining all the automatons and their actions will create a set of actions to be performed by the system, this is like an individual in GA or one individual design of SWAT [15]. In Batch Mode, we created multiple such individual designs. As execution and reward for each individual design is independent of each

other, a PRAM (Parallel Random Access Machines) based algorithm can be used to evaluate them independently.

Batch mode learning is one of the popular learning techniques used in ANN. In batch mode [16], the learning of the ANN is done by taking the average over all training patterns before changing the weights. Choosing a very small learning parameter is not realistic as smaller the learning, the longer it takes to converge. So based on availability of resources and time, learning parameter can be chosen. Batch Mode learning is also done in text categorization [17], where a batch of text documents is used instead of just one. Batch Mode learning is used in medical Image classifications [18] and Content based Image retrieval [19] where instead of selecting a single unlabeled example, a number of unlabeled examples were selected for manual labeling. A discriminative batch mode active learning is done by [20].

People also have tried to combine the LA and GA [21] to escape the problem of local optima. In StGA [22], the authors created a small number of actions of the Learning Automata, which were sampled to construct a population, from which the sampling action was done adaptively by genetic operations. Some authors have also tried to combine the GA with other algorithms as Simulated Annealing to solve some NP-Hard Problems [23].

Another big challenge of performing interactive optimization is User Modeling. The challenge in our case is limited feedback data from the user. There are many scenarios where the total data we have will not be the proper distribution of human ratings. In such a case, an ANN trained network might get stuck to one particular rating. So we formulated a technique of user modeling where this scenario can be avoided, in case of limited data.

A lot of research on user modeling has been done in many disciplines [24] like Human-Computer Interaction, Intelligent Interfaces, Adaptive Interfaces, Cognitive Engineering, Intelligent Information Retrieval, Intelligent Tutoring, Active and Passive Help Systems, Guidance Systems, Hypertext Systems, and Expert Systems, to name just the most prominent application areas. But the user modeling in the domain

of Environmental Science is still in its infancy. User modeling is performed in information retrieval systems [25] where the user's domain experiences and inquiry interests are modeled by the system. Another promising method of user modeling, using an Artificial Neural Network (ANN), was done by [26] where news were shown to the user based on personal interest. The ANN tries to model the interests of the user and then rank the incoming news as relevant or irrelevant. ANN and GA hybrid, ANN-GA [27], is used to solve water quality modeling problems. The hybrid system was created because the total search space was too big and computationally not feasible. Fuzzy Model and ANN are used in estimating the sediment concentration [28]. ANN is used to model rainfall/runoff [29] and to model daily sediment yield [30] in various fields of hydrology. People [31] compared user modeling techniques like ANN, fuzzy logic, etc. for adaptive hypermedia systems and recommended their usage in different scenarios. Huang et.al. [32] used ANN to do multi-objective interactive optimization for reliability optimization.

Some of the popular predictive statistical models are linear models, TFIDF-based models, Markov models, ANN, classification and rule-induction methods, and Bayesian networks [33]. Of those models, ANN is good in expressing non-linear decision [33]. The two of the main issues that user modeling faces are user model representation and acquisition [34]. Generic user modeling technique has been developed by [35]. A user modeling toolkit for cooperative user modeling is done by [36].

Deep Learning is also one of prominent machine learning algorithm used by many people especially in handwriting recognition [37] and [38]. A faster implementation of Deep Learning is done [39] using GPUs, which are fast computing hardware.

In chapter 3, we have provided the description of the system we have built. In chapter 4, we have described the environmental resource optimization problem. In chapter 5, we have discussed various optimization algorithms we can use and proposed our new optimization algorithm. In chapter 6, we have discussed different machine

learning algorithm that we can use. In chapter 7, we discussed different experiments that we performed.

3 THE IGAMI2 DISTRIBUTED SYSTEM

IGAMI2 is Interactive Genetic Algorithm with Mixed Initiative (version 2). We redesigned the IGAMII system used in [1], with a lot of new modifications and added many important features (described below) based on the new set of problems we identified.

The IGAMI2 system is a distributed system which spans a number of computers, clusters, and supercomputers that can be used to run a number of environmental simulations, manage data and provide the decision support for environmental planning. The abstract diagram of the IGAMI2 System is given in Fig. 3.1.

Using this system, we evaluate multiple different environmental designs using faster cluster computers. The designs are shown to the users for their feedback. Their feedback is then used to train the simulated decision maker (SDM), which tries to mimic the preferences made by the user. A set of small searches is performed to learn the preferences of the user. Based on training and learning speed of the SDM, if the learning error is acceptable, then the SDM will be started to search a better set of designs on behalf of the user with a very large set of designs. When the search finishes, the set of optimal solutions is shown to the user, to contemplate with their opinion. The similar process is repeated a number of times, unless the user is satisfied with a sufficient number of good designs.

The IGAMI2 Kernel and DB Server runs on the Master computer while the actual model simulation is done using the HPC System.

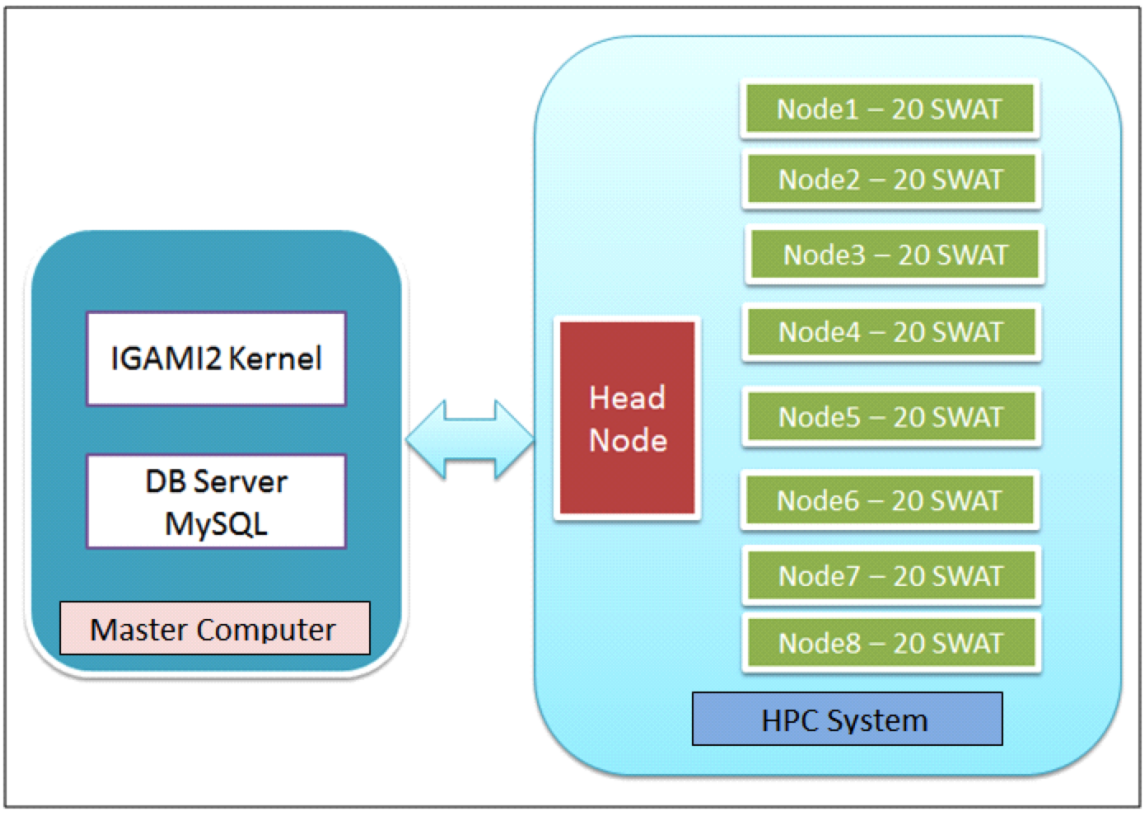

Figure 3.1. IGAMI2 System Abstract Diagram

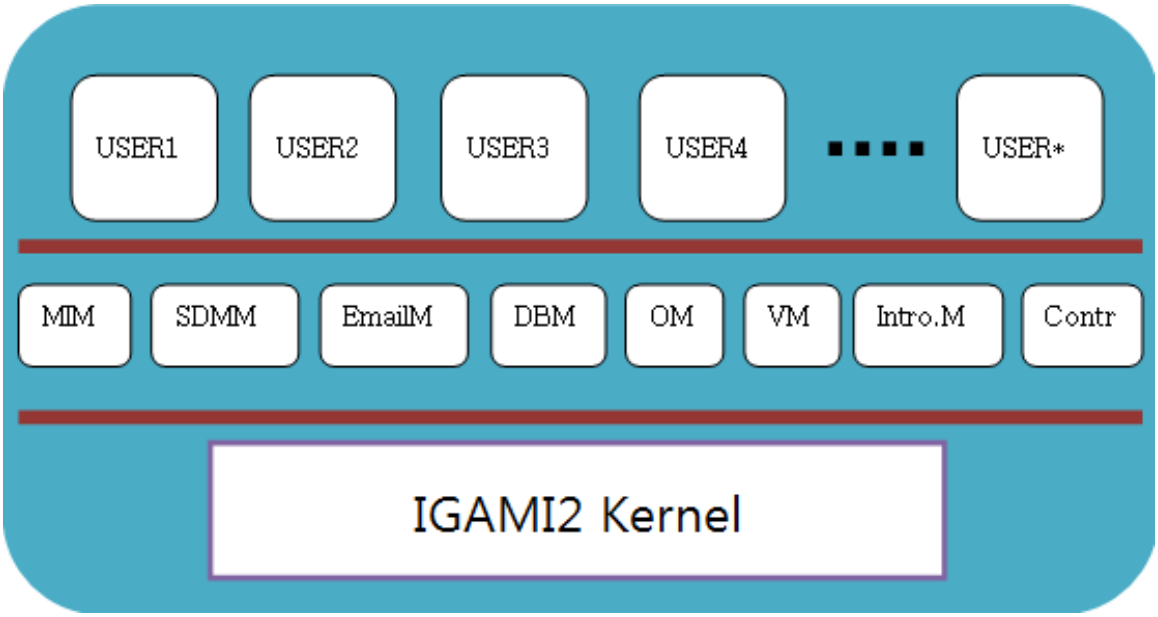

Figure 3.2. IGAMI2 Main Kernel

3.1 Main Components of the System

a. IGAMI2 Kernel: The core of the system is the IGAMI2 Kernel as shown in Fig. 3.2, whose job is to manage different components of the system as described below:

   i. IGAMI2 Main (kernel): The main program that adds and removes users from the system.

   ii. Mixed Initiative Manager (MIM): It manages the mixed initiative approach of the search, i.e., either Human Search or Automated Search. It acts as a central controller during the search.

   iii. SDM Manager (SDMM): It manages different Simulated Decision Makers (SDM), which are created using different machine learning algorithms. The best SDM is chosen to perform an Automated Search on behalf of human.

   iv. Email Manager (EmailM): It handles the emails of the system.

   v. HPC Controller: It is used to connect the system to clusters/supercomputers, allocate and manage various clusters and clouds.

   vi. DB Manager (DBM): It manages the data and interacts with the database, Hibernate framework is also used.

   vii. Optimization Manager (OM): It manages different types of optimization algorithms. Currently, we are using a genetic algorithm - NSGA2 [40] as the main optimization algorithm, but other optimization algorithms can be used simultaneously or inter-mixing with other optimization algorithms.

   viii. Visualization Manager (VM): It is used to display data in various different ways. It can also be used to show the simulation in 3D.

   ix. Introspection Manager (Intro.M): It manages different introspection sessions during the search. An Introspection session is the session in which the user contemplate with their previously found solutions during the search.

x. Individual Design Manager (IDM): It manages different individual designs and perform a number of operations.

b. Data Base Server: It manages the database.

c. High Performance Computing (HPC) system: The powerful computers running the actual simulation. Currently we are using cluster computers to run our model simulations and Big Red II supercomputer for data analysis. The system also supports heterogeneous systems and cloud computers.

d. Web Server (show the User Interface): The web server hosts the website from where users can visualize the designs and give their feedback.

3.2 The Working of IGAMI2

The working of the IGAMI2 system is shown in the Fig. 3.3. A number of users log-in via the User Interface and start their search by specifying their search parameters. The IGAMI2 kernel initiates a search for every user. An initial set of designs is created for each user. The user gives ratings to those designs using the Visualization Interface as shown in Fig. 4.10 and Fig. 4.11. The ratings of the user are used to perform a non-dominated sorting and an initial search begins. A NSGA2 search is initiated with small population fed from the initial set of designs. The HPC Controller sends the designs for evaluation to the HPC System. Once the evaluation of the models finishes the results are passed to respective users for their feedback.

Steps Involved in the IGAMI2 Search are:

i) A user log-in and start the search by specifying the search parameters using the visualization interface via the Web Server.

ii) The IGAMI2 Kernel creates a user search by initializing Managers - MIM, DBM, IM and IDM; and calls the MIM.

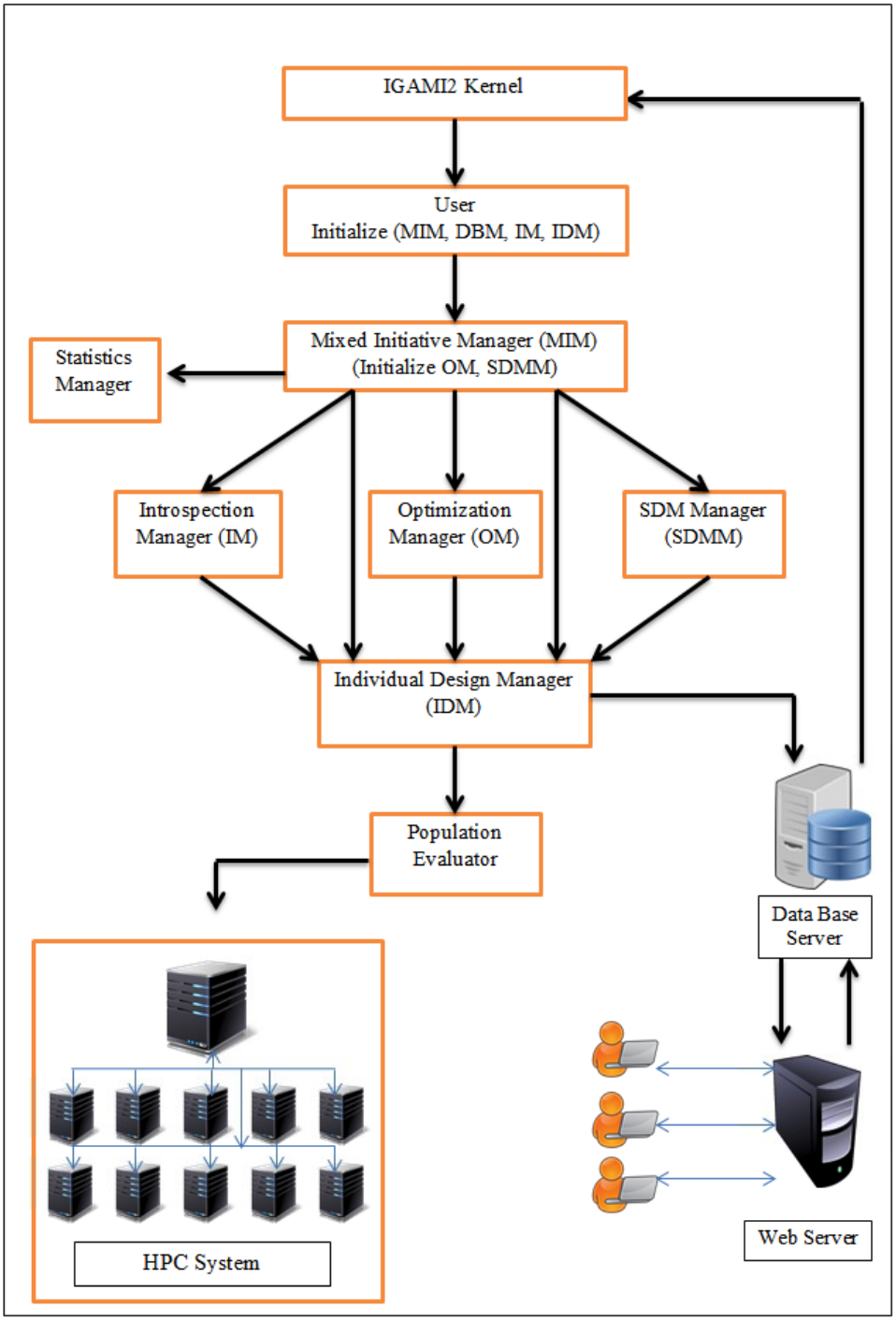

Figure 3.3. Working of IGAMI2

iii) The MIM initializes different search parameters and initializes other Managers - OM, SDMM, and STATM and starts the search.

iv) Search begins with first Introspection. MIM load the initial set of designs from Case Based Memory (CBM) (located in Data Base) and calls the Introspection Manager (IM).

v) The IM sends the designs for Human User's feedback using IDM.

vi) The data received from first Introspection session is saved to CBM.

vii) The MIM begins the NSGA2 search by creating initial set of parent population that is fed with 20 % of designs from the CBM.

viii) The Population Evaluator sends the population to the HPC system for evaluation.

ix) The evaluated population is sent for Human User's feedback using IDM.

x) After the users have given their feedback, the designs are saved to the HDM Archive.

xi) The MIM calls the Optimization Manager to begin the NSGA2 optimization.

xii) A new generation of child population is being created from the parent.

xiii) The Population Evaluator sends the child population to the HPC system for evaluation.

xiv) The evaluated population is sent for Human User's feedback using IDM.

xv) After the user gave their feedback, the designs are saved to HDM Archive (located in the Data Base).

xvi) A non-dominated sorting is performed by the NSGA2 to find the optimal designs for current generation.

xvii) The Statistics Manager (STATM), for further analysis, saves the users average confidence values.

xviii) The steps xii) to xvii) are repeated until enough number of generations reached.

xix) After the Optimization is finished, the final population of the NSGA2 search is saved to the CBM.

xx) The Introspection Manager performs a second introspection from the CBM.

xxi) The MIM calls the SDM Manager to perform various machine learning algorithms to find a better SDM.

xxii) A test is performed, to check the next initiative to be HS (Human Search) or AS (Automated Search), based on the number of searches being performed and change in user's confidence values.

xxiii) In case of a Human Search (HS), everything is repeated from step vii) to xxii).

xxiv) In case of an Automated Search (AS), the same process is repeated from step vii) but instead of human giving the rating, the best SDM Model performs the rating on behalf of the human user.

xxv) The search stops after the last Introspection finish.

Further description is given in MIM section below.

The following sections give a short description of different Managers, their roles and how they work:

3.3 IGAMI2 Kernel

IGAMI2 Kernel is the core of the distributed system. It runs on the Master Computer of the system.

3.3.1 Roles of IGAMI2 Kernel

a) Wait for any new user to log-in to the system.

b) Allocate System Resources to the new user like unique system Id.

c) Deallocate system resources and do system wide cleanup when the user abort or finish the search.

d) Forcefully remove the user if needed by the Admin.

e) Add/remove availability of multiple clusters at run-time it helps in better cluster management.

f) Monitor changes in Clusters like decrease or increase in number of nodes.

3.3.2 Working of IGAMI2 Kernel

a) The kernel periodically monitor for log-in of a user.

b) When a new user log-in via Webserver, IGAMI2 Kernel Allocates the resources for the user.

c) The user process begins which does the initialization of different Managers for the user.

d) The MIM is called which beings the search for the user.

e) Kernel periodically monitors for any changes in the configuration of the HPC System.

f) Kernel also monitors if a user has to be removed from the system.

g) When a user finish or abort their search or if user has to be removed forcefully, the kernel free the system resources and remove the user from the system.

3.4 DBManager

The job of the DBManager is to manage the database connection, store and retrieve the user's data. It passes the user's data from the system to the database and if needed load the data from the database to the system. If we have multiple databases the DBManager will keep track of their usage. Apart from traditional jdbc connection, I have also used Hibernate to implement the POJO (Plain Old Java Object) feature of the Java. The System work with both the connections.

3.5 Simulated Decision Maker Manager (SDMM)

It acts as a manager to manage different types of simulated decision maker(SDM).

3.5.1 Roles of SDM

a) Manage different types of SDMs.

b) Normalize training data.

c) Perform SDM rating during the Automated Search. The best SDM is used to perform the rating.

3.5.2 Working of SDM

a) SDMM is called by MIM to create SDMs.

b) It normalizes the data for training and testing.

c) The SDMM initialize different Neural Net Managers (NNM) for creating different types of ANN.

d) Create Different linear/non-linear models.

e) Create ANFIS Model if needed.

f) Check total test error and find the best ANN, which can be used to do SDM Rating.

g) Do the SDM Rating during the Automated Search.

3.6 Individual Design Manager (IDM)

It acts as an intermediary of storing data in memory, moving it between different managers and performs a number of operations during the search. The IDM stores all the individual design solutions generated during the search and pass it between different managers as needed.

3.7 Mixed Initiative Manager (MIM)

Mixed Initiative Manager act as a central manager to control all the other Managers, perform search for the user.

3.7.1 Roles of MIM

a) Manager all other Managers.

b) Configure Search parameters, local subbasin information, etc. for the user.

c) Initialize different managers for the user.

d) Decide next initiative for the search, whether a Human Search or an Automated Search.

e) Call different managers based on their need during the search.

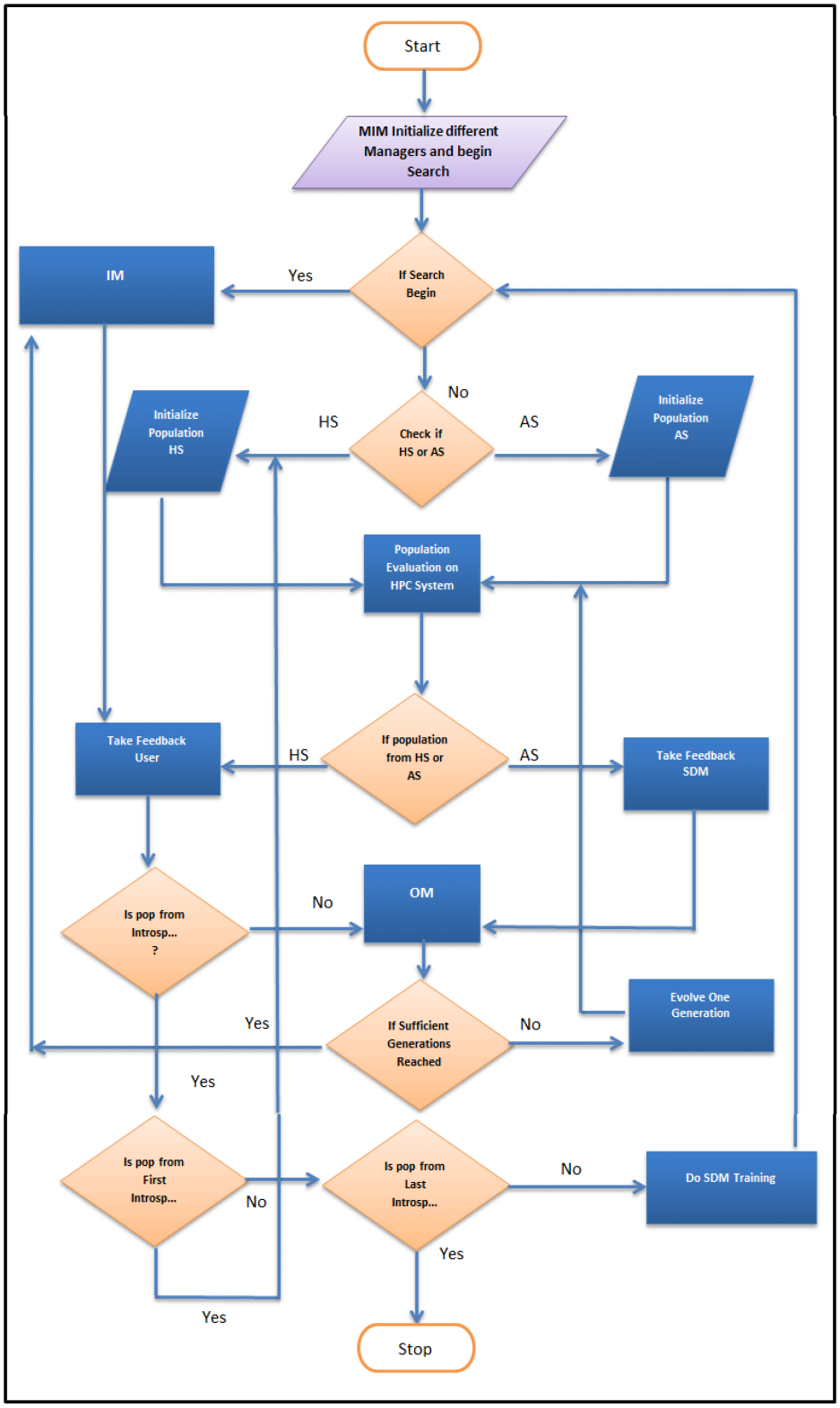

Figure 3.4. Flow Chart working of IGAMI2 Search and Learning

3.7.2 Working of MIM

The Working of MIM is given in Fig. 3.4. IGAMI2 Kernel start the search for the user by creating a User Search that does the initialization of different managers and call MIM. MIM initialize different search parameters, managers and start the search.

i) The search begins with the First Introspection that is done using the IM (Introspection Manager).

ii) The IM sends the data for the User's Feedback to the IDM (Individual Design Manager).

iii) If the feedback data came from Introspection, then it is checked to be from First Introspection, in which case the HS (Human Search) is performed.

iv) The NSGA2 Search begins with small population size.

v) The Population Evaluator sends the population for Evaluation to the HPC System.

vi) The feedback on population is taken from the Human User.

vii) If the feedback is not from the Introspection, the data is given to the OM for performing the optimization.

viii) If sufficient number of generations has not reached, another generation is being evolved.

ix) The population for the new generation is sent for the Evaluation.

x) The steps vi) to ix) are repeated until sufficient number of generations is reached.

xi) If sufficient number of generations reached, then second introspection session is performed.

xii) After the second introspection, a SDM (Simulated Decision Maker) is trained.

xiii) Based on the performance of SDM and learning confidence of the Human, next initiative is decided if it will be a HS or an AS.

xiv) If the next initiative is a HS then steps iv) to xiii) are repeated.

xv) If the next initiative is an AS then NSGA2 Search begins with a very large population size (say 100) and goes for a very large number of generations (say 100).

xvi) The population is sent for the Evaluation to the HPC System.

xvii) The SDM gives the rating instead of the Human User.

xviii) The data is given to the OM for performing the optimization.

xix) If sufficient number of generations has not reached, another generation is being evolved.

xx) The population for the new generation is sent for the Evaluation.

xxi) The steps xvii) to xx) are repeated until sufficient number of generations is reached.

xxii) If sufficient number of generations reached, then the last introspection session is performed.

xxiii) The search finishes after the last introspection.

3.8 The Distributed System

The distributed system contains a number of computers performing different roles to act as a single entity. The HPC part of the distributed system is used to run a number of SWAT evaluation in parallel on various supercomputers/clusters or cloud clusters. The infrastructure diagram of the HPC System is shown in Fig. 3.5. In the HPC System we are using IU's Future Grid's Windows HPC shared Cluster, a

dedicated cluster at Oregon State University and one small local cluster at IUPUI. We are using IU's Big Red II Supercomputer to run data analysis and create User Models.

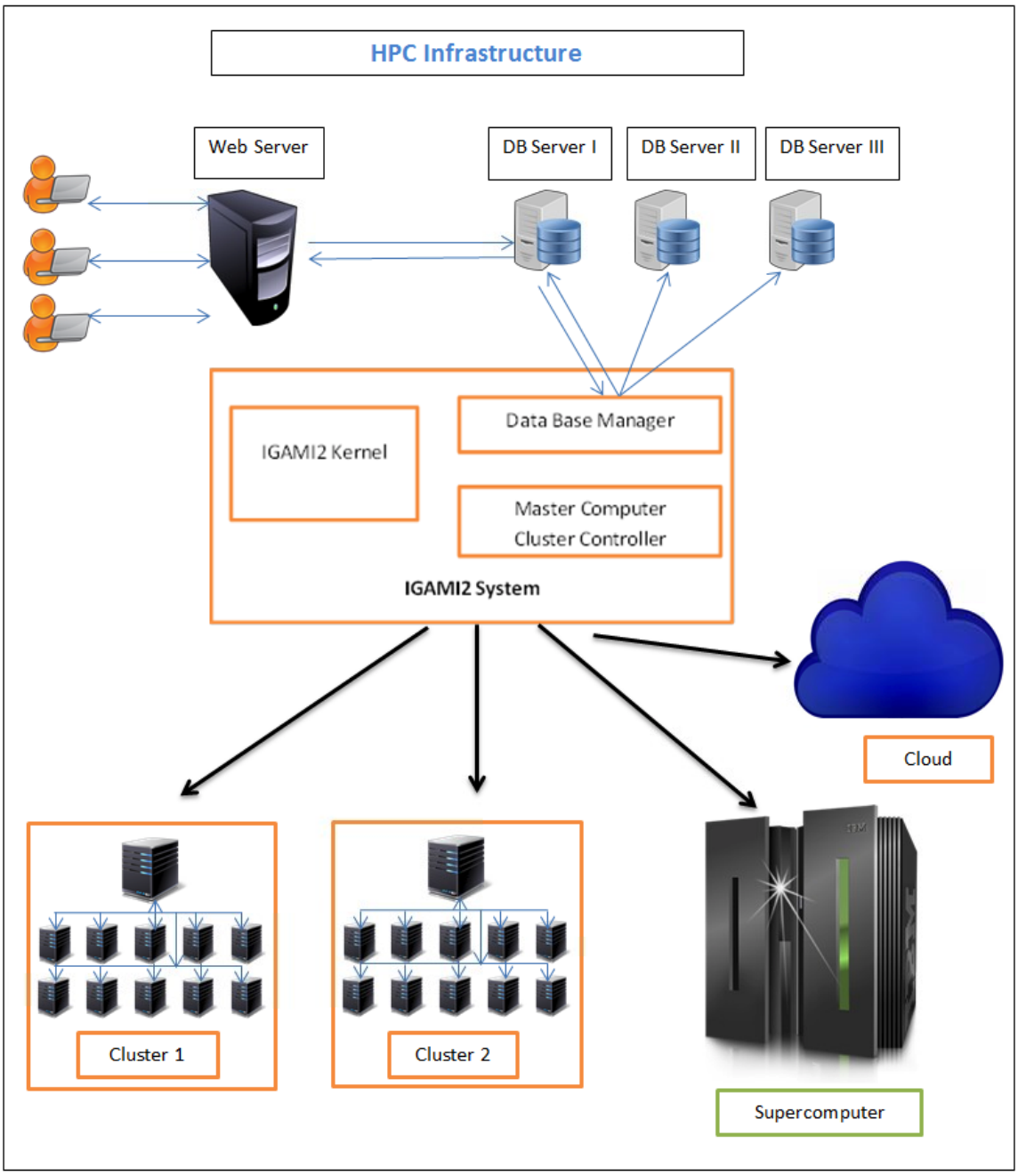

Figure 3.5. HPC System Infrastructure

3.8.1 Components of the Distributed System

Different components of the HPC System are as follows:

a. HPC Controller: It runs on the Master Computer of the Distributed System.

b. Head Node Cluster Controller: It is run on the Head Node of every cluster. It manages all VA and schedule jobs to different nodes. The Head Controller needs to run Head Node Cluster Controller program, which registers all the virtual agents and distribute the jobs to various VA efficiently.

c. Virtual Agents (VA): Each Node runs a Virtual Agent, which runs different Model simulations. Each Virtual Agent has the capacity to run a number of independent SWAT [15] (Soil and Water Assessment Tool) evaluation. All the virtual agents run Agent Main program which is used to register the VA to the Head Node Cluster Controller of the cluster, receives the jobs and send the results back.

3.8.2 HPC Controller

It is part of the IGMAI2 System.

Roles of HPC Controller

a) Allocate and manage resources of different clusters/supercomputers to different users.

b) Keep track of number of free nodes of clusters.

c) Runs a Job Scheduler to submit jobs to different clusters.

d) Distribute the job to different clusters based on the size of job and availability of the nodes.

e) If none of the clusters are free then the job is put in the waiting queue.

f) If any node is free, the job is allocated to that node.

Working of HPC Controller

a) Every user has a population Evaluator handle that they can use to run SWAT evaluation for a set of designs.

b) The population Evaluator put the job to job waiting queue of the HPC Controller and call the job scheduler.

c) The Job Scheduler check if any cluster has a free node. It send the job to that cluster if it has some free nodes.

d) If there is no free node, then the job will be kept in the job-waiting queue.

e) Whenever a node is freed, any waiting job is sent to that node.

f) The results of population evaluations are being sent to respective users.

3.8.3 Head Node Cluster Controller

It runs on the head node of every cluster.

Roles of Head Node Cluster Controller

a) Allocate and manage different nodes called as VA (Virtual Agents) of the clusters.

b) Runs a job scheduler to allocate jobs to different nodes.

c) Distribute the job to different nodes based on the size of the job and availability of nodes.

d) Two different types of job waiting queue, single node waiting queue and multiple node waiting queue.

e) A multiple node waiting queue is used if more than one VA are needed to complete a job.

f) A single node waiting queue is given priority over multiple node waiting queue to prevent starvation of small job.

Working of Head Node Cluster Controller

a) When the HPC Controller gives the job to the Head Node Cluster Controller, it puts the job in the single node waiting queue or multiple node waiting queue depending upon the job size.

b) The job scheduler is called which allocates a particular VA for the evaluation of the job.

c) If sufficient VA is not available then job is kept in the waiting queue.

d) Whenever a VA is freed, any waiting job is sent to that VA.

e) The results of the population evaluation are being sent to the HPC Controller.

3.8.4 Virtual Agent

It runs on every node of a cluster. It is a Virtual Agent that performs various environmental simulations in parallel.

Roles of Virtual Agent

a) Inform the availability of VA to the head node.

b) Perform a number of SWAT evaluation in parallel.

c) Initialize the SWAT IO Files to begin fresh evaluations.

d) Copy a new SWAT software if needed.

e) Cleanup the SWAT IO Files to save disk memory.

Working of Virtual Agent

a) VA waits for a new job from the Head Node Cluster Controller.

b) Once it receives a new job, it creates a number of Parallel threads to run each model simulations.

c) It performs the SWAT evaluation in parallel.

d) Send the result back to the Head Node Cluster Controller.

e) If the Head Node Cluster Controller ask to initialize, copy or delete SWAT IO Files, it perform respective actions.

Details about various parts of the HPC System (clusters/Supercomputer) used for our experiments are shown in tables Table. 3.1, Table. 3.2, Table. 3.3 and Table. 3.4.

3.8.5 HPC Infrastructure

The HPC System is capable of running around 1016 parallel independent SWAT evaluation on all the available clusters. Based on our current capacity of compute nodes around 40 users can log-in and run their search simultaneously. If more number of user will come, they will be either queued or more computing resources will be needed to fulfill the demand.

a. Amazon EC2 (Elastic Compute Cloud) is one of the popular cloud computing system that can be used to meet real time demands of high computing power.

Table 3.1

TEMPEST Cluster

Name	TEMPEST Cluster
CPU	Intel Xeon E7450
Cores per Node	24 (4 Processors)
Clock Speed	2.4 GHz
Memory per Node	48 GB
Memory/core	2 GB
Nodes (total)	32
CPUs (total)	768
Operating System	Windows HPC 2008R2
SWAT (no. of evaluations)	768

Table 3.2

OSU Cluster

Name	OSU Cluster
CPU	Intel Xeon E5-2690
Cores per Node	32 (2 Processors)
Clock Speed	2.9 GHz
Memory per Node	96 GB
Memory/core	3 GB
Nodes (total)	7
CPUs (total)	224
Operating System	Windows Server 2008 R2
SWAT (no. of evaluations)	224

Table 3.3

ESAIG Cluster

Name	ESAIG Cluster
CPU	Intel Core i7 870
Cores per Node	4
Clock Speed	2.93 GHz
Memory per Node	16 GB
Memory/core	4 GB
Nodes (total)	6
CPUs (total)	24
Operating System	Windows Server 2008 R2
SWAT (no. of evaluations)	24

Table 3.4

Big Red II Supercomputer

Name	Big Red II Supercomputer
CPU	AMD Opteron (16 core)
Cores per Node	32
Clock Speed	2.93 GHz
Memory per Node	64 GB
Memory/core	2 GB
Nodes (total)	1,020
CPUs (total)	21,824
Operating System	Cray Linux (SUSE Linux)
SWAT (no. of evaluations)	NA

The IGAMI2 system is connected to Amazon Web Services EC2 Cloud service. If more number of nodes is needed to perform large number of simulations, a cloud based cluster can also be connected to run the SWAT evaluation. AWS (Amazon Web Service) provides three types of pricing options for Amazon EC2 viz On-Demand Instances, Reserved Instances and Spot Instances. In preliminary test I used the Spot Instance for c3.4xlarge instance that is Compute Optimized instance which has capability to run sixteen SWAT evaluation in parallel. The advantage of using AWS EC2 is if we want to run ten thousand independent SWAT evaluations in parallel for a short duration then Amazon EC2 can be used. But till now our problem has not scaled to that kind of requirement yet, but it maybe used in future.

b. Big Red II Supercomputer can also be used to run a number of SWAT evaluation but our current SWAT software can not run on it as the version is not compatible. A newer version of SWAT software can be used on Big Red II. One challenge on Supercomputer is, as it is a shared system, a real time evaluation of SWAT is not feasible. But it can be used to run automated search or SWAT evaluations for research.

The distributed system is based on JAVA RMI (Remote Method Invocation) in asynchronous mode, as shown in Fig. 3.6. The requests are being transferred from one component to another in an asynchronous manner. The system is a multi-user system, just like a multi-user operating system. A number of users can log-in at the same time and run their SWAT evaluations, independent of knowing whose model runs where.

3.9 Enhancements from the Older System

Some of the enhancements from the old IGAMII system are as follows:

a. Emailing System: The system sends emails to the users when their designs are ready for feedback. It helps in solving the human fatigue problem.

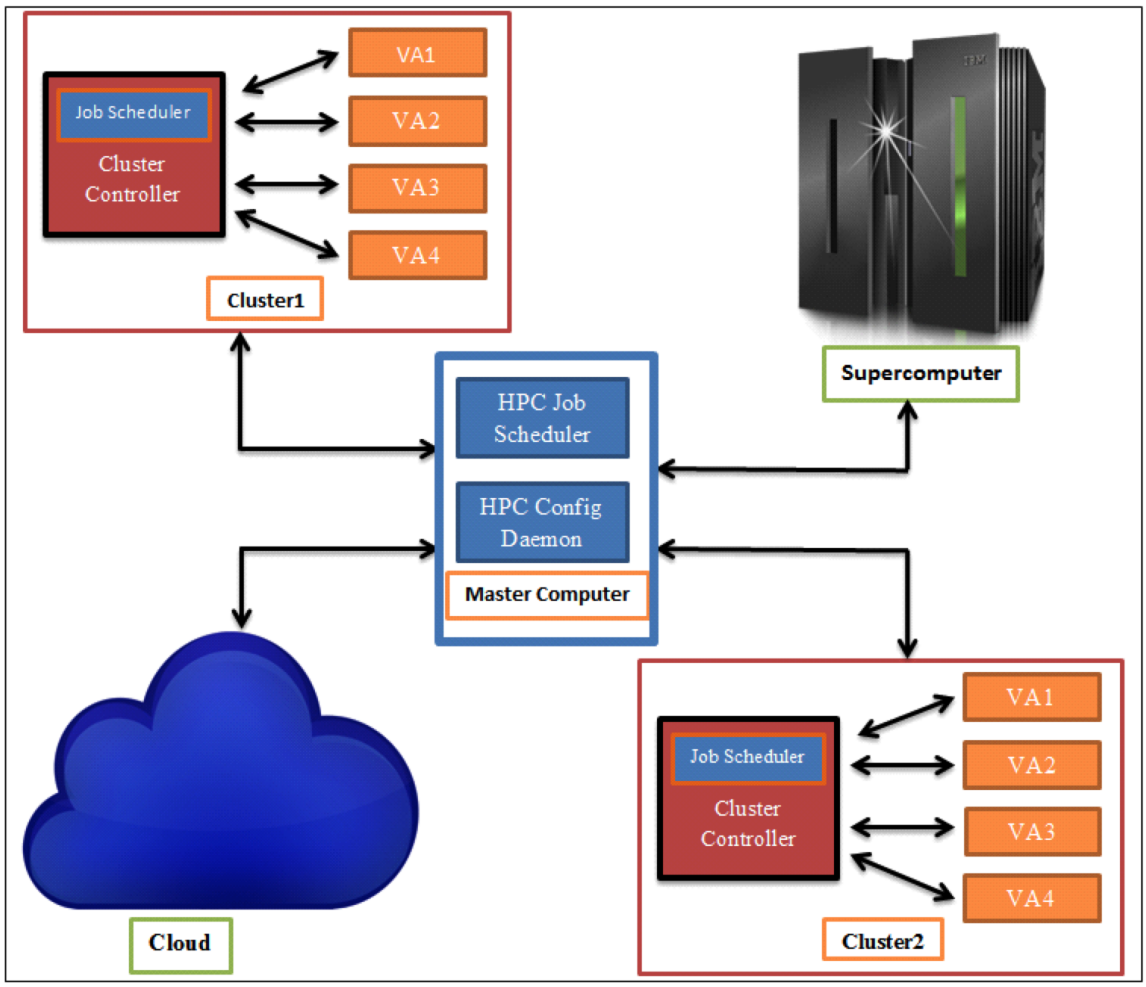

Figure 3.6. HPC System Working

b. Multi-user system: The system is a multi-user system. It provides simultaneous use of the system by many users, which is useful in a multi-user collaborative search.

c. Running the simulations on powerful computers: Each individual design evaluation takes a lot of time to evaluate on a normal computer. To run many simulations in parallel, we created a distributed system that connects the system with powerful supercomputers/clusters, which can be a heterogeneous system. This helps in reducing the overall waiting time.

d. Data management using Data Base: A lot of data is being generated in each search, so a DBMS is good tool to manage data for analysis. We used the open source MySQL as the DBMS server.

e. Different types of SDM: To create the simulated decision maker, different types of machine learning techniques can be used like ANN, Fuzzy inference system, SVM, Bayesian Network, etc. The machine learning algorithm that performs the best is chosen to mimic the human preferences and helps in the collaborative search. Currently we are focusing on ANN and comparing its performance with Fuzzy and other linear/non-linear modeling techniques.

f. Different types of Optimization Algorithms: For the optimization many different kinds of optimization algorithms can be used like NSGA2 [40], Decentralized Pursuit Learning Automata, PSO, etc. Currently we are using the distributed version of NSGA2 called Distributed NSGA2 to do the optimization [10].

3.10 Factors Affecting the Experiments

Some of the factors that affects the experiments:

a. Human Fatigue: A human can work for a limited amount of time and can not work on a large set of data. Rating merely 20 designs at a time is a cumbersome

work. Also, there is a maximum limit, the number of times a human can perform such feedback.

b. Long waiting time: Each feedback session takes around 13 minutes. The advantage is, because of long waiting time; the human has time to relax for the next session. But because of human fatigue, there is a limitation to number of such feedback sessions that can be performed in any experiment.

c. Compute Intensive Simulations: In this project, each model takes around 10 to 13 minutes to run each simulation. With the help of distributed system and HPC System we achieved evaluating many simulations at the same time.

d. Resource allocations: Managing multi-users has its own set of challenges. One of them is the allocation of compute nodes. To solve this problem we created client server type hierarchical system described above in the HPC System.

e. Limited learning Data: Because of human fatigue, we can perform the feedback sessions only a limited number of times, so data generated for learning is limited. To solve this problem we used other machine learning techniques to generate more data from this limited data set.

f. Correct Machine Learning Algorithm: Because of the limited amount of learning data, it is difficult to determine which machine learning algorithms will perform better. Also, human personal biases and randomness in rating are other factors that affect in deciding the correct learning algorithm.

4 METHODOLOGY

The IGAMI2 System allows users to test multiple solutions (alternatives) for locating and designing conservation practices in a simulated environment of their watershed landscape. Based on the overall performance of the practices in the simulated environment, users can then identify alternatives most suitable to their needs. The interactive framework takes feedback from the users and then uses an iterative search and learning method to search for better potential solutions that incorporate the user's feedback. For example, a farmer concerned with the problem of erosion on land can explore multiple types of best management practices and locations where the practices can be implemented on landscape. The following components are derived from our project website.

4.1 Eagle Creek Watershed

Eagle Creek Watershed is located North West of Indianapolis, as shown in Fig. 1.1.

The IGAMI2 is used to simulate seven different Best Management Practices(BMP) (described in later section) and four optimization functions on these practices.

The seven BMPs are:

- Wetlands
- Filter Stripes
- Grassed Waterways
- Crop Rotation
- No Till

Photo credit: *wrestore.iupui.edu*

Figure 4.1. Outline of Eagle Creek.

The outline of Eagle Creek Watershed is shown in Fig. 4.1.

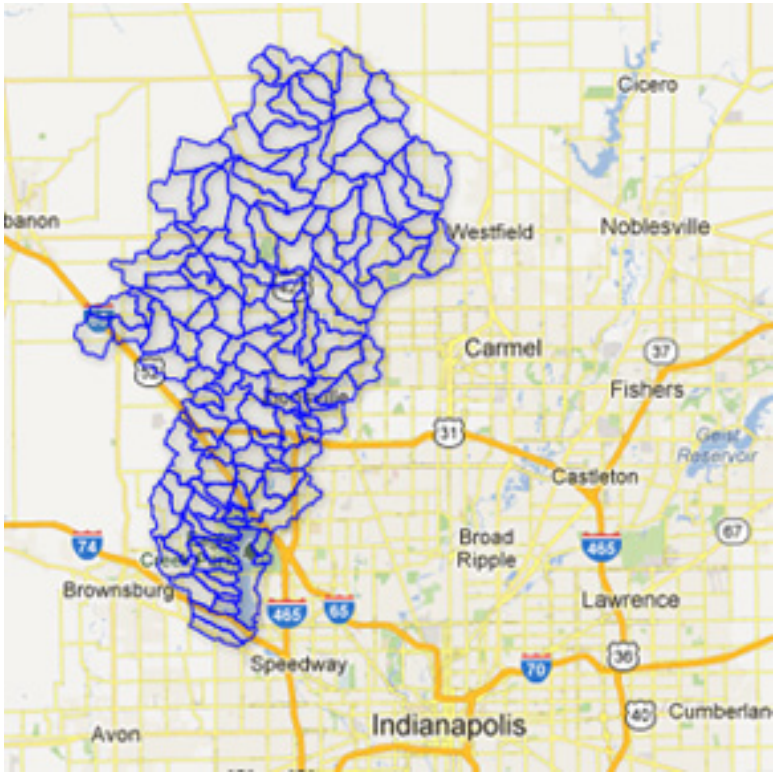

Photo credit: *wrestore.iupui.edu*

Figure 4.2. Subbasins of Eagle Creek Watershed.

The Eagle Creek Watershed is divided into 130 subbasins as shown in Fig. 4.1. Sub basins are smaller set of water bodies.

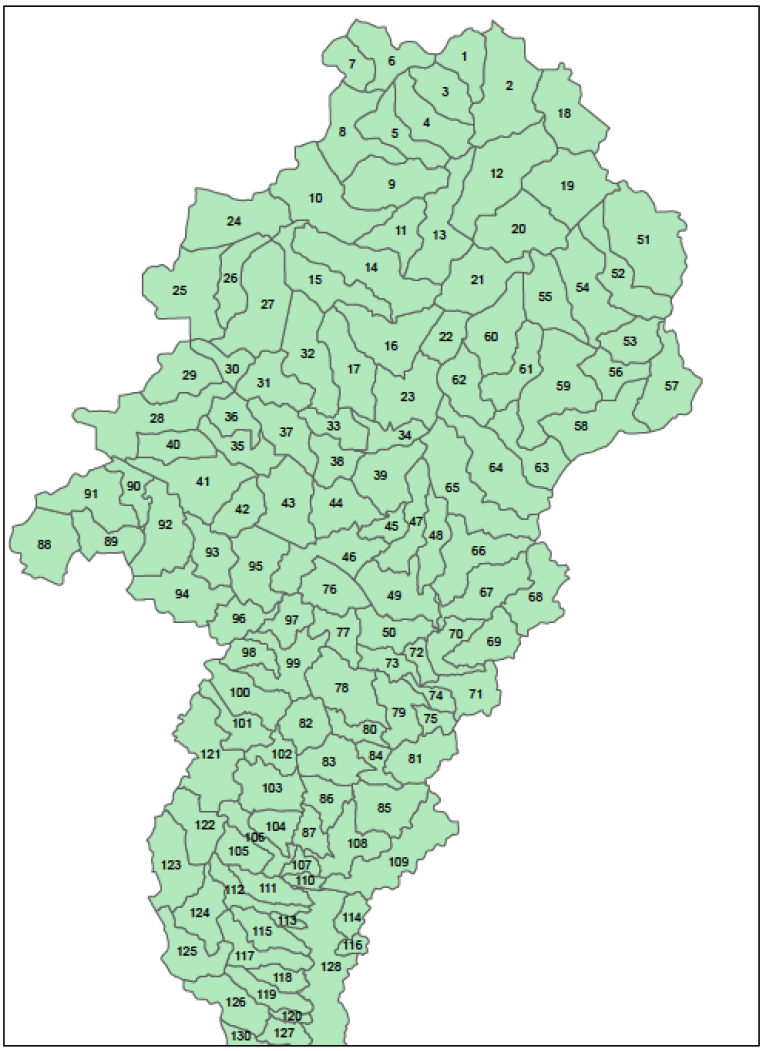

Photo credit: *wrestore.iupui.edu*

Figure 4.3. Subbasin IDs of different subbasins

Fig.4.1 shows subbasin IDs given to different subbasins in the Eagle Creek Watershed. These IDs are used by the system to generate different solutions and perform optimization.

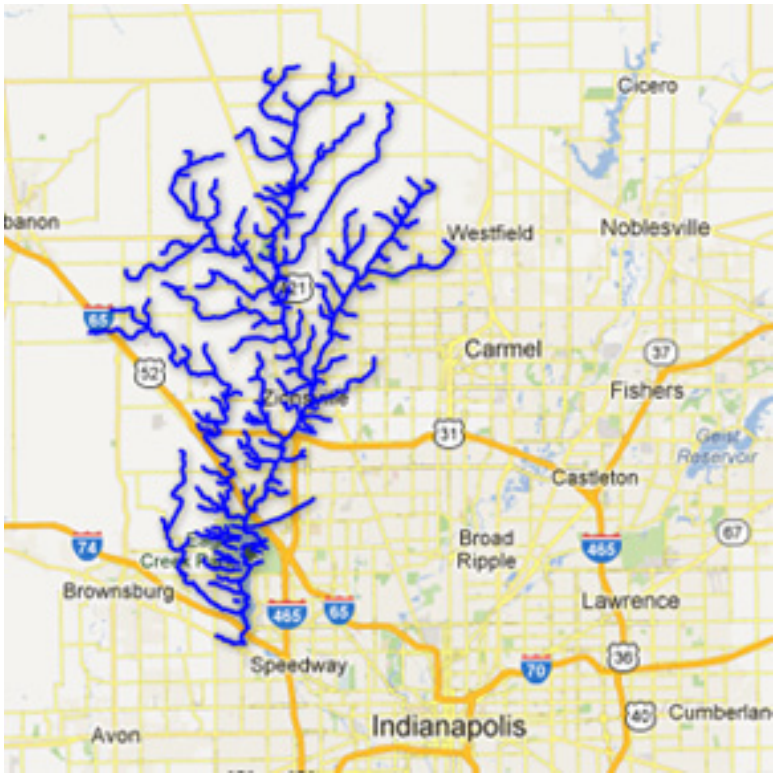

Photo credit: *wrestore.iupui.edu*

Figure 4.4. Streams

Various streams of water flow in Eagle Creek are shown in Fig. 4.1.

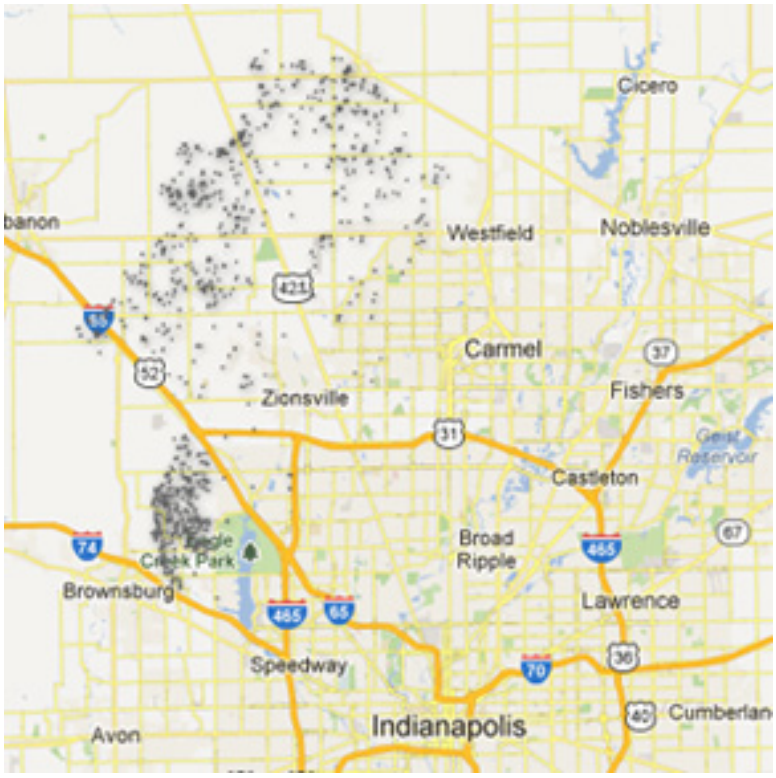

Photo credit: *wrestore.iupui.edu*

Figure 4.5. Potential Wetland Storage

Potential storage locations in Eagle Creek are shown in Fig. 4.1. These storage locations are water storage bodies like wetlands, ponds, etc. Which can be used to store excess water. These water storage locations helps in preventing soil erosion and fertilizer loss. It also increases the availability of water for usage during dry seasons and prevents drought situation.

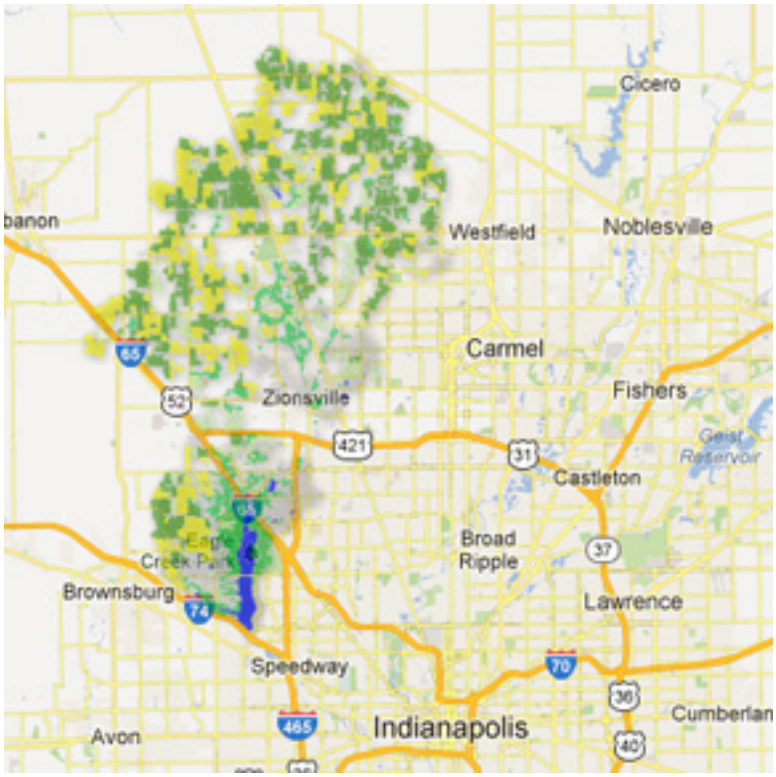

Photo credit: *wrestore.iupui.edu*

Figure 4.6. Land use of Eagle Creek

Land use land cover is shown in Fig. 4.1.

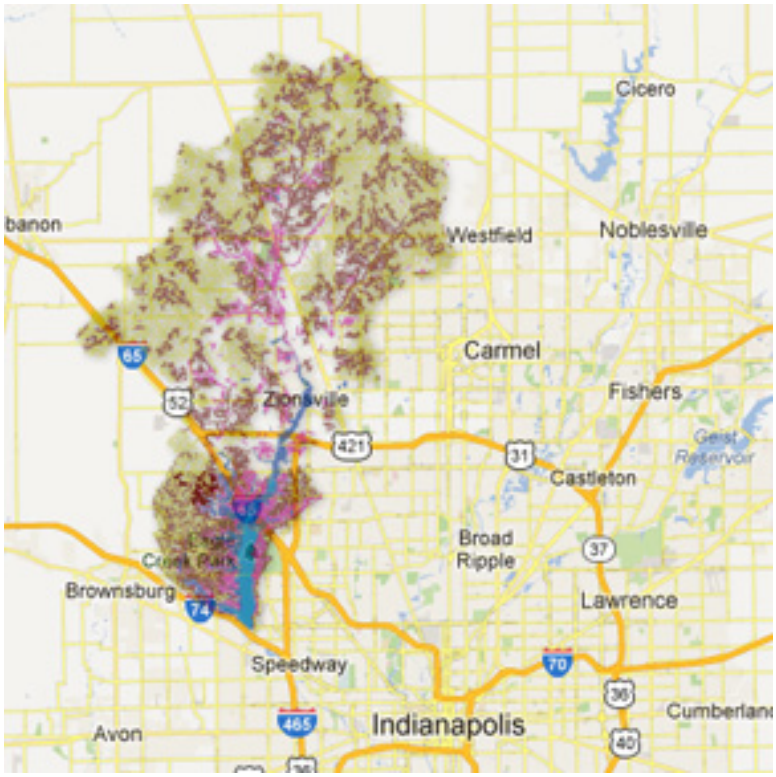

Photo credit: *wrestore.iupui.edu*

Figure 4.7. Soil Information of Eagle Creek

Soil types and their draining characteristics are shown in Fig. 4.1.

- Strip Cropping
- Cover Crops

4.2 Fitness Functions for Optimization

The different fitness functions for optimization are as follows:

A. Peak Flow Reduction (f0): f0 is the normalized value of Peak Flow Reduction Function. It is used to show decrease in the flooding of the region.

B. Economic Cost (f1): f1 is the normalized value of Economic Cost Function. It is used to show decrease in the total economic cost for different farming practices being used.

C. Soil Erosion Reduction (f2): f2 is the normalized value of Nitrate Reduction Function. It is used to show decrease in the total soil erosion of the land.

D. Nitrate Reduction (f3): f3 is the normalized value of Soil Erosion Reduction Function. It is used to show decrease in the total fertilizer loss.

E. Human Rating (f4): A human rating is given by the real human or SDM.

4.3 Individual Design Solution

An individual design solution is one of the solution in the search space. An individual design solution is defined as a string of values assigned to different subbasins for a BMP or set of BMPs. The subbasins and their IDs are shown in Fig. 4.1. Although there are 130 subbasins but in our research we have used only 108 subbasins, because in certain subbasins many of the practices can not be implemented. So in our research an individual design solution is a string of values for 108 subbasins for every BMPs.

For every BMP a subbasin can have values:

- Binary Values: 1 (practice is implemented) or 0 (practice is not implemented). So there are total 2^{108} possible individual design solutions.
- Real Number Values (R): A number can be between $[-\infty, +\infty]$. So there are R^{108} possible individual design solutions. Although the actual number of solutions are much lesser depending upon number of possible assignments a practice can take.

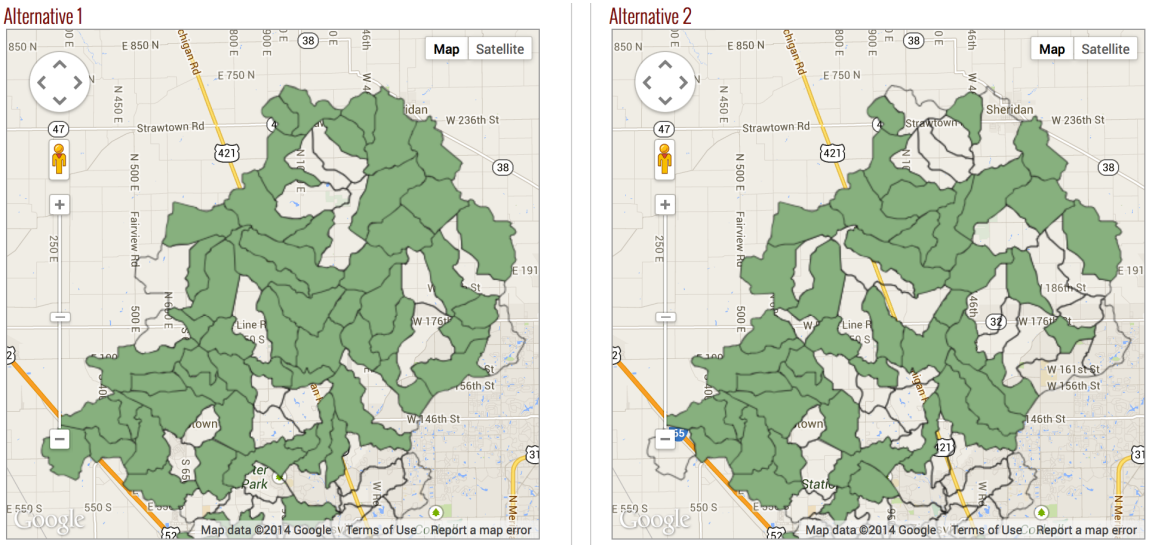

Figure 4.8. Alternatives for Strip Cropping

Individual Design Solution alternatives for Strip Cropping.

e.g., for Strip Cropping a subbasin can have values 0 or 1. As shown in Fig. 4.8 as we can see that for different alternatives individual design solutions some subbasins are colored i.e., value 1 (Strip Cropping is implemented) or not colored i.e. value 0 (Strip Cropping is not implemented).

For every individual design solution a SWAT evaluation is performed and fitness function values are computed at both subbasin level and at the entire watershed level. These fitness function values viz. peak flow reduction, economic cost, soil

erosion reduction and nitrate reduction, can then be used for optimization or for analysis by the user using the visualization interface.

4.4 SWAT

SWAT stands for Soil and Water Assessment Tool [3]. SWAT is software developed by USDA Agricultural Research Service(USDA-ARS) and Texas A&M AgriLife Research to simulate the quality and quantity of surface and ground water and predict the environmental impact of land use, land management practices, and climate change for a particular watershed. It is used for soil erosion prevention and control, non-point source pollution control and regional management in watersheds.

4.5 User Interface

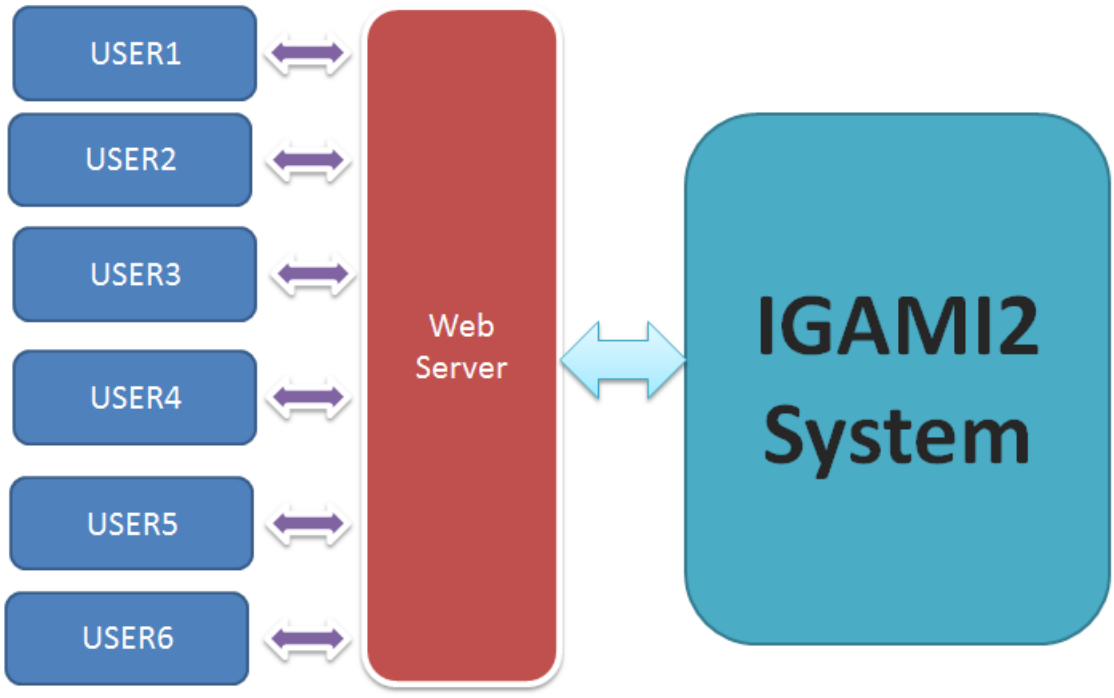

Figure 4.9. Users Working simultaneously on the System

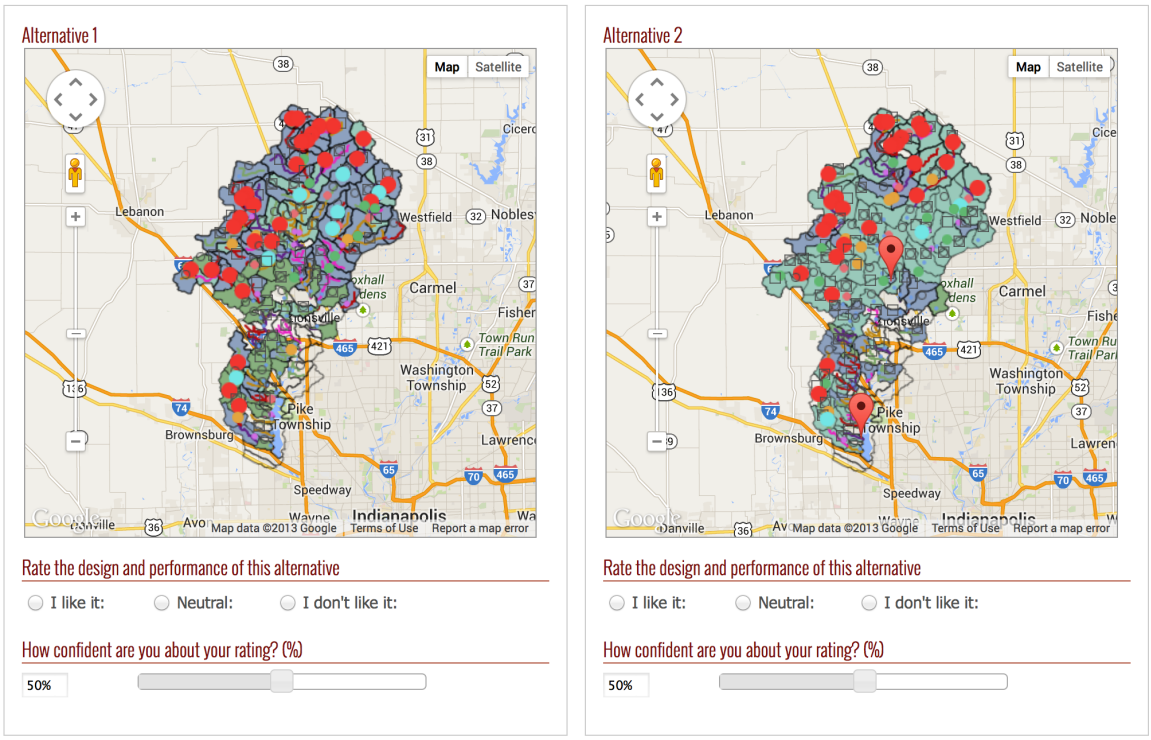

Figure 4.10. Visualization of different BMP

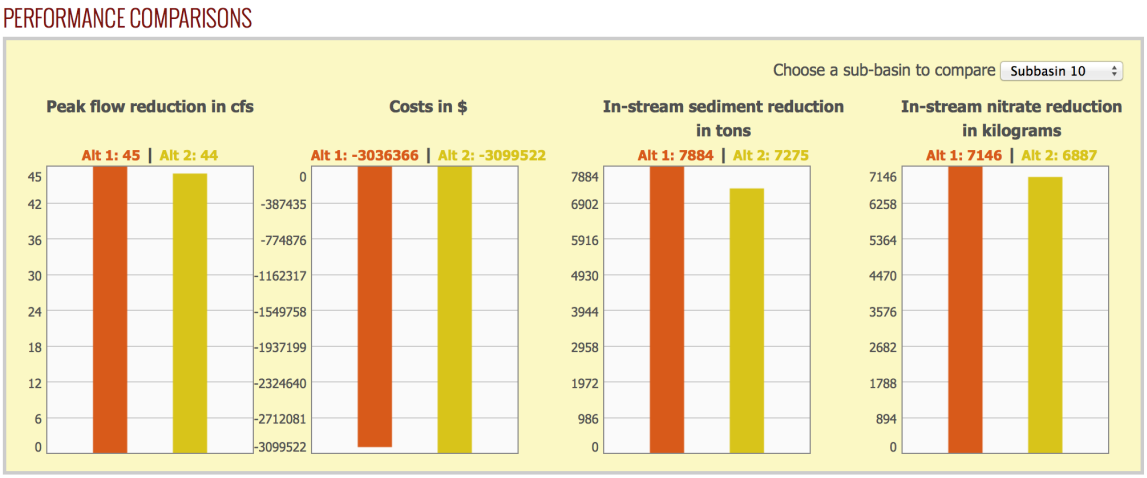

Figure 4.11. Visualization for various Subbasins

The working of User Interface is shown in Fig. 4.9. A number of Users log-in to the system via Web Server. The Web Server connect to the IGAMI2 System and pulls the data for respective users and show that to each user.

The Visualization is provided at two different levels, one at the entire Watershed Level and another at the Subbasin Level. The Filed Scale Level [41] can also be shown in later version of our system. Fig. 4.10 shows the values for different BMP values in color coded format for various subbasins.

Fig. 4.11 shows the values of different fitness functions used for optimization viz. Peak Flow Reduction, Economic Cost, Sediment Reduction and Nitrate Reduction. These values can be visualized for entire watershed as well as for particular subbasins.

Based on above visualizations the users provide their feedback by giving rating 1 (bad), 2 (Neutral) or 3(Good). These ratings are used by the NSGA2 to run genetic search and involve the user's feedback as an additional criteria for optimization.

The optimization is performed for seven BMPs and the five fitness functions. For every BMP or a pair of BMPs the values of the fitness functions are calculated. The values are calculated for each subbasin and the entire watershed. The users then visualize these using the visualization interface as shown in the Fig. 4.10 and Fig. 4.11. The users make fuzzy decision about rating the designs, give their feedback and specify how much confident they are about their ratings.

SDM Modeling: Based on the feedback of the user, a machine learning algorithm is used to mimic the preferences made by the user. In which these fitness functions data for the watershed or the subbasins are mapped to the user's feedback data.

4.6 Different Best Management Practices

Different Best Management Practices are described in the following sections, the contents are derived from the Wrestore Project Website: `http://wrestore.iupui.edu/resources/best-management-practices/` (accessed Jan, 21, 2014).

4.6.1 Wetlands

Photo credit: *USDA-NRCS*

Figure 4.12. Wetlands

Wetlands are unique ecosystems that often occur at the edge of aquatic (water, fresh to salty) or terrestrial (upland) systems. They may be wet year-round, wet during certain seasons, or wet during part of the day. Wetlands generally include swamps, marshes, bogs, and similar areas.

Benefits of Wetlands:

- Flood damage reduction: Wetlands can be used to reduce the flood damage of the region.
- Erosion control: By dissipating wave energy and stabilizing shorelines, wetland vegetation buffers the adjacent upland from wave action and intensive erosion.

- Water quality: Wetlands play a major role in maintaining water quality. Wetlands absorb excess inorganic and organic nutrients such as farm fertilizers and septic system runoff, filter sediments such as eroded soil particles, and trap pollutants such as pesticides and some heavy metals. These materials can seriously degrade the quality of groundwater and surface water resources, but wetlands trap and hold them, recycling some of them within the wetland system.
- Aesthetics and recreation: Wetlands can be used for aesthetics and recreation activities.

4.6.2 Filter Strips

Photo credit: *USDA-NRCS*

Figure 4.13. Filter Strips

A filter strip is an area of vegetation, generally located at the lower edge of a field, established for the purpose of removing sediment, organic material, and other pollutants from runoff and waste water. Filter strips remove pollutants from runoff before the material enters a body of water. They also serve as a buffer between water and the fields above the water so that pesticides and other chemicals are not applied directly adjacent or into the water body.

Benefits of Filter Strips:

- Decrease maintenance.
- Enhance wildlife habitat.
- Provide additional income.
- Improve water quality.
- Increase aesthetic value.
- Reduce stream bank cutting.
- Provide year long access.
- Increase safety.

4.6.3 Grassed Waterways

A grassed waterway is a natural or constructed channel that is shaped or graded to required dimensions and established with suitable vegetation. Grassed waterways are used to convey runoff from concentrated flow without causing soil erosion, to control gully erosion, and/or to protect and improve water quality. Depending on the type of vegetation established, grassed waterways may also provide wildlife habitat for a variety of farmland wildlife such as quail, pheasants, and rabbits. The width of the grassed waterway depends upon several factors including the slope of the field, the soil type, the drainage area, and the conservation practices used in the field.

Photo credit: *USDA-NRCS*

Figure 4.14. Grassed Waterways

Benefits of Grassed Waterways:

- Excess surface water from natural drainage areas or constructed structures (e.g., terraces, diversions, etc.) can be safely disposed.
- Gully formation is inhibited by vigorous, dense vegetation.
- The vegetation also reduces pesticides and other soluble nutrients in surface water because of improved infiltration.
- Decrease in sediments and chemicals in surface water increases availability of dissolved oxygen for growth of aquatic life.

4.6.4 Crop Rotation

Crop Rotation involves growing various crops on the same field in a planned sequence. The rotation usually involves growing forage crops in rotation with various field crops. Crop rotation also means that succeeding crops are of a different genus, species, subspecies, or variety than the previous crop. Examples would be barley after wheat, row crops after small grains, grain crops after legumes, etc.

Benefits of Crop Rotation:

- Reduced runoff and erosion.
- Increased organic matter.
- Improved soil tilth.
- Reduced pests.
- Fewer chemicals needed.
- Better moisture efficiency.
- Higher yields.
- Improved aesthetics and wildlife habitat.

Photo credit: *USDA-NRCS*

Figure 4.15. Crop Rotation

4.6.5 No Till

Photo credit: *USDA-NRCS*

Figure 4.16. No Till

No-till is a conservation practice that leaves the crop residue undisturbed from harvest through planting except for narrow strips that cause minimal soil disturbance. Crop residues are materials left in an agricultural field after the crop has been harvested. These residues include stalks and stubble (stems), leaves and seed pods.

Benefits of No-Till:

- Increased earthworm populations that improve soil quality.
- Increased water in filtration.
- Reduced tilling time per acre by as much as two-thirds.

- Improved wildlife habitat.
- Optimized soil moisture, thereby, enhancing crop growth in dry periods.

4.6.6 Strip Cropping

Photo credit: *USDA-NRCS*

Figure 4.17. Strip Cropping

Strip cropping is growing crops in a systematic arrangement of strips across the field to reduce soil erosion by water and/or wind. This practice is used on cropland and certain recreation and wildlife lands where field crops are grown. The crops are arranged so that a strip of grass or close-growing crop is alternated with a clean tilled strip or a strip with less protective cover. Generally the strip widths are equal across the field. On sloping land where sheet and rill erosion are a concern, the strips are

laid out on the contour or across the general slope. Where wind erosion is a concern, the strips are laid out as close to perpendicular as possible to the prevailing erosive wind direction.

Benefits of Strip Cropping:

- Reduced sheet and rill erosion.
- Reduced wind erosion.
- Increased infiltration and available soil moisture.
- Reduced dust emissions into the air.
- Improved water quality.
- Improved visual quality of the landscape.

4.6.7 Cover Crops

Cover crops are an excellent option for producers to consider for protecting their soil and increasing productive capacity for succeeding years. Cover crops are grown between regular crop rotations like corn, soybean and wheat. Examples of cover crops are annual rye-grass, crimson clover, oats, oil-seed radishes, and cereal rye. A key concept is to ensure that vegetation is green and growing during all times of the year. Cover crops are essential during summer months when primary crops are not feasible (such as when crops are damaged and it is too late for replanting) and during the fall/winter months following harvest. Cover crops are not intended as a harvest crop, but are grown to enhance productivity.

Benefits of Cover Crops:

- Improving soil structure by increasing soil organic matter and root penetration.
- Protecting otherwise bare soil from wind and water erosion.
- Using nitrogen left in the soil, thereby, preventing it from polluting waterways.

Photo credit: *USDA-NRCS*

Figure 4.18. Cover Crops

- Cycling nutrients back into the soil that will be available for corn and soybean crops.
- Increase biodiversity.
- Promote biological nitrogen fixation and reduce energy use.
- Suppress weeds.

5 OPTIMIZATION ALGORITHMS

One of the challenges that we faced is the fast optimization algorithm. There are a number of multi-objective optimization algorithms that can be used in our research.

From the work of [4] we created some faster optimization algorithms which are described in the sections below.

Learning Automaton has been used for a long time for solving problems [42] like identical pay-off games, etc. In this research we are using the Learning Automaton to solve the multi-objective problems as done in [4]. The Learning Automaton we used is Decentralized Pursuit Learning Gaming Algorithm. It is an indirect learning method.

The use of learning automaton has been applied in ground water monitoring problems in Environmental Science [43]. We have used NSGA2 [40] as the genetic algorithm, which is one of the most popular multi-objective GA available today.

We implemented multi-objective version in LA by assigning multiple weights to the different functions and evaluating the population over a period of time. It gives the result comparable to GA, we call it era, as each weight pair set is independent of other weight pairs. Since the objective functions and their weight pairs, i.e. eras, are independent, we can use PRAM (Parallel Random Access Machine) based algorithms to run the entire multi-objective eras in parallel. Based on the availability of computational power we can use smaller weight pairs, i.e. smaller slices of eras, for more precision and accuracy. GA is often criticized as slow in the cases where the evaluation of fitness function is very slow. Since the fitness function we are using in this work is very slow, we created distributed version of the NSGA2, called distributed NSGA2 based on the work of [44]. To make the work faster we implemented the parallel work at two levels, one at processor level and other at machine level.

5.1 Similar Work

Reinforcement Learning is used by [13] and [14] as a learning technique in LA. LA takes a lot of time to converge to good solutions, so multiple studies have investigated faster learning approaches via parallelization [13]. The parallelization techniques were done for common pay-off games, parameterized learning automata, and pattern classification problems. The motivation [13] was N independent Agents acting simultaneously should speed-up a process roughly by a factor of N, similar to a population as in GA. But this parallelization was tested for direct learning algorithm such as LRI Algorithm. In our work, the N LA's take M independent actions at a time. Combining all the automatons and their actions will create a set of actions to be performed by the system, this is like an individual in GA or one design of SWAT. In Batch Mode, we created multiple such individuals. As execution and reward for each individual is independent of one another, a PRAM based algorithm can be used to evaluate them independently. Batch mode learning is one of the popular learning techniques used in ANN. In batch mode [16], the learning of the ANN is done by taking the average of all training patterns before changing the weights. Choosing a very small learning parameter is not realistic as the smaller the learning, the longer it takes to converge. So based on availability of resources and time, learning parameter can be chosen.

Batch Mode learning is also done in text categorization [17], where batch of text documents are used instead of just one. Batch Mode learning is used in medical Image classifications [18] and Content based Image retrieval [19] where instead of selecting a single unlabeled example, a number of unlabeled examples were selected for manual labeling. A discriminative batch mode active learning is done by [20]. People also have tried combining the LA and GA [21] to escape the problem of local optima. In StGA [22], the authors created a small number of actions of the Learning Automata which were sampled to construct a population, from which the sampling action was done adaptively by genetic operations. Some authors have also tried to

combine the GA with other algorithms like Simulated Annealing to solve some NP-Hard Problems [23].

5.2 Theory

Our concept used here is similar to the one used by [4]. One of the objectives is to minimize the flooding of water by creating small watersheds throughout the Eagle creek by using minimum area. The total wetlands are divided into 108 aggregated wetlands. For a binary problem of choosing or not choosing a wetland, the total search space will be 2^{108} which is computationally not feasible, so it was divided into eight regions. Currently we ran the tests for region seven, Fig. 5.1 to test our algorithms.

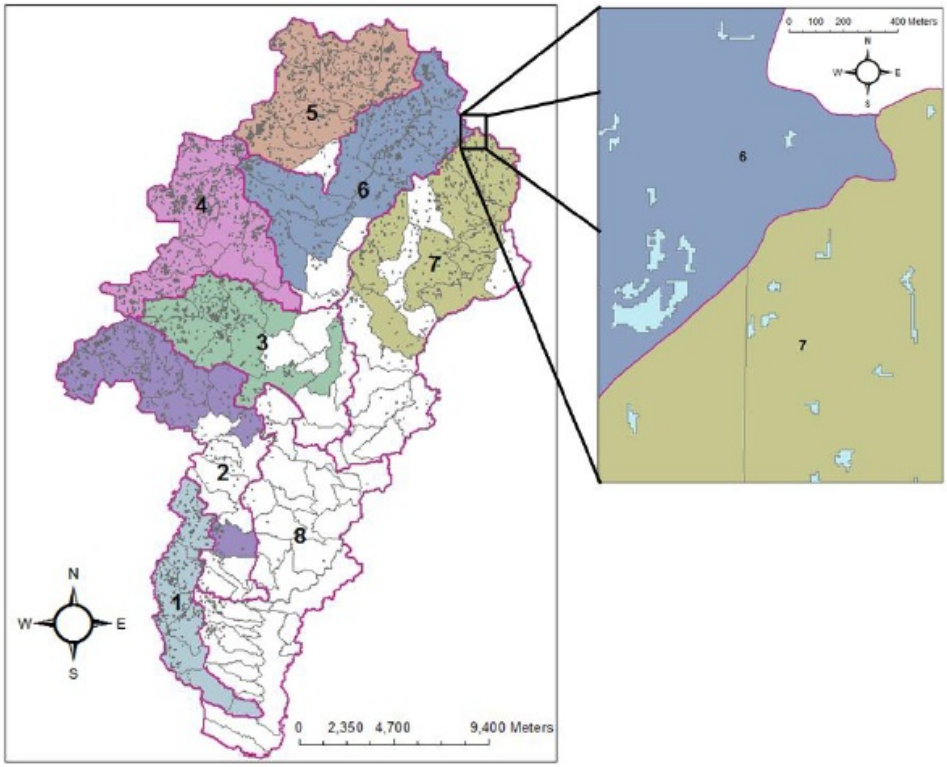

Photo credit: *Ref. Omkar [4]*

Figure 5.1. Sub-basins and Potential wetland polygons

The multiple objectives were to minimize the root mean-square error between flows in streams when all the regions were installed, with the minimum area. Learning automata are implemented by assigning one learning automaton for each sub-basin, which decides if it will participate or not in the system. It is similar to identical pay-off game model. Two scaling parameters, S^i_{Area} and S^i_{Flow}, for region i, to scale the values of area and flow. S^i_{Area} is the total area of all the aggregated wetlands as shown below.

$$S^i_{Area} = \sum_j Area^i_j \tag{5.1}$$

S^i_{Flow} is calculated based on baseline flow data set (Baseline) and running the SWAT software, and using the following equations:

$$S^i_{Flow} = \sum_{region} \ln(1 + [Baseline - Output_{noWetlands}]^2) \tag{5.2}$$

$$P^i_{Flow} = 1 - \sum_{region} \ln(1 + [Baseline - Output_{subsetOfWetlands}]^2) \tag{5.3}$$

$$P^i_{Area} = 1 - \sum_{region} (flag^i_j \times Area^i_j) \tag{5.4}$$

Where,

$$flag^i_j = \begin{cases} 1, & \text{if wetland j in region i is installed} \\ 0, & \text{if wetland j in region i is not installed} \end{cases}$$

The common pay-off is calculated using:

$$P^i_{Total} = W_{Area} \times P^i_{Area} + W_{Flow} \times P^i_{Flow} \tag{5.5}$$

In the DPLA algorithm only one individual design was created at random for each iteration. The design was created by randomly selecting the actions of the automaton based on their action probability vector. We call it individual design, as used in GA.

In the distributed version of the DPLA, called Dist DPLA, all the weight pairs were run separately as they were independent of each other. Applying the similar parallelization concept and the batch mode technique we created different distributed

versions of DPLA, we call it RLBM (Reinforcement Learning in Batch Mode) where a set of multiple individual designs were created. We can get more accuracy as more and more individual designs are being evaluated and learning is done from multiple individual designs. It is similar to taking opinions from many individuals instead of just one individual. The opinion can be taken in many different ways. We tried three different ways of taking the opinion and perform the learning.

For each weight pair, all LAs together create a set of designs, instead of just one as in the previous work. The multiple set of designs are similar to multiple individuals of the GA.

Now there are multiple ways we can use these set of designs to do the learning.

- i) Do the learning based on average of all the solutions.
- ii) Do multiple learning instead of just one time. (Converge faster, but compromise accuracy).
- iii) Do the learning based on the best solutions from all the individual designs (optimal weight pair).

In our current research, we tried type I, type II and type III, but other techniques can also be tried.

5.3 The Algorithms

We are using the DPLA algorithm as proposed by [4].

5.3.1 DPL Game Algorithm

The DPL game algorithm proceeds as follows:

1. At every time step, the i^{th} automaton chooses action α_i at instant k, based on the probability distribution $p^i(k)$.

2. Each automaton i obtains a common pay-off $\beta(k)$ based on the set of all actions.

3. Based on the pay-off, each automaton i updates its own R, Z and $\hat{D}$ matrices.

where, R, Z and $\hat{D}$ are vectors used for total reinforcement received, number of times a particular action is chosen and to model the environment and do the learning, respectively.

The action probability vector is updated as follows:

$p(k+1) = p(k) + \lambda(e_{M(k)} - p(k))$,

where,

- $0 < \lambda < 1$ is the learning parameter,
- $e_{M(k)}$ is a Unit Vector used to store information about various actions,
- An action has value of 1 if it correspond to maximum element of $\hat{D}$, otherwise its value is 0,
- index M(k) is determined by $\hat{D}_{M(k)} = max_j \hat{D}^i_j(k)$.

We created the distributed version of the algorithm, in the following three ways:

5.3.2 RLBM Type I

Reinforcement Learning in Batch Mode, Type I:

1. For each weight pair of the multi-objective, run in parallel.

2. Create a set of n individual designs at instant k, each individual design has

3. The i^{th} automata randomly chooses action α_i, based on the probability distribution $p^i(k)$.

4. Evaluate the individual designs in parallel.

5. Take the average of the all individual designs.

6. Each automaton i obtains a common pay-off $\beta(k)$ based on the set of all actions.

7. The actions which was chosen most is rewarded.

8. Based on the pay-off, each automaton i updates its own R, Z and $\hat{D}$ matrices.

The action probability vector is updated as the above algorithm. The number n must be an odd number to make sure that one action will be selected most, in our test we used n = 7, based on computational power we had during the research.

5.3.3 RLBM Type II

Reinforcement Learning in Batch Mode, Type II

1. For each weight pairs of the multi-objective, run in parallel

2. Create a set of n individual designs at instant k, each individual design has

3. The i^{th} automata randomly chooses action α_i, based on the probability distribution $p^i(k)$.

4. Evaluate the individual designs in parallel.

5. Use all the individuals to do the learning one by one randomly or sequentially.

6. Each automaton i obtains a common pay-off $\beta(k)$ based on the set of all actions.

7. Based on the pay-off, each automaton i updates its own R, Z and $\hat{D}$ matrices.

The action probability vector is updated as the above algorithm.

5.3.4 RLBM Type III

Reinforcement Learning in Batch Mode, Type III

1. For each weight pairs of the multi-objective, run in parallel

2. Create a set of n individual designs at instant k, each individual design has
3. The i^{th} automata randomly chooses action α_i, based on the probability distribution $p^i(k)$.
4. Evaluate the individual designs in parallel.
5. Use a set of best or average of best individuals to do the learning one by one randomly or sequentially.
6. Each automaton i obtains a common pay-off $\beta(k)$ based on the set of all actions.
7. Based on the pay-off, each automaton i updates its own R, Z and $\hat{D}$ matrices.

The action probability vector is updated as the above algorithm. In our current experiments we used the top three best individuals based on their pay-off.

Using Type I, we can get more accuracy as more and more individuals are being evaluated and learning is done from multiple individuals. It is similar to taking opinions from many individuals instead of just one individual. As we are taking the average, the final reward is given to that action of automaton which is selected the most. Each automaton is capable of performing only two actions [4]. Like in the voting system, the most selected action of design set is being rewarded. The process goes to multiple generations aka iterations, unless some convergence criteria are met. Also, the learning parameter and convergence criteria can be varied, to get more accuracy or speed.

Using Type II, we can get a faster convergence. Learning is performed from the opinion of multiple individuals, who are siblings, instead of just one. Instead of taking their average opinion, the automata are rewarded directly from all the individuals. It is similar to taking the opinion of everybody, considering all the opinions to be equally true. There will be some loss of accuracy as multiple learning steps are performed using the same action probability; instead of getting one feedback, the automata are getting multiple feedback. In this way the algorithm converges very fast. To compensate the loss of accuracy, we can decrease the step size of learning.

Using Type III, we can get a compromise of speed and accuracy. It is kind of hybrid of Type I and Type II. The opinion can be taken as average or all of the best. The selection of the best individual is a big task in this algorithm. In our current experiments we used the top three individuals, based on pay-off, but other techniques can also be used to find the best set of individuals. To perform the learning, the individuals can be selected randomly from all the individuals, generated for learning, or simply perform the learning in the order the individuals are generated, i.e sequentially.

5.4 Limitation of RLBM and DPLA

RLBM and DPLA both work on the principles of Reinforcement Learning, i.e., for the problems where number of actions are limited. The action probability vector can only be computed if the number of actions are discrete. But what if the number of actions are in a continuous domain? Using the RL as an optimization for a continuous domain is not feasible. In some cases the domain's data is highly stochastic and the actual action does not matter as long as it is close to some optimal point. In such a scenario the domain of action can be changed to discrete classes. An action can be defined as the value falling in that class. Then we can implement RLBM or DPLA optimization algorithm in such cases. Although actual implementation needs further investigation.

6 MACHINE LEARNING ALGORITHMS

Another Big challenge that we faced was finding a good machine learning algorithm in the domain of environmental planning. In this work we tried to formulate mechanisms that can map the environmental data to user's feedback data.

On the one hand in the domain of Environmental Science, the data is highly stochastic. On the other hand in the case of interactive optimization the number of data points, that can be gather, is limited due to human fatigue. Training an Artificial Neural Network might not always give the accurate results when we have limited data. There are many situations where a simple linear/non-linear modeling technique works better than ANN. In such cases we can fine tune the ANN by generating more data from the other models.

Deep Learning is also one of the prominent machine learning technique that can be used in our research. In Deep Learning, the leaning is done at multiple levels of abstraction. In our research we tried to formulate ways of implementing deep learning for environmental planning system design.

6.1 Challenges in Creating a SDM

1 High number of input parameters: The Number of input parameters can be very high depending upon the subbasins and BMP selected by the user.

2 Varying size of input parameters: The input parameter can vary from one user to another based on the selection of the subbasins.

3 Limited Feedback from the User: The user can not give a lot of feedback because of human fatigue; therefore, the data available to train the SDM is very limited.

4 Limited number of output parameters: The number of output parameters is only one with three possible options viz., rating 1, 2 or 3.

5 Skewed Training data: The SDM training data might be skewed to only a few ratings.

6.2 Varying Number of Parameters

In the Environmental Planning System domain the number of output parameter is only one which can take three values. The number of input parameters vary tremendously. The problem can be formulated as a three class classification problem with varying number of input dimensions.

- Total Number of Fitness functions can vary from 1 to 4.
- Total Number of BMPS can vary from 1 to 7.
- Total Number of Subbasins can vary from 0 to 108.

The number of input parameters vary as

$$f(x, y) = 4 + (4 + x)) * y \tag{6.1}$$

where $1 <= x <= 7$ and $0 <= y <= 108$

This gives the range of input parameters as [4,1192]. Although the actual number of input parameters depends upon the number of subbasin a user is interested in.

Although the number of input dimensions varies, in the simulated user's experiments that we performed, we kept the size to four inputs only so that the model comparisons could be done. Also, as we can see the size of input dimension varies a lot, we can use different dimension reduction techniques like kohonen maps to reduce the total input dimension. Decision variable prediction algorithm can also be tried to find what particular subbasin a user is actually interested in.

6.3 Why Artificial Neural Network is the Best Choice?

The multi-layer perceptron (MLP) Artificial Neural Network is the universal function approximator [45]. A proof has been [46] provided that a specific recurrent architecture with rational valued weights has the full power of a Universal Turing Machine using a finite number of neurons and standard linear connections.

In our research we needed a machine learning algorithm technique which can adapt changes as user's needs and performances vary over time. ANN is the best candidate to adapt these changes. Also, as the number of input dimension vary for different users, the ANN will be able handle high dimensional input variables. ANN works better than other linear modeling techniques [47] [48] as the other linear models are good in global approximation while ANN works for both local and global situation.

6.4 Scaling Online Data

Another challenge we faced is scaling the parameter values. Because the data values are generated online, finding precise values of the actual maxima and minima prior to experiment is not possible. If the data is not scaled properly it would not give a good result.

The biggest challenge was having a limited amount of data and how could we use that limited data to use for training a better SDM . One of the solutions that we proposed is using the other machine learning algorithms to fine tune the ANN as described in the upcoming sections.

6.5 Artificial Neural Network

An ANN is a parallel, distributed information processing structure consisting of processing elements (which can possess a local memory and can carry out localized information processing operations) interconnected together with unidirectional signal channels called connections. Each processing element has a single output connection

which branches ("fans out") into as many collateral connections as desired (each carrying the same signal - the processing element output signal). The processing element output signal can be of any mathematical type desired. All of the processing that goes on within each processing element must be completely local: i.e., it must depend only upon the current values of the input signals arriving at the processing element via impinging connections and upon values stored in the processing element's local memory [49].

A diagram of feed forward ANN is shown in Fig. 6.1

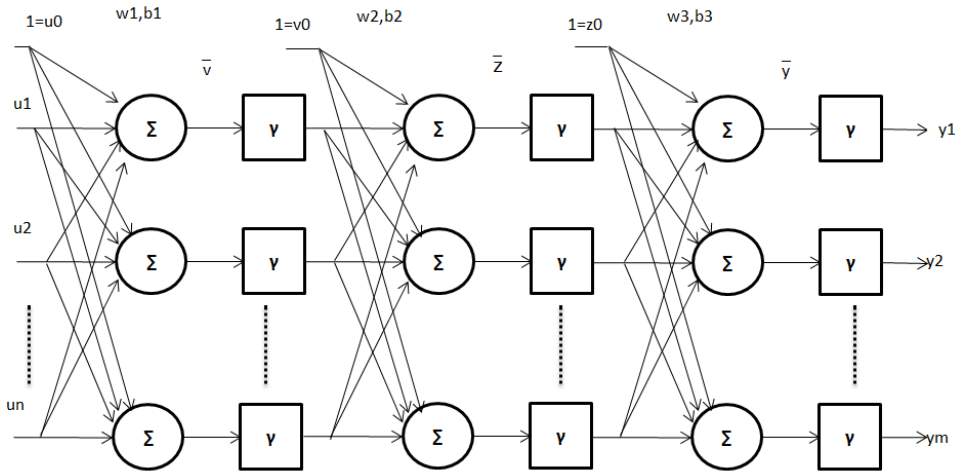

Figure 6.1. A Feed Forward Artificial Neural Network

6.6 Similar Work

Many user modeling techniques have been used in the past in various fields. user modeling is performed in the fields like in information retrieval systems [25] where the user's domain experiences and inquiry interests are modeled by the system. The main need of user modeling systems is the proper representation of the user model and the acquisition of assumptions about the user [50]. Another promising method of user modeling, using an ANN, was done by [26] where news were shown to the

user based on personal interest. The ANN tries to model the interests of the user and then rank the incoming news as relevant or irrelevant.

ANN and GA hybrid, ANN-GA [27], is used to solve water quality modeling problems. The hybrid system was created because the total search space was too big and computationally unfeasible. Fuzzy Model and ANN are used in estimating the sediment concentration [28]. ANN is used to model rain fall/runoff [29] and to model daily sediment yield [30] in various fields of hydrology. People [31] compared user modeling techniques like ANN, fuzzy logic, etc. for adaptive hypermedia systems and recommended their usage in different scenarios. Huang et.al. [32] used ANN to do multi-objective interactive optimization for reliability optimization. Some of the popular predictive statistical models are linear models, TFIDF-based models, Markov models, ANN, classification and rule-induction methods, and Bayesian networks [33]. Of those models ANN is good in expressing non-linear decisions [33].

Deep Learning is also one of prominent machine learning algorithm used by many people especially in handwriting recognition [37] and [38]. A faster implementation of Deep Learning is done [39] using GPUs which are fast computing hardware.

6.7 Limitations of SDM using ANN

Training an ANN with a limited set of data, especially if the number of output parameters is low, while the number of input parameters is very high, is not successfully done. We tried to formulate methods to train ANN for interactive optimization specific to environmental planning systems, where the number of input parameters is very high and the number of output parameters is very low. The advantage of training the ANN in such a manner is that the ANN will be able to capture randomness more effectively. The ANN is good at capturing non-linearity [33] in data; it is a very useful way of extending the performance of the ANN in situations where we have very limited data. Since the ANN is good in adapting to the changes [51], via change in

the neuron's weight, this technique is helpful in simulating the behavior of humans which changes with time.

In our current work, we have run simulated user experiments in which the human rating criteria is provided using simple/complex functions as described in Tables 7.1 and 7.2. Also, we have shown that the situations where ANN fails to learn because of limited learning data, we can use other learning methods to generate more learning data and incorporate that in ANN to get better performance by minimizing the learning error.

6.8 SDM Using ANN

Simulated Decision Maker Learning using ANN. Four different types of ANN are:

a) Simple Neural Network.

b) Neural Network with Extended Data.

c) Neural Network with Cumulative data (from previous generations).

d) Neural Network with Cumulative data with Extended Data.

The extended data Neural Network is created using the data generated from Multi Variant Normal Random Distribution of one of the best-fit linear/non-linear models.

6.9 SDM Using Other Models

A number of Simulated Decision Maker using other Models are being created. Different kinds of linear and non-linear modeling machine learning techniques are used to model the users search preferences.

6.9.1 ANFIS

Fuzzy Model: We have used Fuzzy model ANFIS [52] [53], which is an adaptive network based fuzzy inference system.

Limitation of ANFIS: Although ANFIS is a good machine learning model that can be used for user modeling but its limitation is for high dimensional input data it takes lot of time to create rule base which is computationally not feasible, [54]. Although, using powerful GPUs, this problem can be solved. Because the fuzzy rules can be created in parallel, GPUs can be used to create large rule based for high dimensional data, [55].

6.9.2 Linear/Non-linear Classification Models

We also used different linear and non-linear modeling techniques to create different kinds of SDM. Various different Classification models that we used in our research are:

- linear: Fits a multivariate normal density to each group, with a pooled estimate of covariance.
- diaglinear: Similar to linear, but with a diagonal covariance matrix estimate (naive Bayes classifiers).
- quadratic: Fits multivariate normal densities with covariance estimates stratified by group.
- diagquadratic: Similar to quadratic, but with a diagonal covariance matrix estimate (naive Bayes classifiers).
- mahalanobis: Uses Mahalanobis distances with stratified covariance estimates.
- Least Square Fit using Matrix Inversion.

(sources Matlab 2011 Statistics Toolbox)

For every NSGA2 search with small populations size, we create around ten different SDM Models. The six models are linear classification models and four others are different kinds of Neural Network Models described above. Out of these models, the

classification models are used to fine-tune the ANN for their extended versions. The simulated experiments show promising results in using such method.

6.9.3 SDM Using Deep Network

Because the accuracy of ANN is still limited based on the availability of good training and testing data, we have used Deep Learning technique to create Deep Network to get more accurate SDM. In Deep Learning, learning of the network is done at multiple levels of abstractions [56]. The Higher level of abstraction is defined using the lower levels.

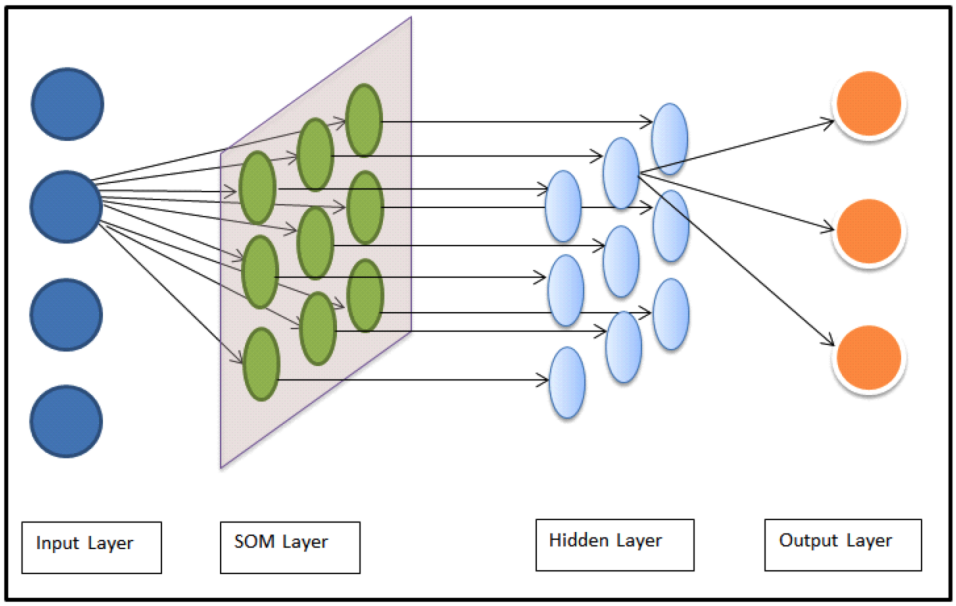

Figure 6.2. Deep Network

The Deep Network, as shown in Fig. 6.2, in our research is a two layer Network in which the first layer is Self Organizing Map(SOM) aka Kohonen Map. The second layer is created as two layer Multi Layer Perceptron (MLP). The training of each layer is done as greedy layer wise training [57].

Table 6.1

Deep Network

Model Name	SOM Nodes	Model Number
Deep1	SOM 10	1
Deep2	SOM 25	2
Deep3	SOM 50	3
Deep4	SOM 100	4
Deep5	SOM 200	5
Deep6	SOM 250	6
Deep7	SOM 500	7

To Create Deep Network, we have used the training data from different Searches and used that data to create a pre-training of Deep Network, which is done as creating the SOM. After pre-training the deep network an ANN is trained and tested for different search data. By applying this technique the first layer acts as a filter or an abstraction of the input data which is then transferred to Next Layer for further processing.

For the experiments we created Seven different Deep Networks whose details are shown in Table 6.1. The SOM nodes tells how many nodes are used to create the map.

7 EXPERIMENTS AND DISCUSSION

In this chapter I discussed various experiments we performed to prove the effectiveness of our algorithms in running the environmental planning system.

7.1 Experiments Batch Mode Learning

We did experiments for different optimization algorithms for the region 7 of Eagle Creek Watershed as shown in Fig. 5.1. These experiments prove the successful usage of different optimization algorithms that we have developed.

7.1.1 Optimal Solution

Since the search space was small, we exhaustively obtained all the possible solutions as shown in Fig. 7.1 for the region 7 of Eagle Creek. We choose small search space to compare the performance of different Algorithms. If the search space will be very large like the entire watershed, then comparing the algorithm with optimal solution would be difficult. We have generated the optimal pareto-front denoted as Optimal, from those solutions using Non-Dominated Sorting Algorithm of NSGA2. Based on the optimal solution we compared the results of NSGA2, Dist. NSGA2, DPLA, Dist. DPLA, Dist. RLBM (Reinforcement Learning in Batch Mode) Type I, Type II and Type III.

7.1.2 NSGA2 Vs Dist NSGA2

Test results are shown in Fig. 7.2 for the NSGA2 and Dist. NSGA2 Algorithms. The tests were done for twenty-eight individuals for over twenty generations. We can see that except a few points Dist. NSGA2 performed closer to NSGA2. But the Dist.

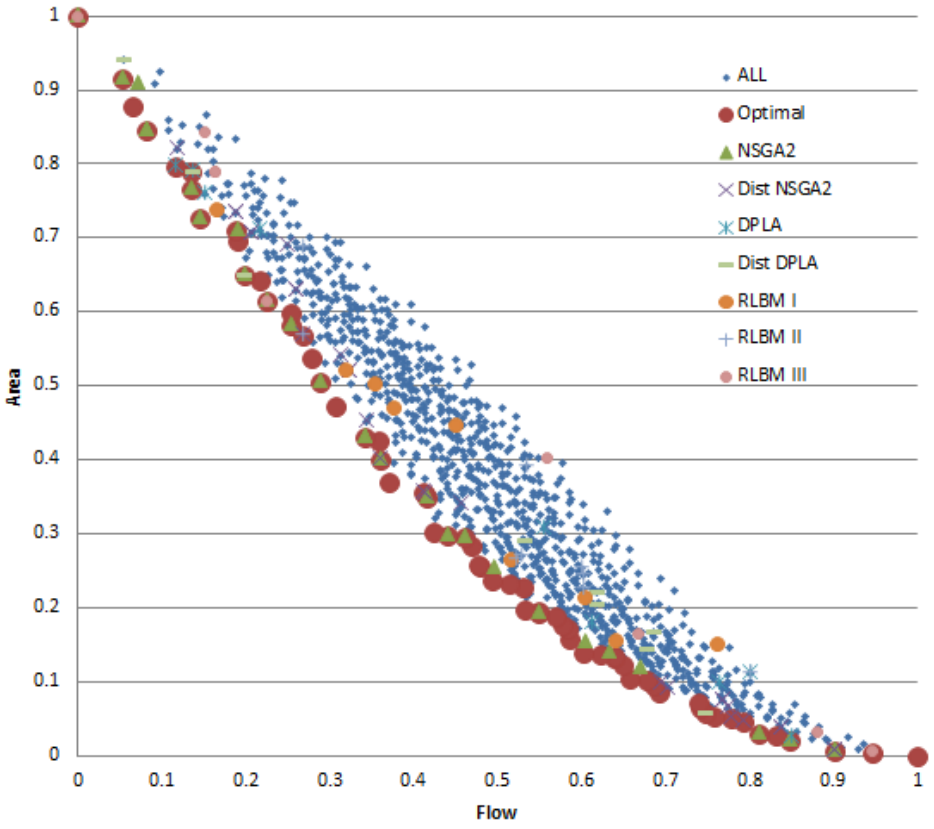

Figure 7.1. All Solutions

NSGA2 was much faster in execution as the entire generation was evaluated at once, but in NSGA2 the individuals were evaluated in sequence.

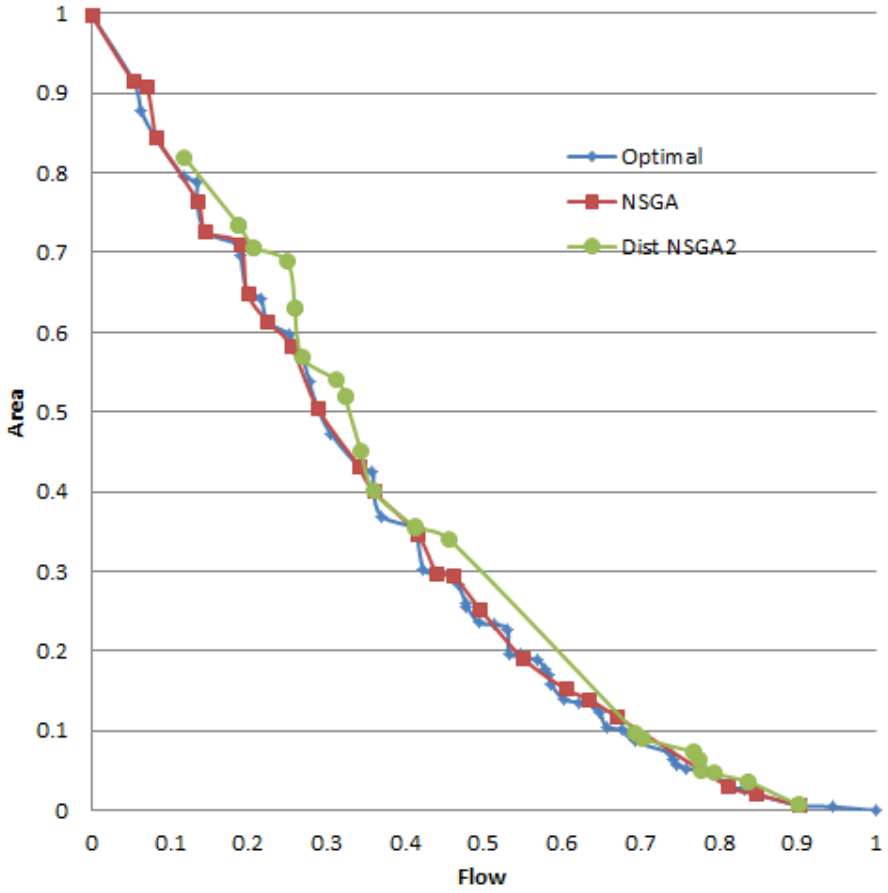

Figure 7.2. NSGA2 Vs Dist. NSGA2

7.1.3 DPLA

Tests were performed for DPLA and Dist. DPLA. As show in Fig. 7.3 we can see that at some point Dist. DPLA gave better results than DPLA while at some points DPLA is better, otherwise the results are almost the same for both. However, in Dist. DPLA all the weight pairs were run separately as they were independent of

each other, which gave almost 9X speed-up to actual DPLA algorithm, which was sequential. So, Dist. DPLA is much faster than DPLA.

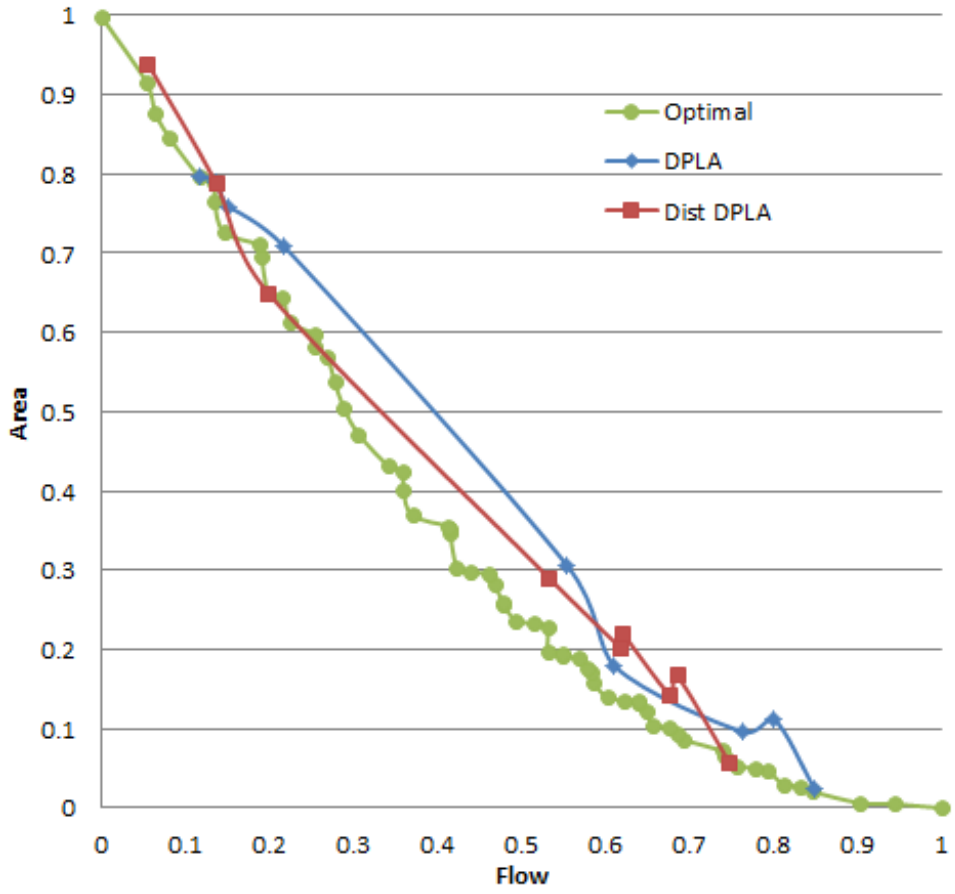

Figure 7.3. DPLA

7.1.4 NSGA2 Vs DPLA

Comparing the results of NSGA2 with DPLA as shown in Fig. 7.4, we can see that NSGA2 and Dist. NSGA2 gave better results than DPLA. However at some points Dist. DPLA gave better results than others. But the running time of NSGA2 is much higher in this case. When we use Dist. NSGA2, its running time was much

less than DPLA, while Dist. DPLA was much faster than all the others. Overall Dist. NSGA2 is better in speed and accuracy.

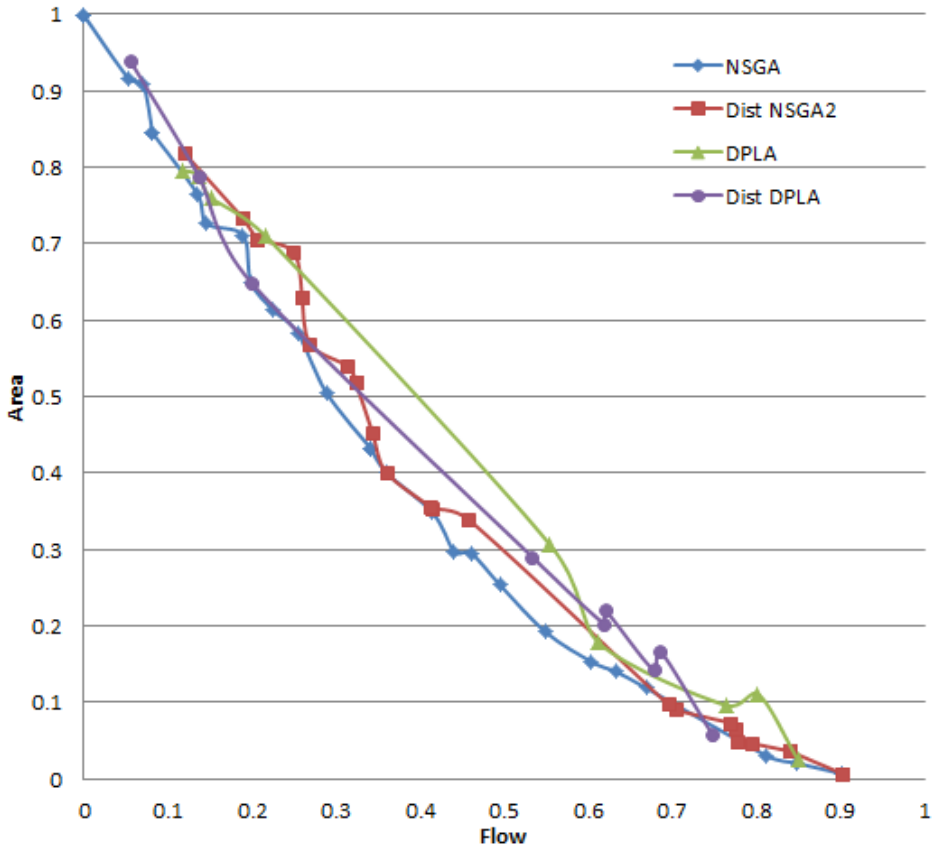

Figure 7.4. NSGA2 Vs DPLA

Based on the above observations and applying the concept of Batch mode learning in Dist. DPLA, the following set of RLBM algorithms have been tested.

7.1.5 RLBM

As we can see in Fig. 7.5 , based on points which are close to optimal, of all the three types, Type I gave better result than all others, while the result of Type III is in between Type I and Type II. In case of speed Type II is the fastest, while Type III is slower than Type II and Type I is the slowest of all.

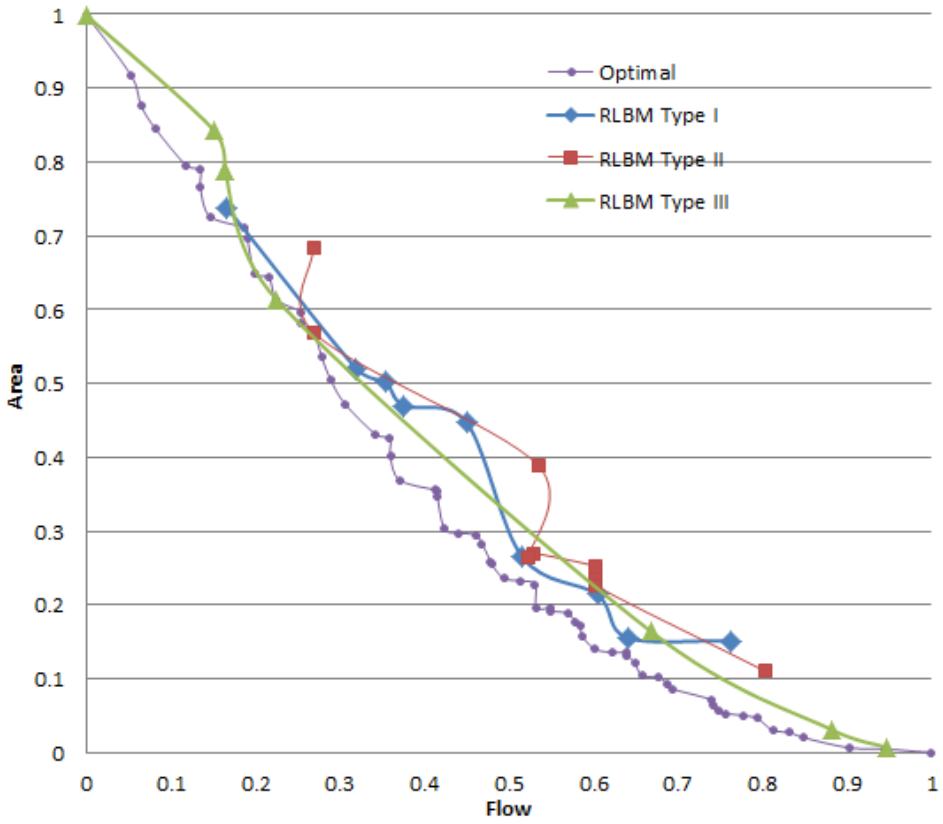

Figure 7.5. RLBM

7.1.6 DPLA vs RLBM

Here we compared the test results for DPLA, Dist. DPLA, RLBM Type I, RLBM Type II and RLBM Type III as shown in the Fig. 7.6. We can see that DPLA, Dist. DPLA, RLBM Type I, RLBM Type II and RLBM Type III, all intersected the Optimal line.

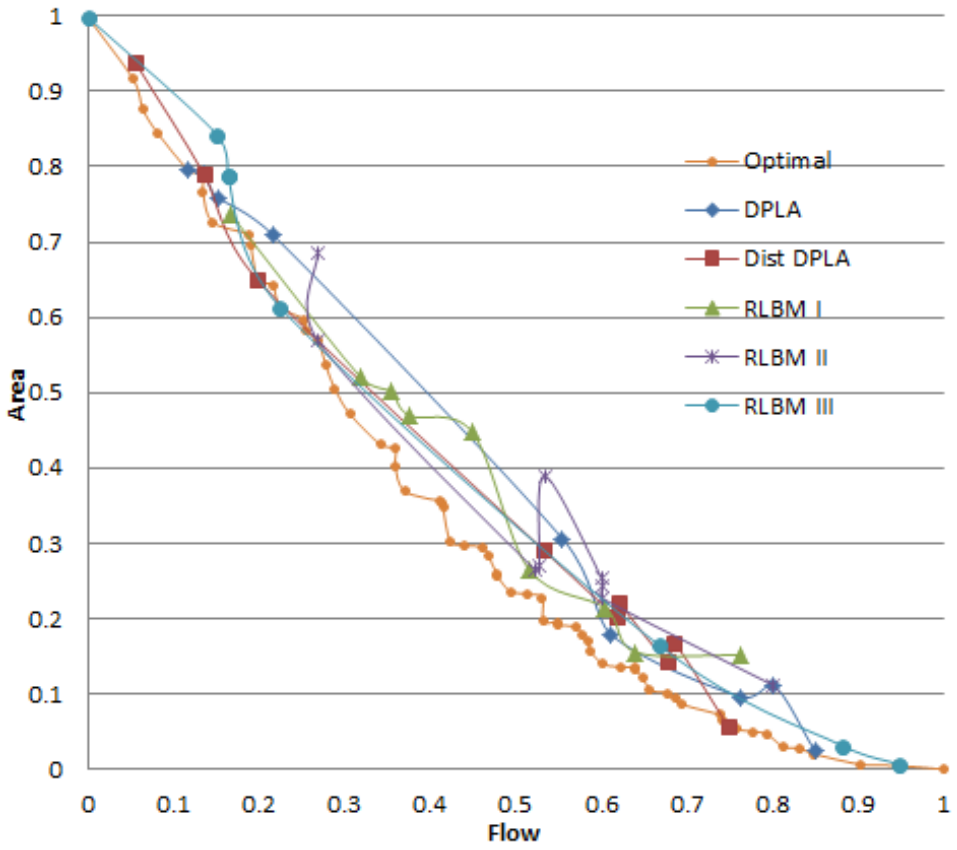

Figure 7.6. DPLA vs RLBM

Convergence Time

As shown in Fig. 7.7 and Fig. 7.8, when we compare the number of iterations and time of convergence, we find that RLBM Type II is the fastest of all while RLBM

Type I is the slowest. RLBM Type III takes time in between Type I and Type II, while Dist. DPLA takes time little higher than RLBM Type III. DPLA takes time higher than Dist. DPLA. Note that the time for DPLA is cumulative as it is a sequential algorithm, so at the end of the search the overall time will be the sum of all the convergence time for DPLA.

The number of Iterations and actual time differ for some algorithms because in the case of DPLA and Dist. DPLA only a single SWAT evaluation was performed while in the case of RLBM a set of seven SWAT evaluations were performed (A batch of individuals). Running a set of seven SWAT evaluations was much slower than running a single SWAT evaluation. This is one of the reason RLBM Type I is very slow apart from the slow rate of convergence.

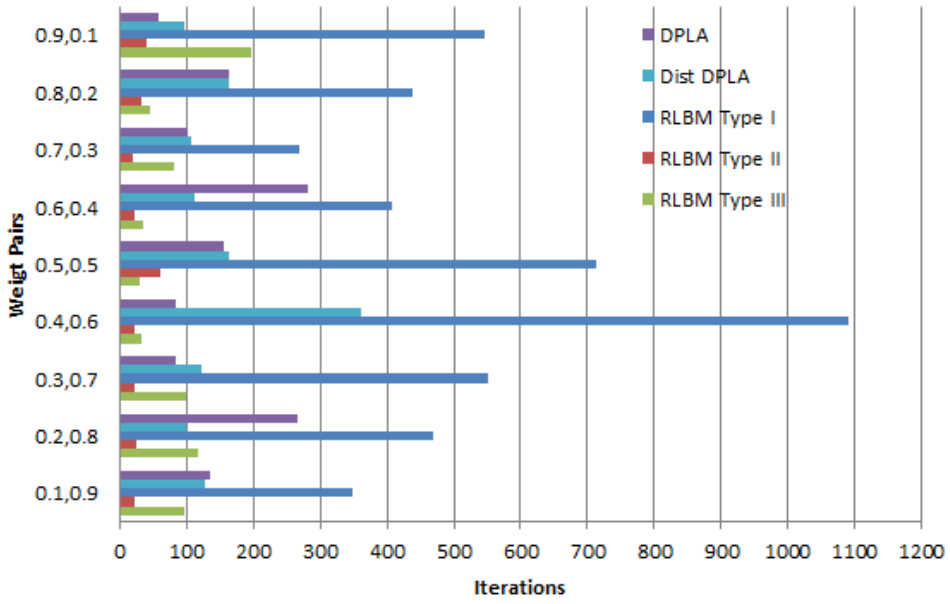

Figure 7.7. No. Of Iterations

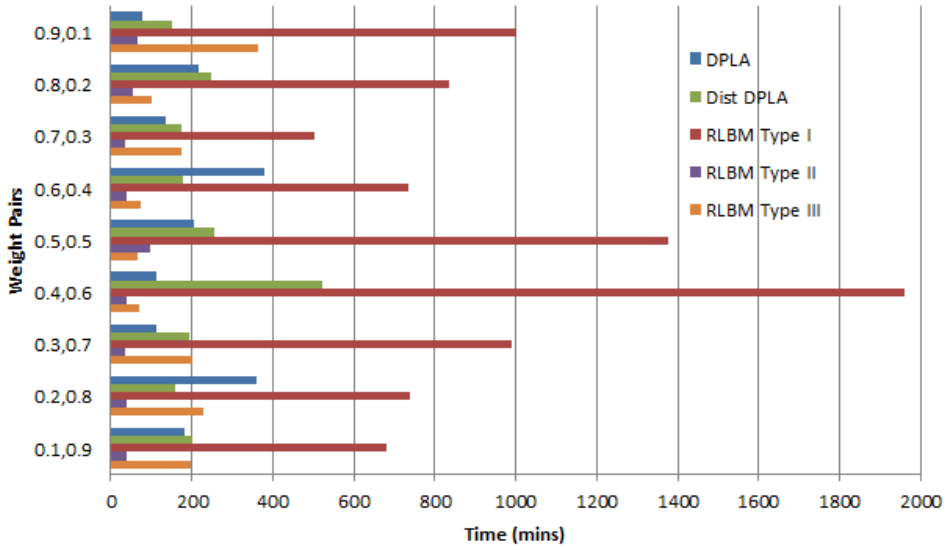

Figure 7.8. Time to Converge

7.1.7 Comparing All

Overall, all the algorithms gave comparatively similar results as shown in Fig. 7.9. Some are better in convergence time and some are better in accuracy.

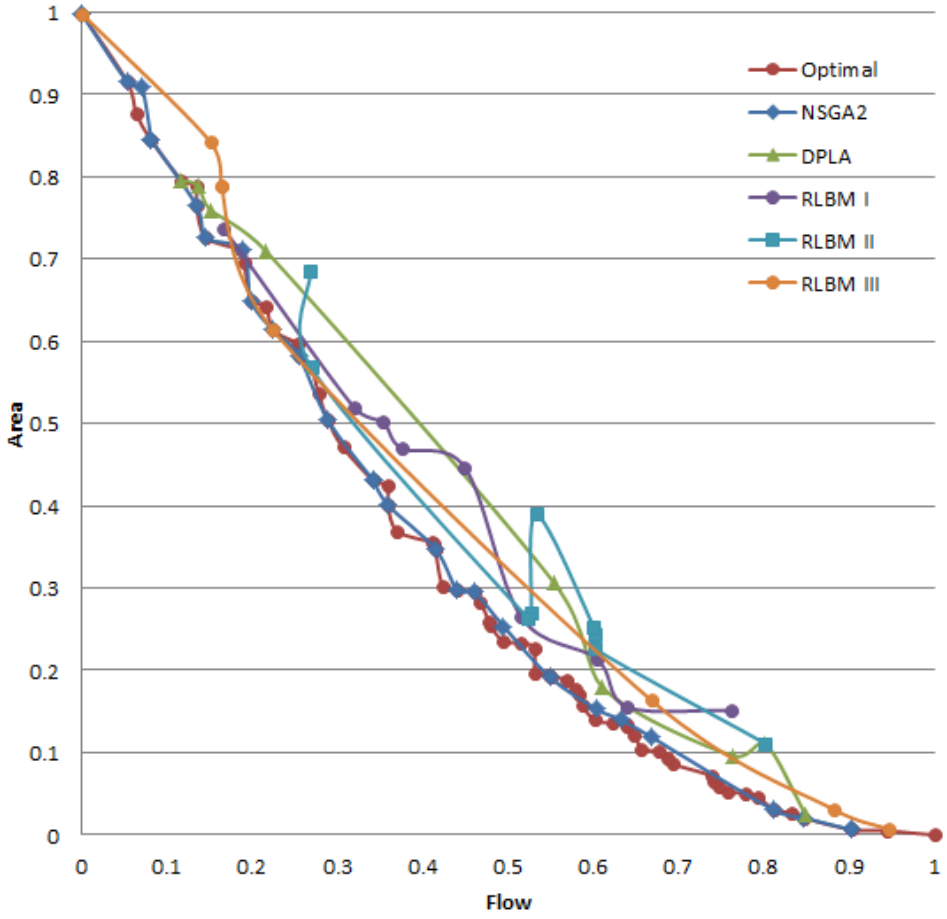

Figure 7.9. Compare All

In these experiments, we successfully tested the performance of Batch Mode Learning Technique in Decentralized Distributed Pursuit Learning Automata in multiple ways. These algorithms can be used to optimize various problems in the environmental system domain. We can see that the faster optimization algorithm can produce results faster and easy in decision making.

Table 7.1

Simple Rating Functions

Name	Rating Function	1	2	3
f0	$f0 * 1000$	$<= 700$	> 700 and $<= 725$	> 725
f1	$f1 * 1000$	$<= 670$	> 670 and $<= 730$	> 730
f2	$f2 * 1000$	$<= 640$	> 640 and $<= 675$	> 675
f3	$f3 * 1000$	> 670	$<= 670$ and > 635	$<= 635$

7.2 Experiments User Modeling - Simulated Users

We performed the user modeling using simulated user to test the working of our system. We created different rating functions as simple functions and mixed functions. Simple Functions are based on only single fitness function. The Simple Functions' ratings are computed based on the rules shown in Table. 7.1. Mixed Functions are created randomly based on linear/non-linear combination of simple functions. The Mixed functions ratings are computed based on the rules shown in Table 7.2. These rating functions are created by empirically analyzing the previously collected data by running simple genetic optimization. The functions created are mathematically linear and non-linear. Although in the real world, a user's preferences are hardly linear, we used these functions to check the performance of our system. Our assumption is if the system will be able to mimic these rating functions then it will also be able to map the actual human rating criterion.

We compared the performance of the system by creating different SDM using machine learning techniques like ANN, Fuzzy Logic (ANFIS), and other simple linear/non-linear modeling techniques. The job of the SDM is to map the environmental data to the rating criteria specified by the user, which in the current case are the rating functions.

Table 7.2
Some Linear and Non-linear Mixed Rating Functions

Name	Rating Function	1	2	3
F0	$(f0+f1+f2+f3)*1000/4$	$<= 700$	> 700 and $<= 725$	> 725
F1	$(f0^2 + f1^2 + f2^2 + f3^2) * 1000/4 + 200$	$<= 670$	> 670 and $<= 730$	> 730
F2	$(4f0 + 3f1 + f2 + f3/2) * 1000/8$	$<= 640$	> 640 and $<= 675$	> 675
F3	$(f0 * f3/f1 * f2) * 1000/2$	> 670	$<= 670$ and > 635	$<= 635$
F4	$(f0 * f2/f1 * f3) * 1000/2$	> 520	$<= 520$ and > 490	$<= 490$
F5	$(f0 * f1/f2 * f3) * 1000/2$	$<= 270$	> 270 and $<= 300$	> 300

7.2.1 Simulated Users Preliminary Test

For each rating function, a simulated user was created to provide a rating mimicking as a real user. Three different SDM models were created for each function. To create a SDM model, we collected 120 different designs based on the rating given by the simulated user.

Legend for the figures:

- HR-Human Rating is the rating given by the Simulated Human using the above rating functions.
- NNR-Neural Network Rating is the rating given by the SDM created using ANN.
- ANFISR- Adaptive Network based Fuzzy Inference System Rating is the rating given by the SDM created using ANFIS.
- LSFR - Least Square Fit using Matrix Inversion Rating is the rating given by the SDM created using Least Square Fit.

In the following figures, we are showing different ratings given to the testing individuals for different models by different functions. The simulated ratings for simple functions are shown in figures Fig. 7.10, Fig. 7.12, and 7.13. While the simulated ratings for the mixed functions are shown in figures Fig. 7.11, Fig.7.14, and Fig. 7.15.

As we can see in the Fig. 7.10 and Fig. 7.11, the ratings given by ANN is consistent with the actual human rating (HR).

As shown in figures Fig. 7.12, 7.13, Fig.7.14, and Fig. 7.15, we see that there are many places where ANN did not capture the correct rating criteria. As we can see, the solid dot (shown as Neural Network Rating) is constant, i.e., Neural Network gave the same rating. Although the ANN was slightly better at capturing variation in ratings for mixed functions, in the case of simple functions f2 and f3, the rating is constant. This shows that the ANN is able to capture non-linearity for mixed

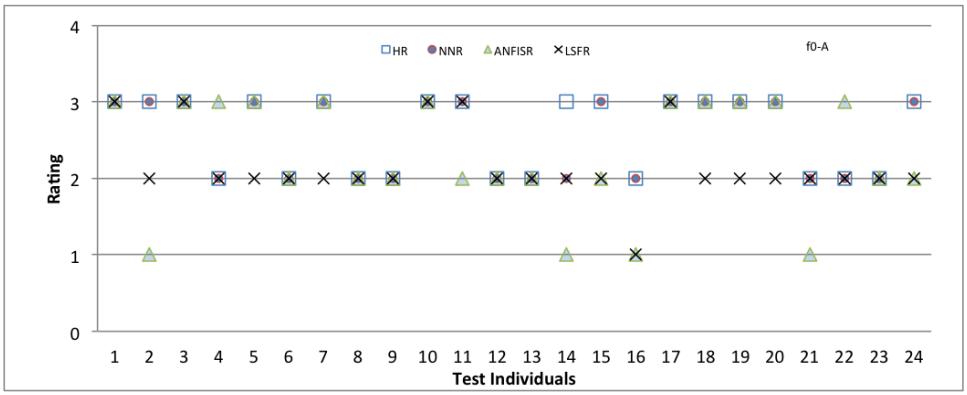

Figure 7.10. Simulated User with Rating f0

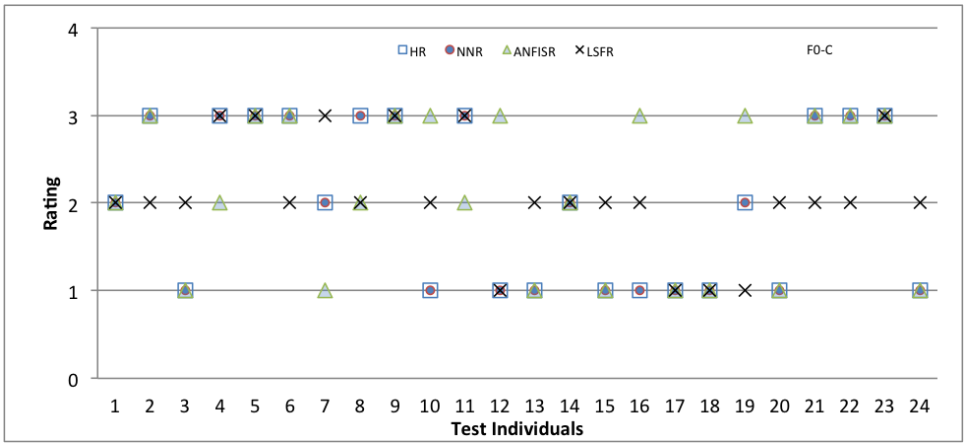

Figure 7.11. Simulated User with Rating F0

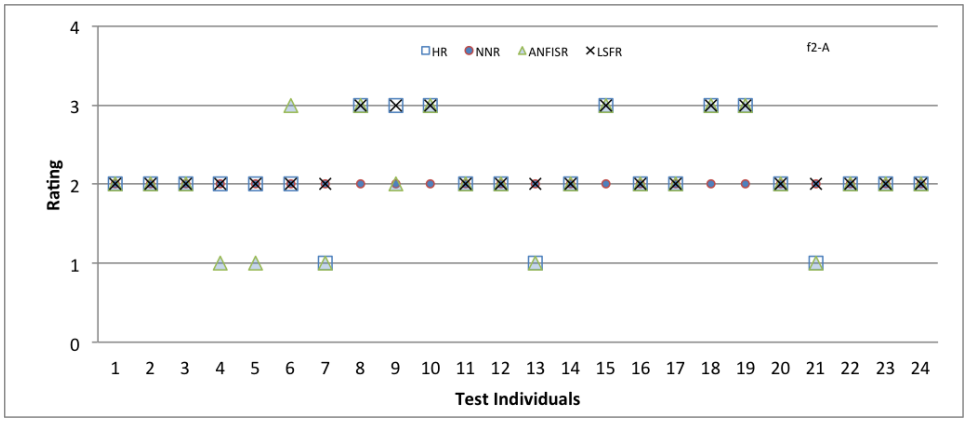

Figure 7.12. Simulated User with Rating f2

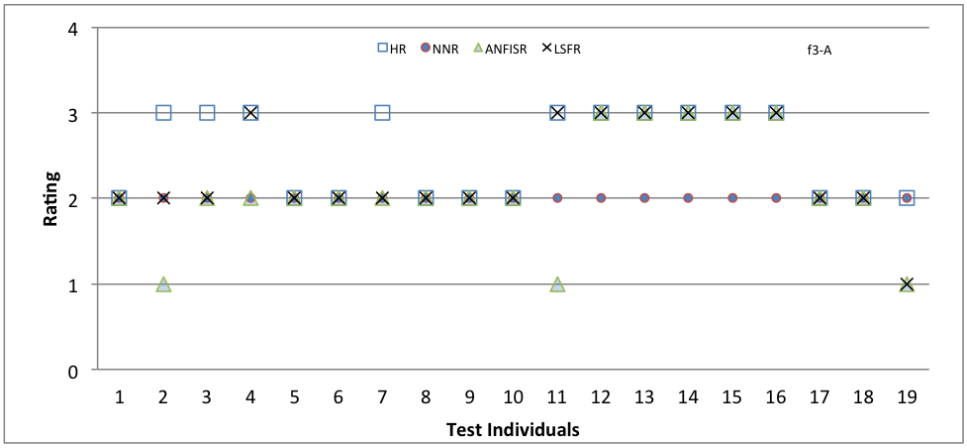

Figure 7.13. Simulated User with Rating f3

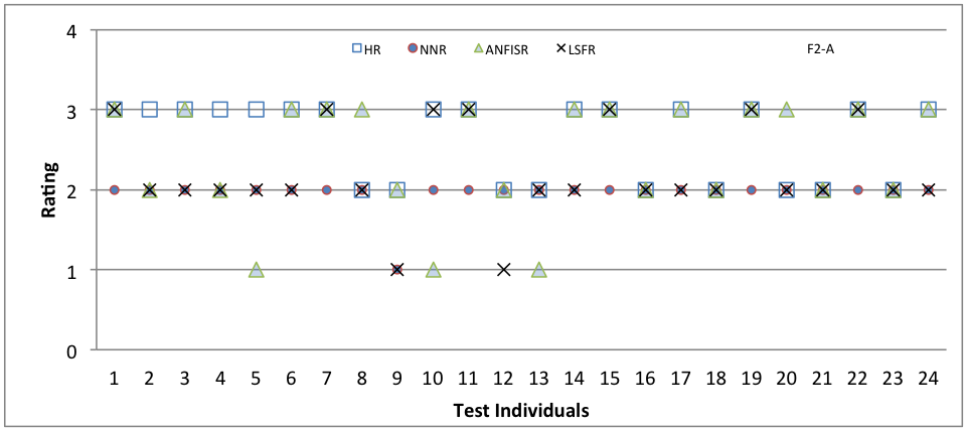

Figure 7.14. Simulated User with Rating F2

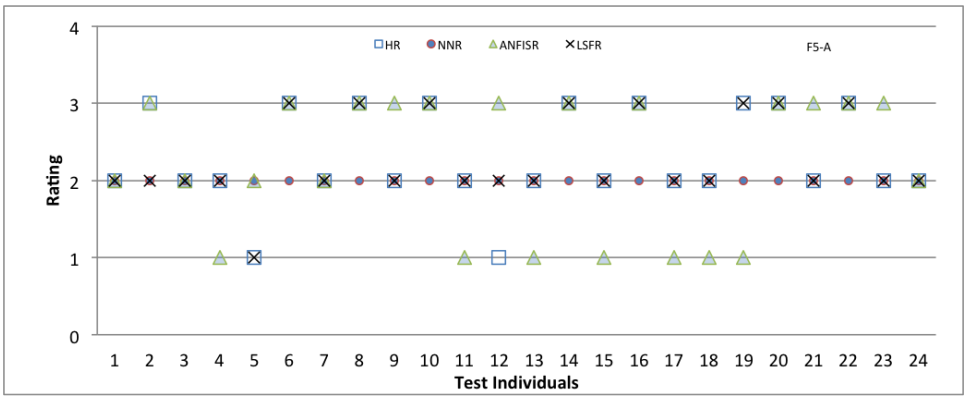

Figure 7.15. Simulated User with Rating F5

functions better when compared to simple functions, in the case of limited data. The ANFIS is also good in predicting the user's rating criteria, but it doesn't work for high dimensional data [54].

7.2.2 Simulated Users Improved Model Test

To solve the problem created by the limited data, we used the other linear/non-linear modeling techniques to generate more random data to train Neural Network (NN Ext.). The extra data is used to fine-tune the ANN. We assumed that the data is normally distributed, which is true in the cases when the number of samples tends to infinity. The data is generated using Multi Variate Normal Random Distribution. One of the linear/non-linear models having min error is used to generate more random data. We ran the tests for the different rating functions using the simple function and mixed functions. We also performed a test using Random rating to check the effectiveness of this technique.

The graphs in figures Fig. 7.16, Fig. 7.17, and Fig. 7.18, show the improvement in ANN created by using data generated by linear/non-linear modeling techniques over the previous ANN. As we can see, there are many places where the NN Ext Error (the number of errors from Modified Neural Network) is lesser than the NN Error (the number of errors from the original Neural Network). In the graphs we can see the values for triangle (NN Ext Error) is less than square (NN Error). Thus, fine tuning the ANN is possible in case of limited data where we can generate more random data using other linear/non-linear modeling technique.

7.3 Experiments User Modeling - Real Human Stake Holders

We ran the tests with the help of real human stake holders. Every user worked on either their local subbasin or on the entire watershed. They used the visualization data of their subbasin or watershed to give the feedback rating. The rating criteria is solely formulated by the user, based on their preference. Three different SDM

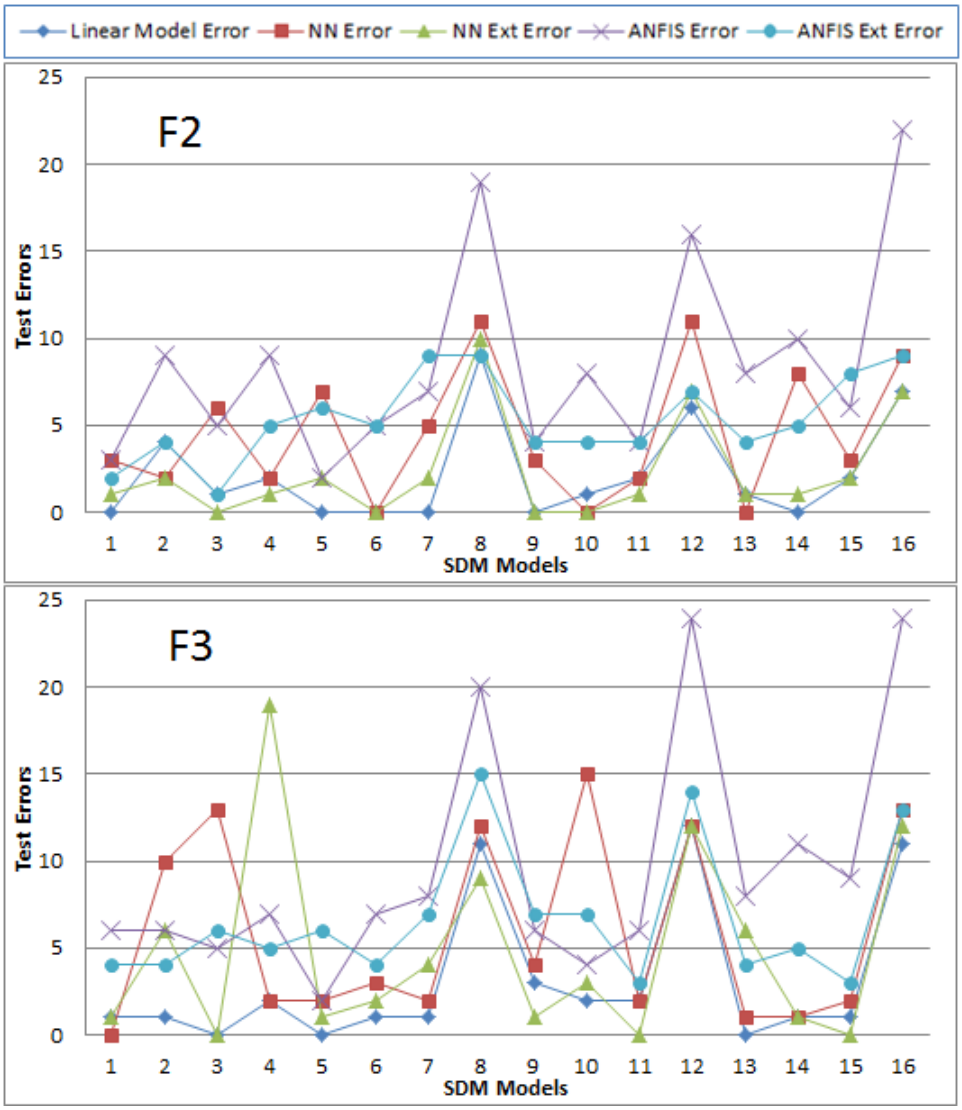

Figure 7.16. Mixed Rating Functions

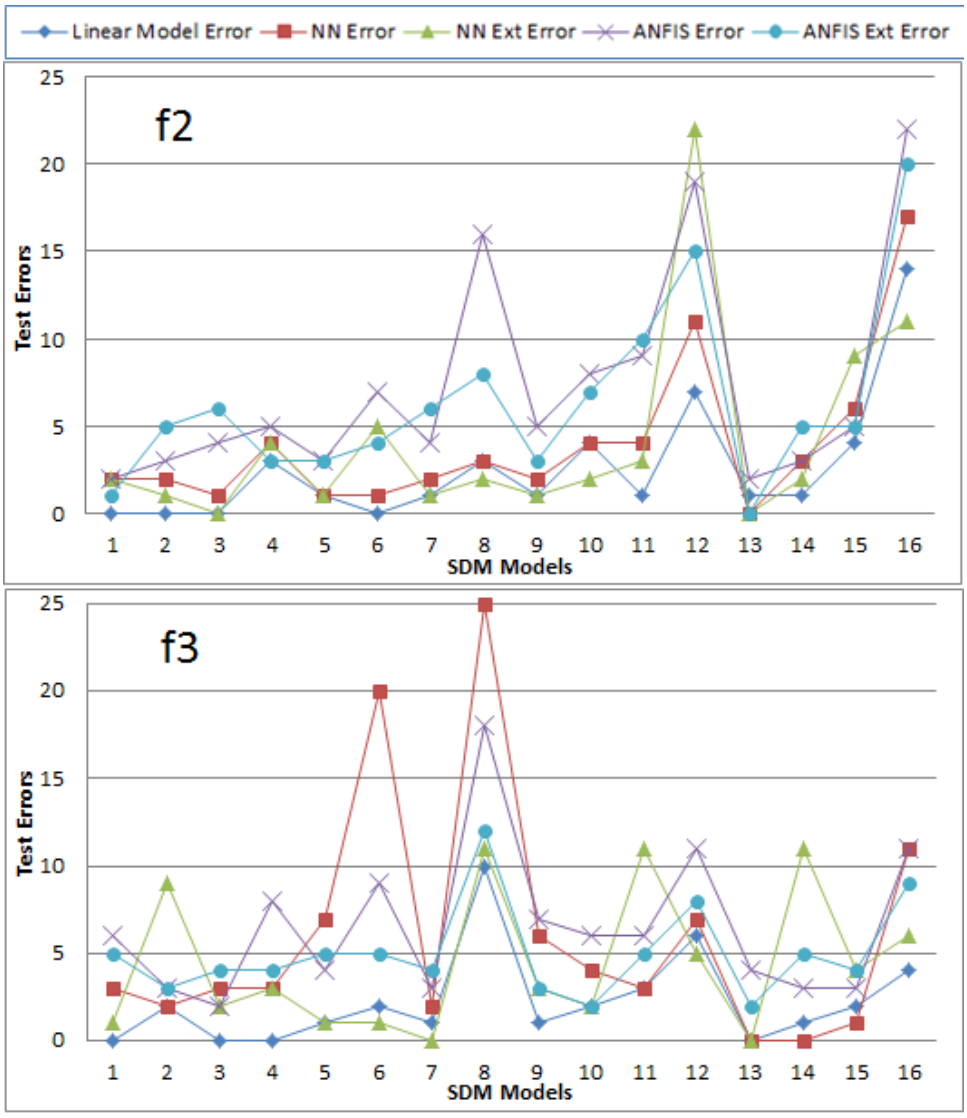

Figure 7.17. Simple Rating Functions

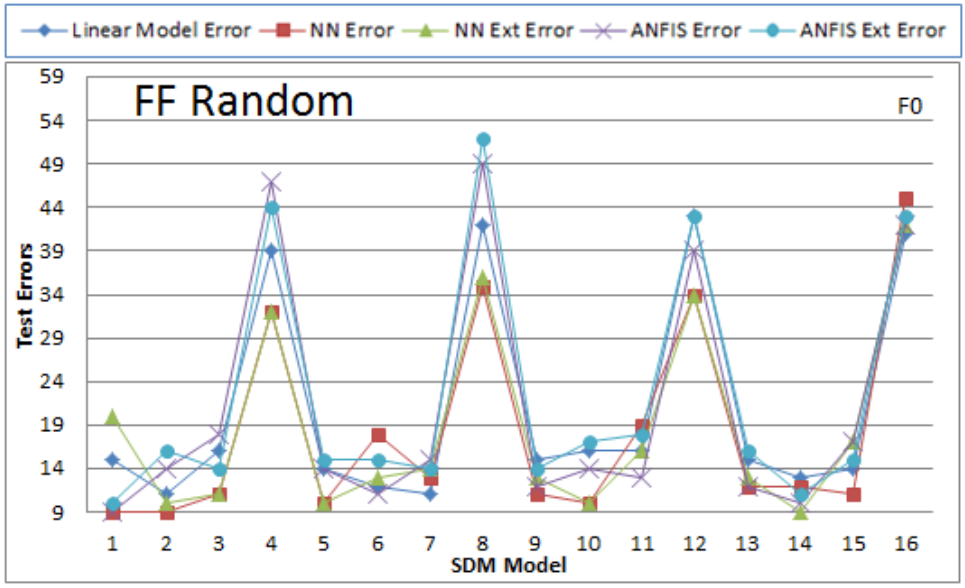

Figure 7.18. Random Rating Function

Table 7.3
SDM Model - Data Selection Criteria

Model Name	Data Selection from
SDM1	data from Search1
SDM2	data from Search2
SDM3	data from Search3
SDM4	data from Search1 and Search2
SDM5	data from Search1, Search2 and Search3

Searches were performed for every user. Five different SDM models were created based on the search data. Because the data is limited we used all the available data to create different SDMs.

As shown in Fig. 7.19, Fig. 7.20 and Fig. 7.21 we can see that Neural Network Ext. performed much better than Neural Network in each cases. In some of the cases ANFIS Ext. performed better than ANFIS. The Ext. model is created using data generated from one of the linear/non-linear models to fine tune the Neural Network or ANFIS as proved in the previous section, simulated user experiment.

7.4 Experiments User Modeling - Effect of Local Decision Making

The SDM models are created at three different levels of subbasins. Levels of Subbasins:

- Watershed Level: The fitness functions at the entire Watershed Level are used as input variable to SDM.
- Local Subbasin Level: The fitness functions at the Local Subbasin Level are used as input variables to SDM.

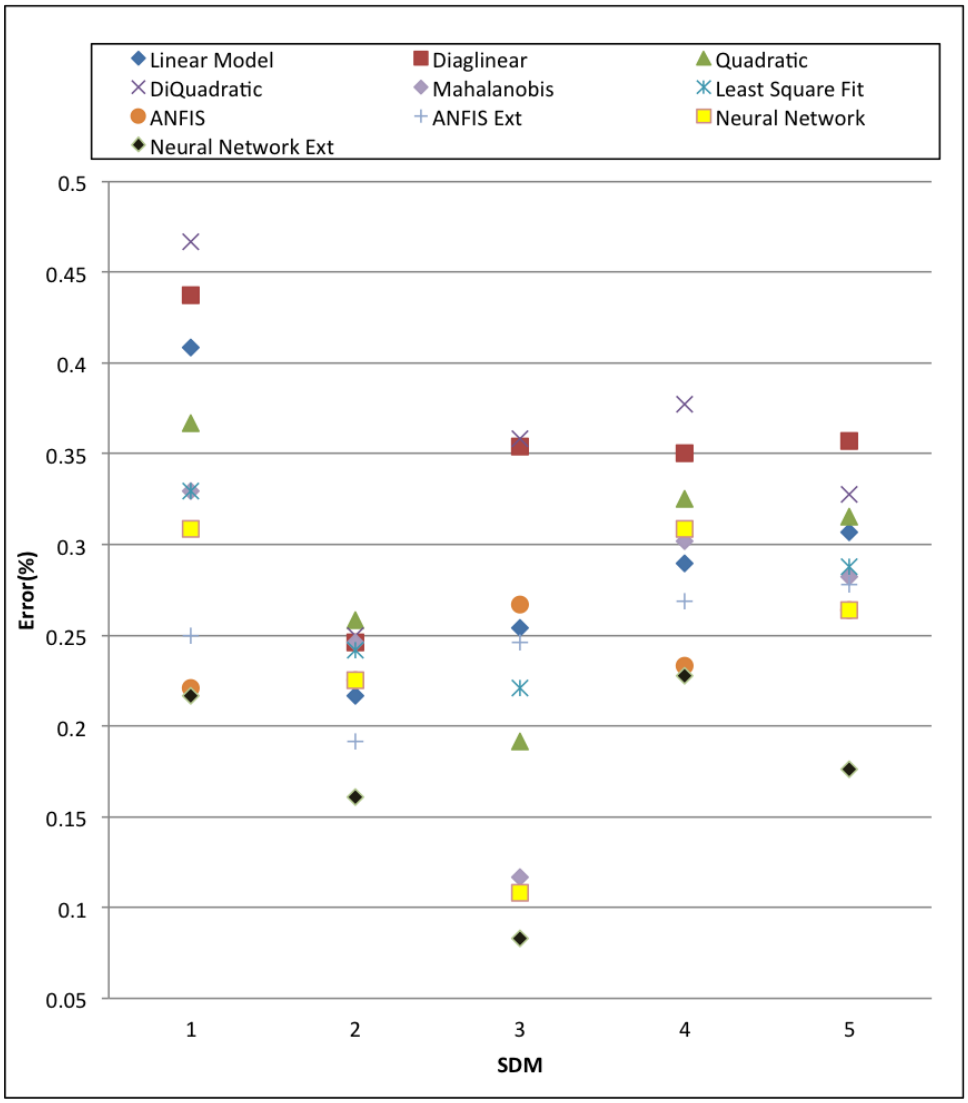

Figure 7.19. Real Stake Holder User I

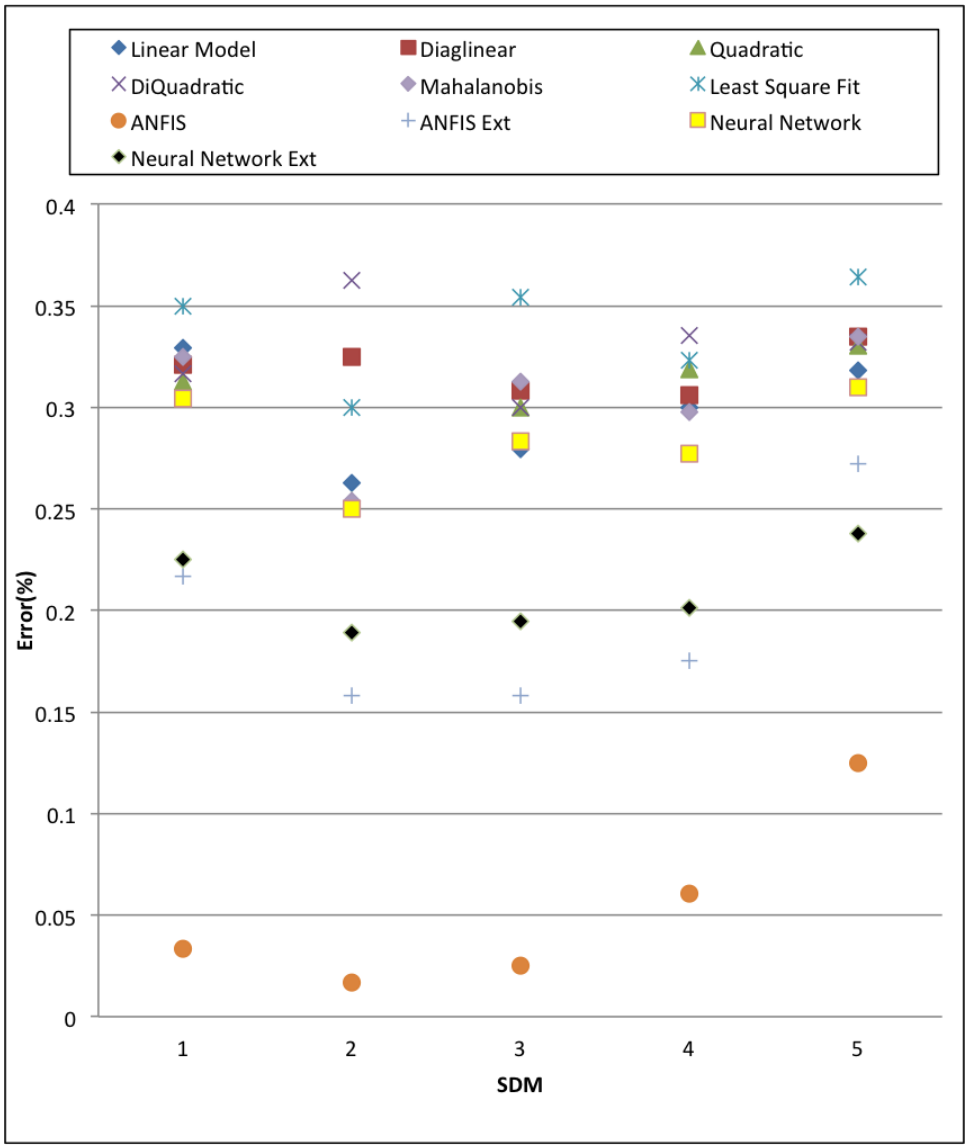

Figure 7.20. Real Stake Holder User II

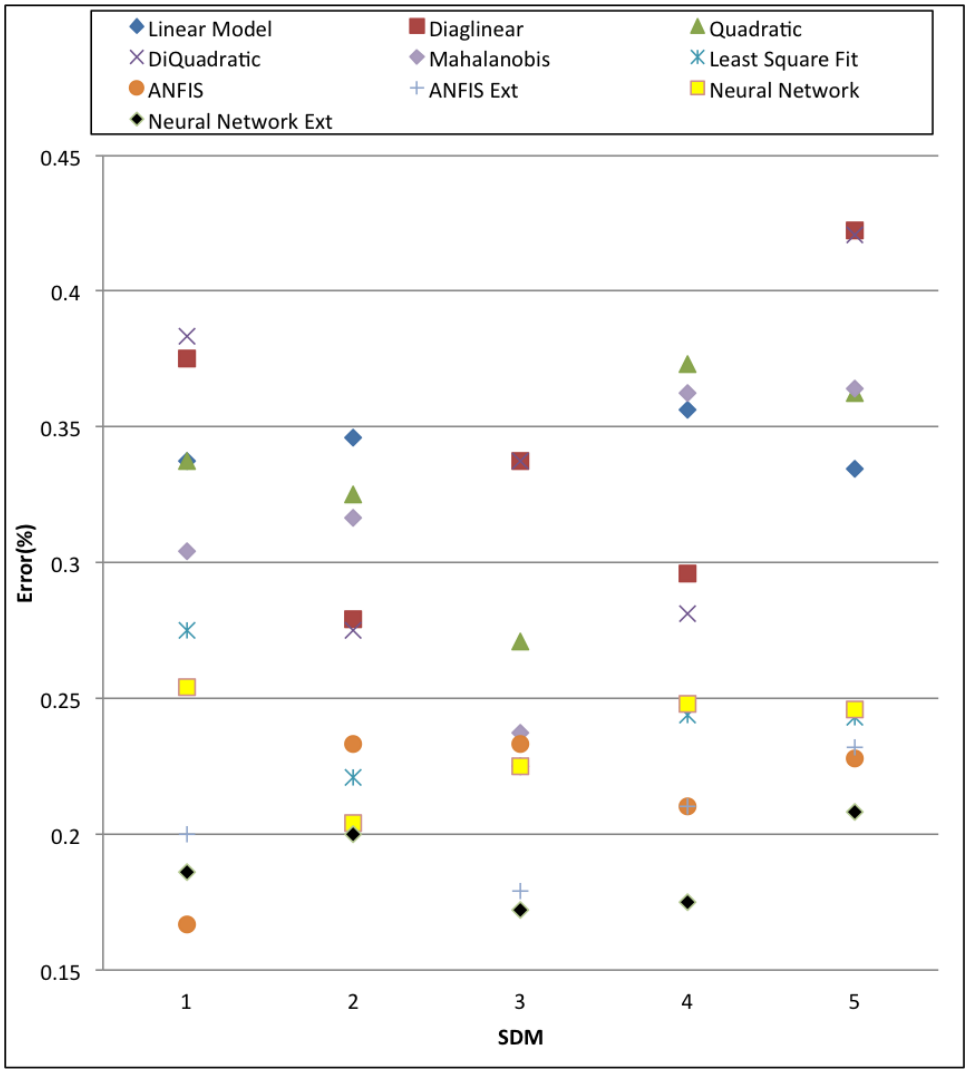

Figure 7.21. Real Stake Holder User III

Table 7.4
SDM Model - Other Models

Model Name	Model Number
Linear Model	1
Diaglinear Model	2
Quadratic	3
DiQuadratic	4
Mahalanobis	5
Least Square Fit	6
ANFIS	7
ANFIS Ext	8
Neural Network	9
Neural Network Ext	10

- Combined Level (Watershed + Local Subbasin): The fitness functions of both the Watershed Level and the Local Subbasin Level, together are used as input variables to SDM.

In this experiment we tried to identify which level fits the SDM perfectly to the user's rating criteria.

As shown in Fig. 7.28, Fig. 7.29, and Fig. 7.30 we can see that SDM created at Watershed level is the worst. The SDM created using either combined level or only local subbasins level are the best way of creating the SDM. One possible reason for this is because the user formulated the rating criteria based on his/her local subbasins instead of the watershed level.

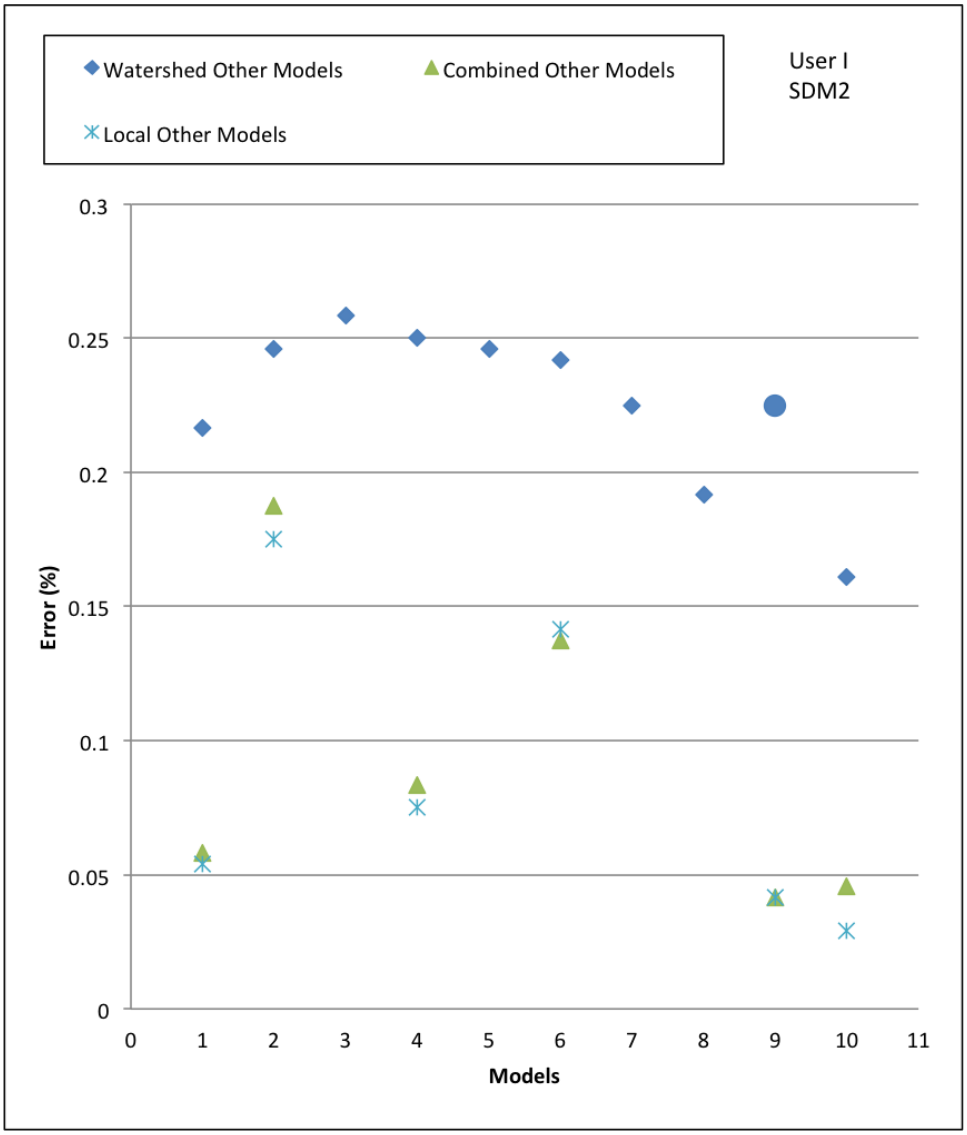

Figure 7.22. Local Decision Result - User I

For Details about Models Refer Table 7.4.

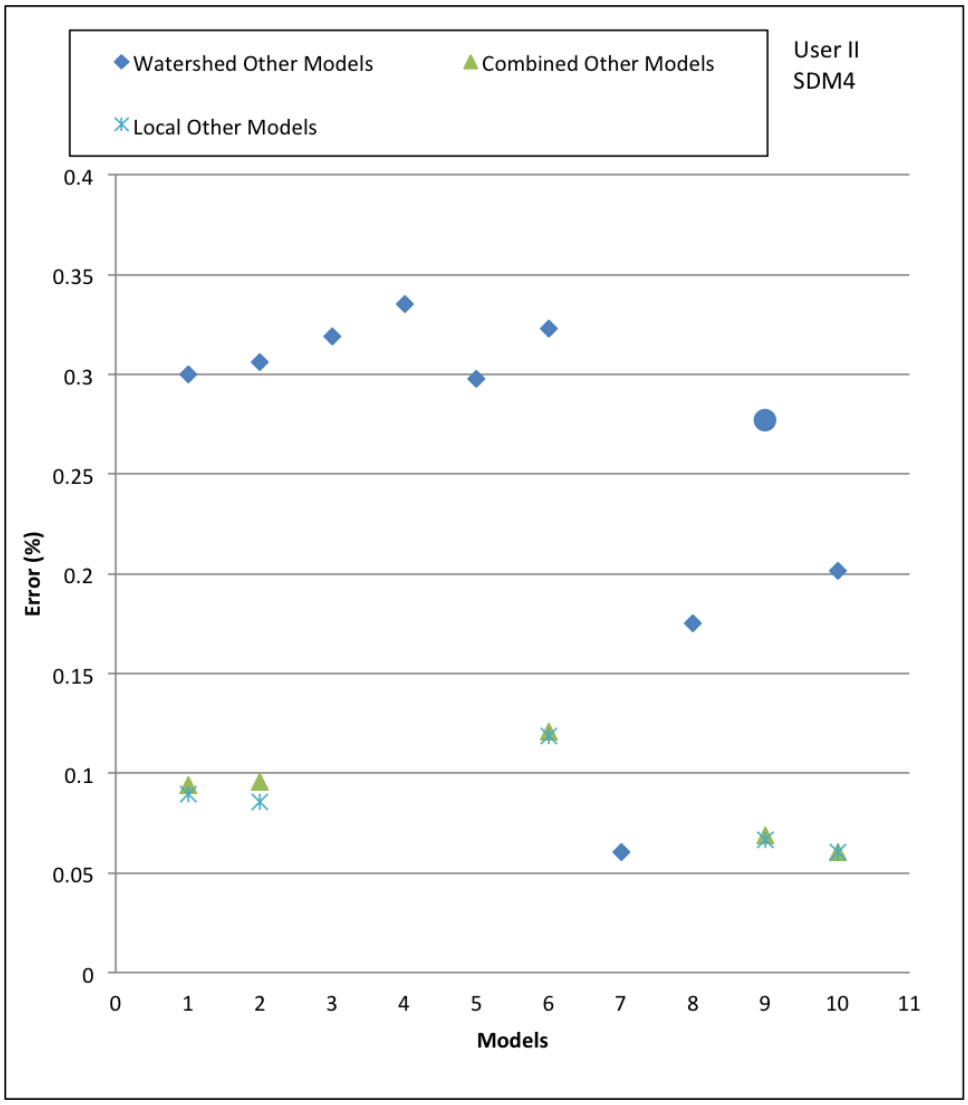

Figure 7.23. Local Decision Result - User II

For Details about Models Refer Table 7.4.

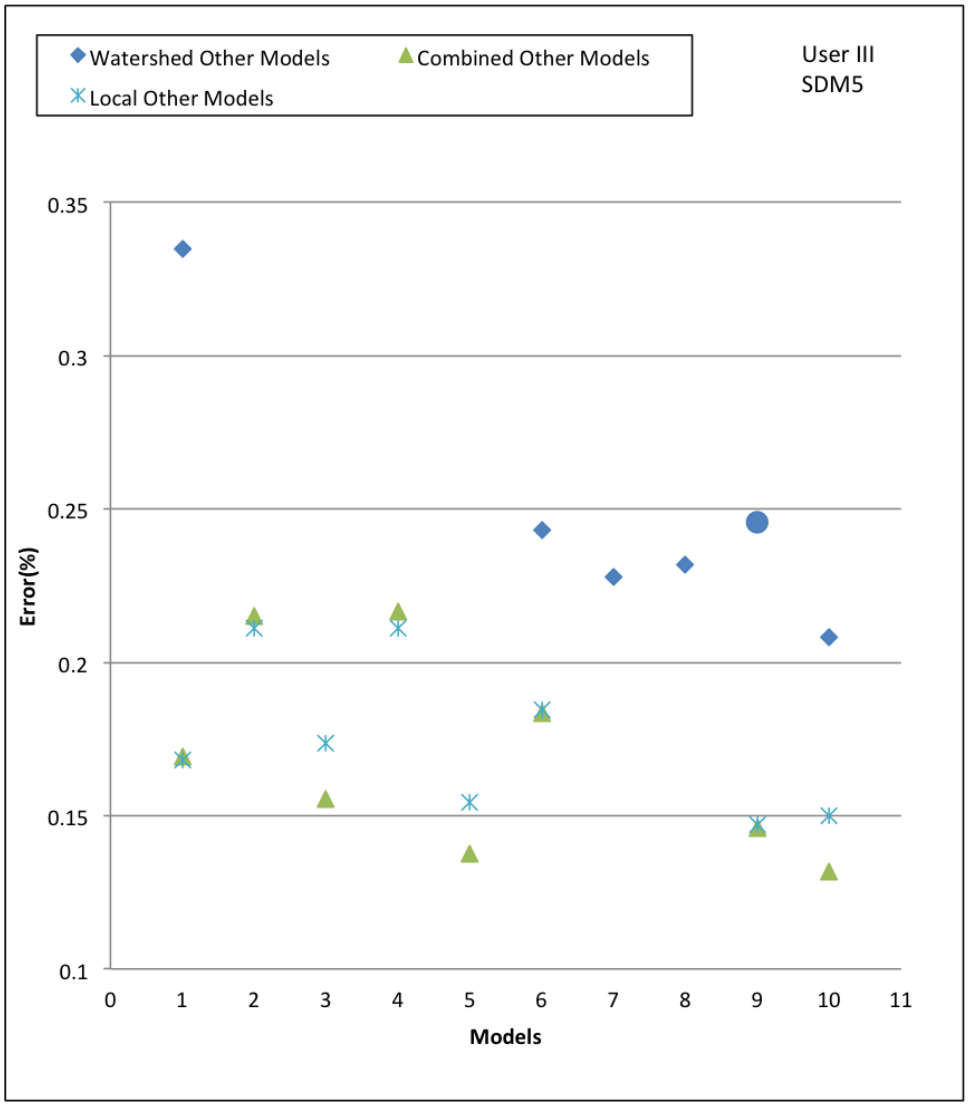

Figure 7.24. Local Decision Result - User III

For Details about Models Refer Table 7.4.

7.5 Experiments User Modeling - Deep Learning

To check the performance of deep learning network, we performed a number of experiments and compared its performance with other linear/non-linear modeling techniques, ANFIS, and ANN.

A set of seven different Deep Network Models were created for every level of subbasins. In the figures Fig. 7.25, Fig. 7.26, and Fig. 7.27 different deep networks are shown as Models numbered from 1 to 7. The details of different Deep Network is show in Table 6.1. The information about Other linear/non-linear models are given in the Table 7.4.

The SDM "Deep Using Search 1,2,3" is the deep network created using the data of Search 1,2,3 as the pre-training data of the SOM Layer. While in other Deep Networks, all the data from previously performed searches were used to do the pre-training. The main reason for the Deep Network created using Search 1,2,3 to perform better than the others is due to less noise in the data. So pre-training a Deep Network with a lot of data is not effective and bring unnecessary noise.

We can see the following observations about the Deep Networks using the Figures Fig. 7.25, Fig. 7.26 and Fig. 7.27:

- The Deep Networks created at the Watershed level performed better than the Watershed Other Models. The ANN shown as big blue dot did not perform as good as the Deep Network.
- The Deep Network created at the Combined level performed better than the Combined Other Models.
- The Deep Network created at the Local level performed better than the Local Other Models.
- The Deep Network created using the data from Search 1,2,3 is the best SDM model created with minimum error.

Thus, the Deep Network created using Search 1,2,3 is the best SDM model.

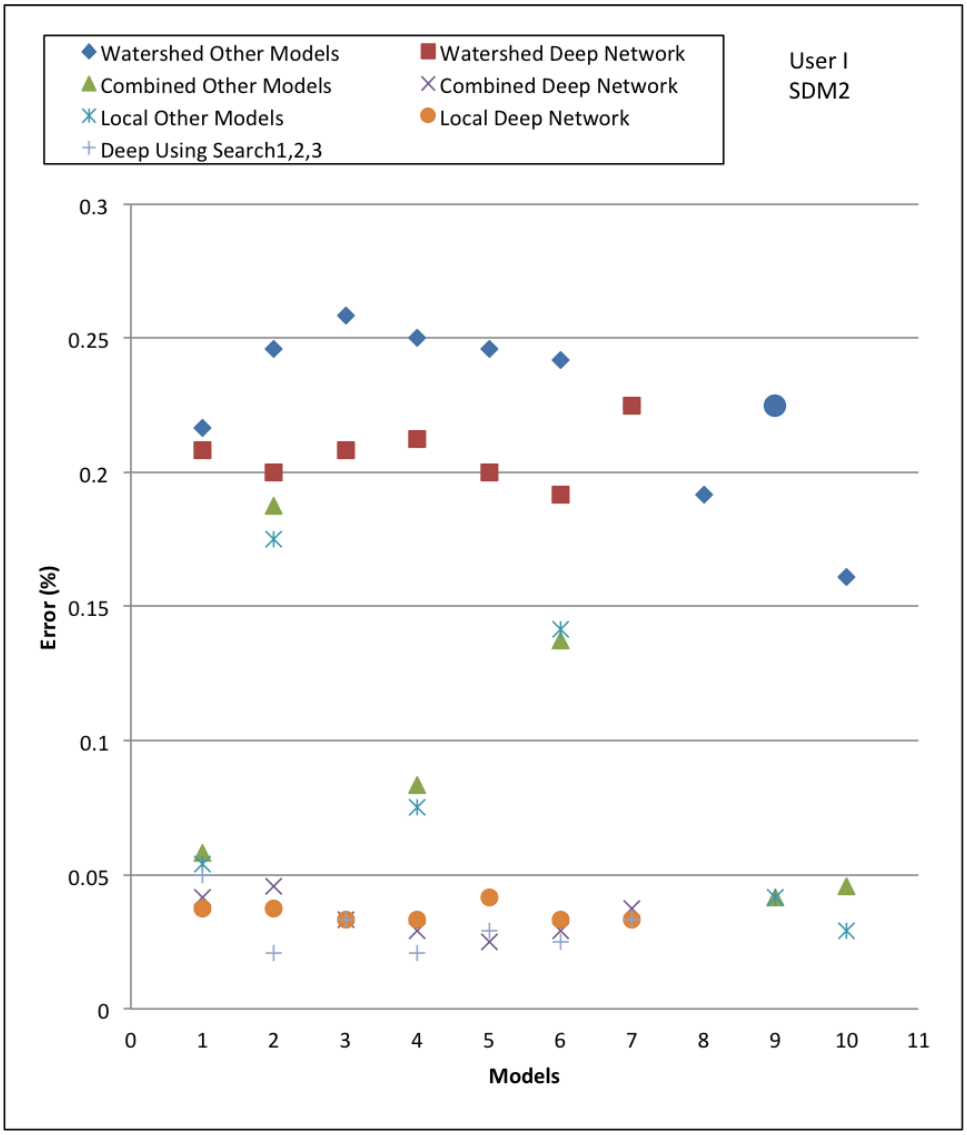

Figure 7.25. Deep Learning Result - User I

For Details about Models Refer Table 7.4.

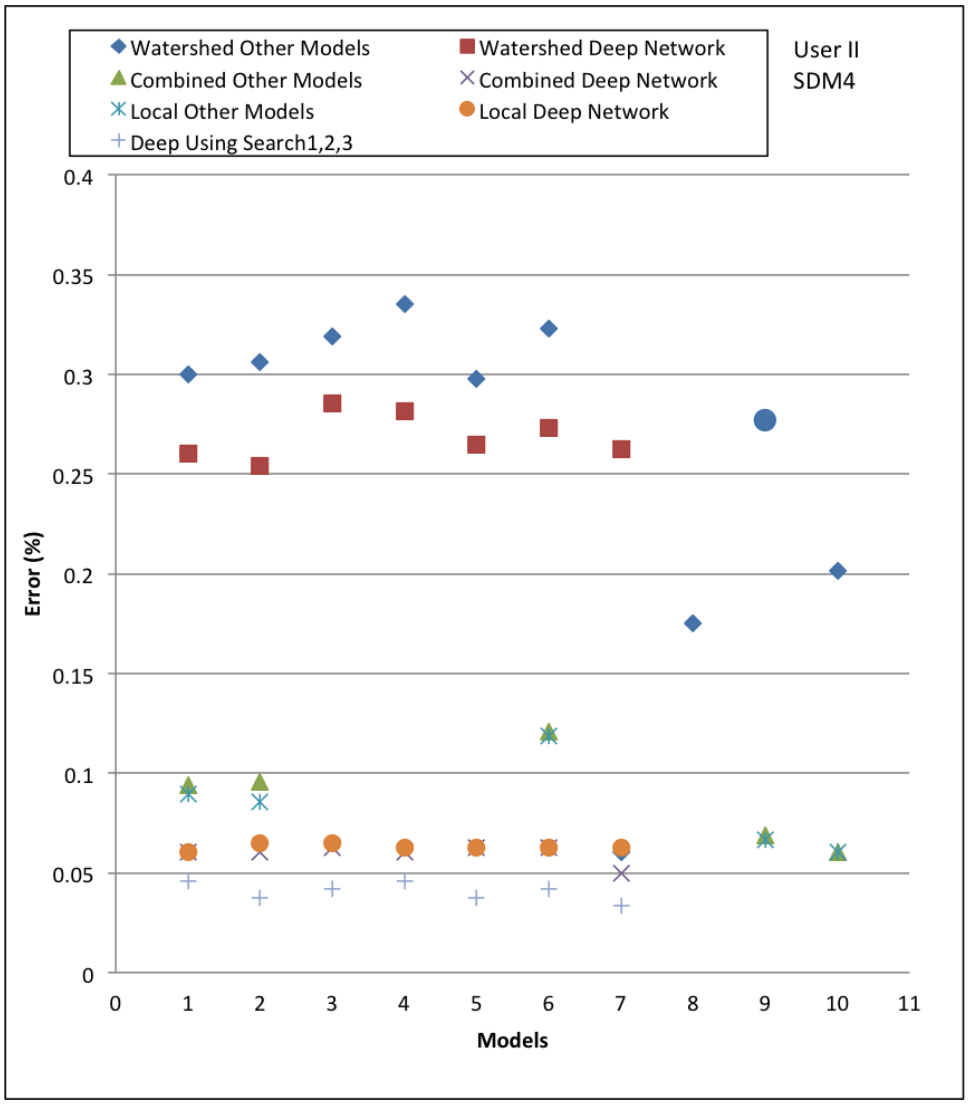

Figure 7.26. Deep Learning Result - User II

For Details about Models Refer Table 7.4.

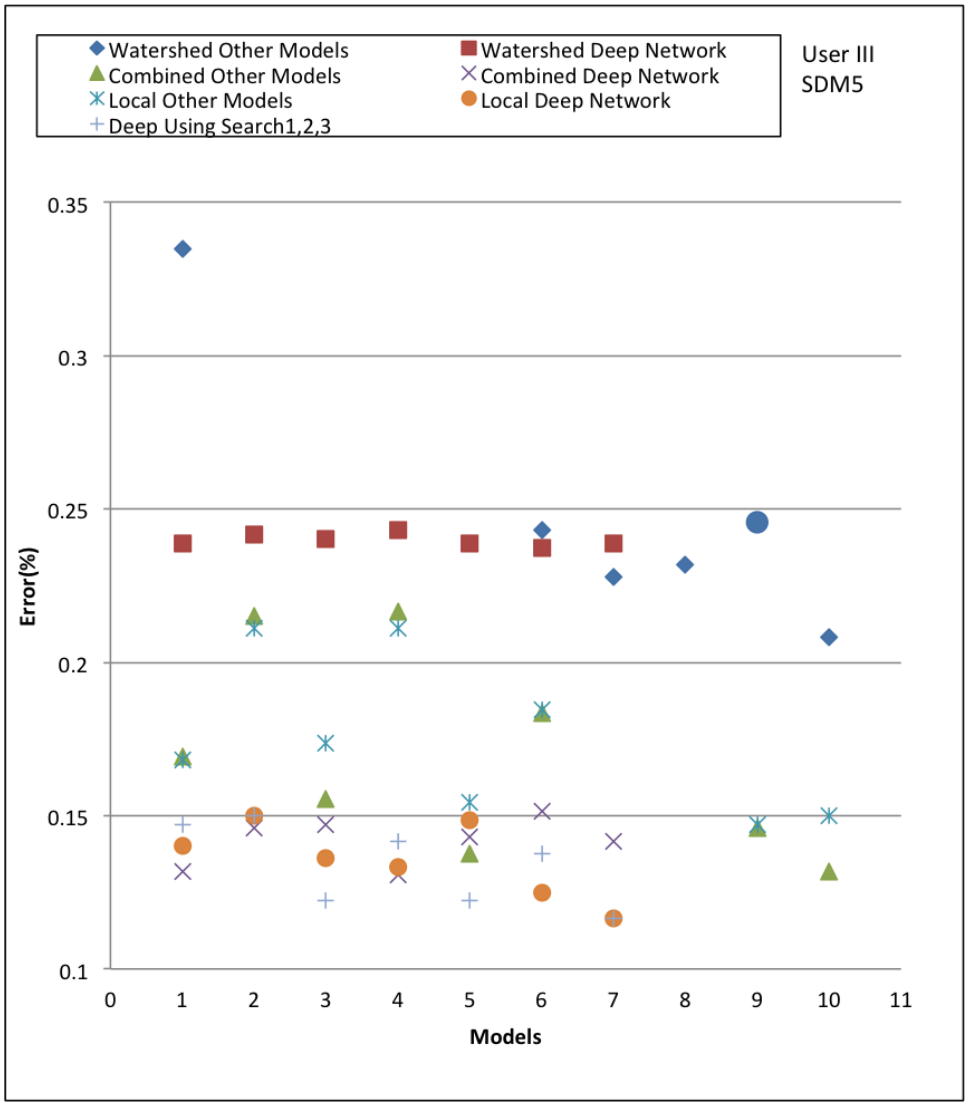

Figure 7.27. Deep Learning Result - User III

For Details about Models Refer Table 7.4.

Effect of Local Decision Making

As we have seen the effect of local decision making in the other models we can see the same pattern in the Deep Network as well.

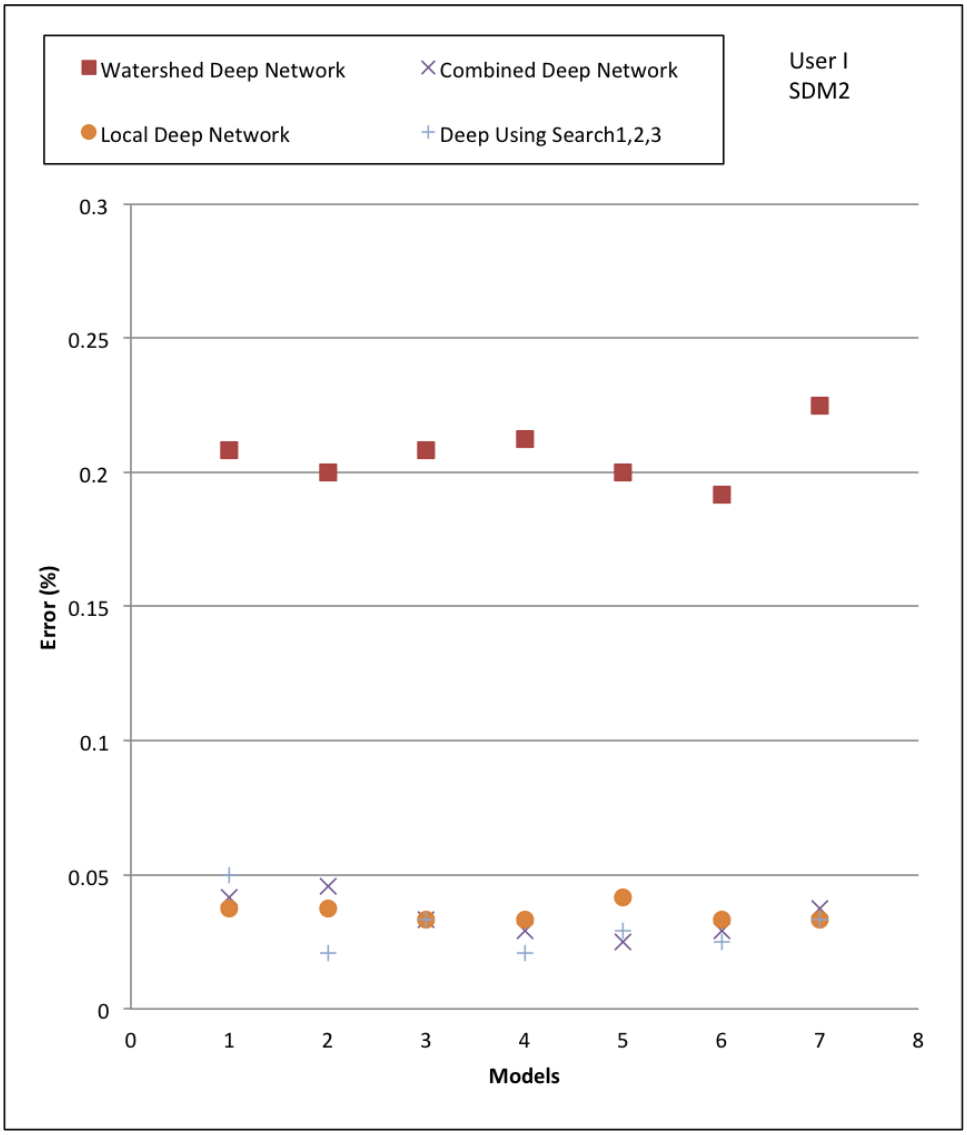

Figure 7.28. Deep Learning local decision Result - User I

As shown in Figures Fig. 7.28, Fig. 7.29, and Fig. 7.30 we can see that the SDM created at the Watershed level is the worst. The SDM created using either the

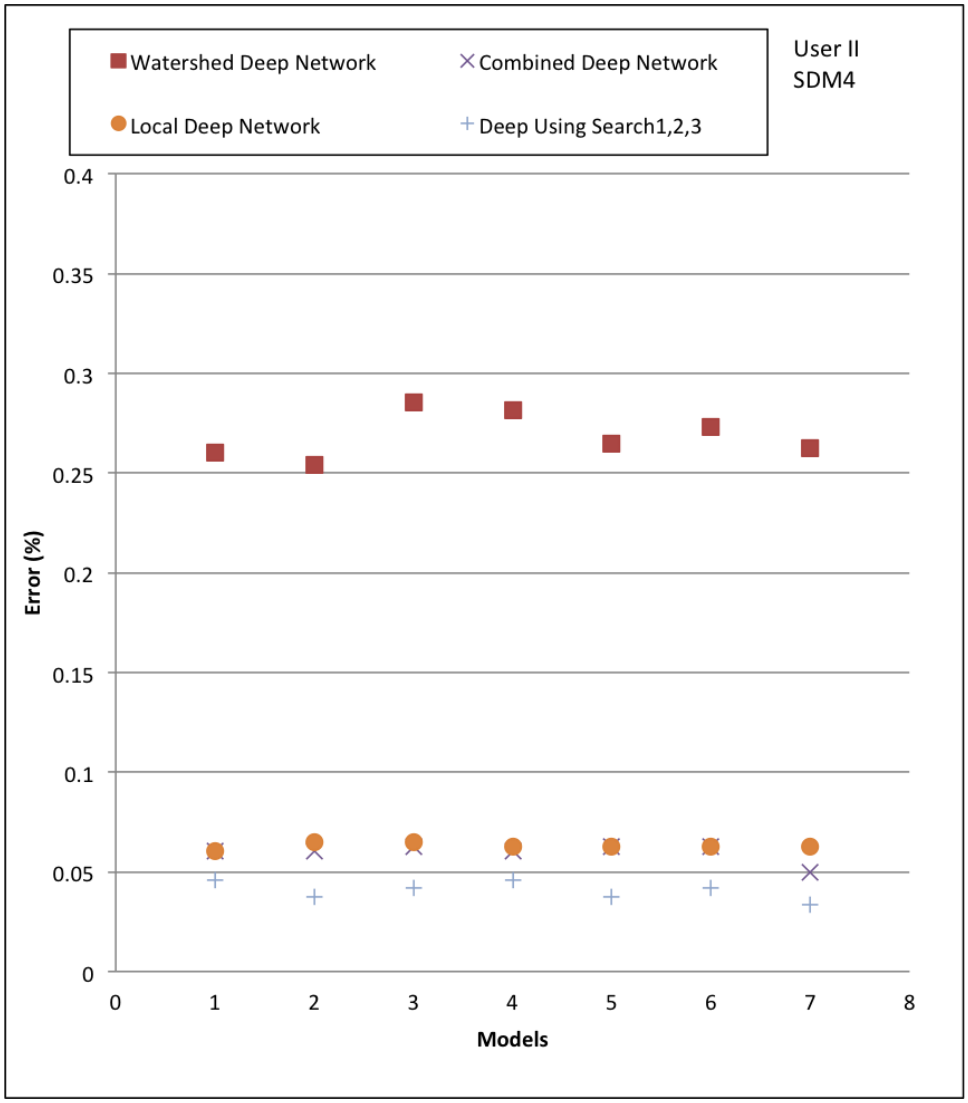

Figure 7.29. Deep Learning local decision Result - User II

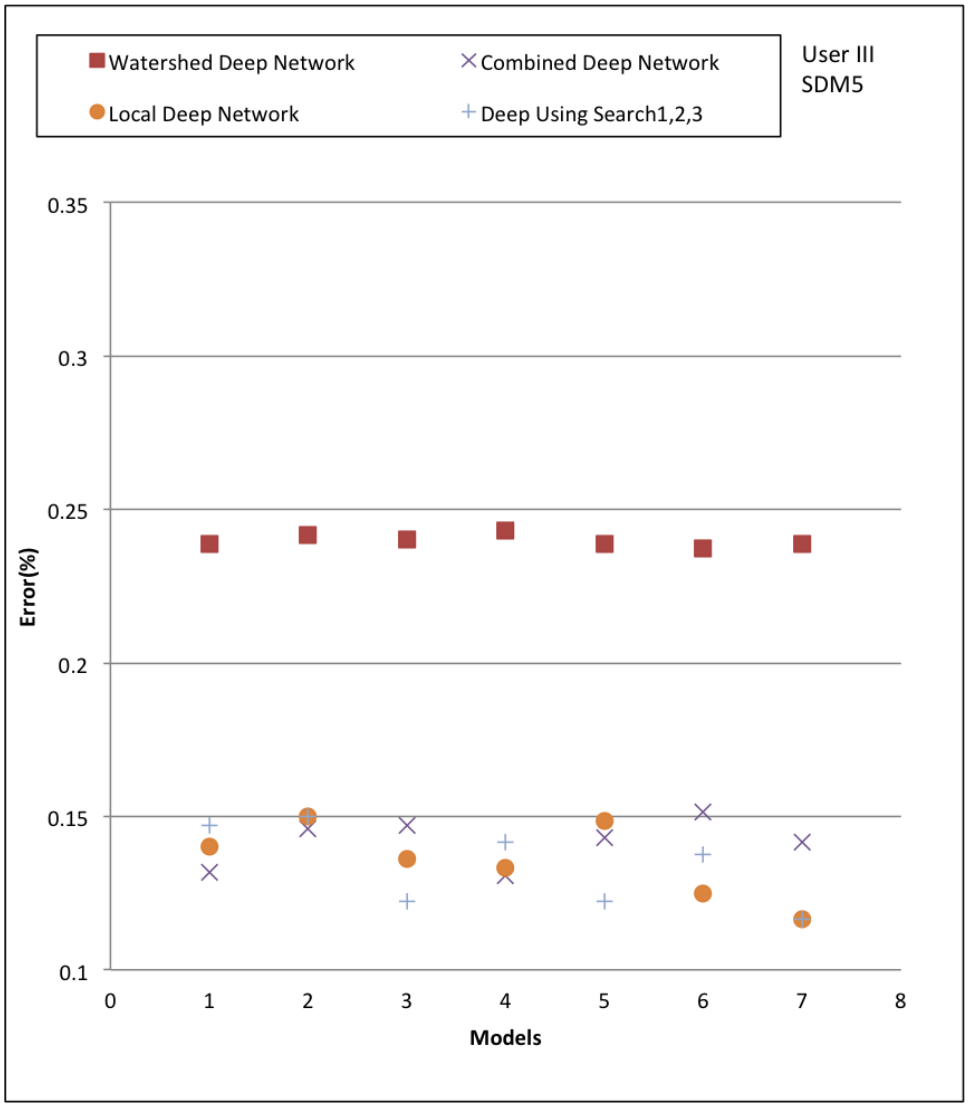

Figure 7.30. Deep Learning local decision Result - User III

combined level or only the local subbasins level are the best way of creating the SDM. The user formulated the rating criteria based on their local subbasins instead of the watershed level.

7.6 Experiments Decision Variable Analysis

In this experiment we tried to find which subbasins a user had used to formulate his/her rating criteria. Every user is assigned a set of subbasins as his/her local subbasins or the entire watershed (subbasin 0). The users formulated their own rating criteria based on the subbasin assigned to them as shown below:

- User I is assigned Subbasins 0, 103, 105, 106, 121 and 122.
- User II is assigned Subbasins 0, 10, 11, 14 and 15.
- User III is assigned Subbasins 0, 58, 59, 61 and 63.
- User IV is assigned Subbasin 0 i.e., watershed only.

A deep network is used to identify the decision variables from all the possible combinations of different subbasins. Three different pairs were used to check which set of subbasins fit perfectly to the user's rating criteria. One big computational challenge was running so many different user modeling programs in parallel. So we used the IU's Big Red II Supercomputer to speedup the work.

7.6.1 Decision Variable - One Subbasin

Only one Subbasin is used to formulate the rating criteria. A deep network is created using the data from only one subbasin. All possible subbasins are tried on Big Red II Supercomputer.

As shown in Table 7.5, we can see that User I mostly formulated the rating criteria using subbasin 106. The rating criteria of User II did not match to any local subbasin.

Table 7.5
Decision Variable - One Subbasin

User Name	SDM	Error(%)	Subbasin
User I	SDM1	0.125	106
User I	SDM2	0.033333333	106
User I	SDM3	0	104
User I	SDM4	0.110416667	106
User I	SDM5	0.094444444	106
User II	SDM1	0.075	31
User II	SDM2	0.045833333	16
User II	SDM3	0.0625	103
User II	SDM4	0.089583333	16
User II	SDM5	0.123611111	17
User III	SDM1	0.158333333	34
User III	SDM2	0.1	63
User III	SDM3	0.054166667	59
User III	SDM4	0.185416667	59
User III	SDM5	0.1625	59
User IV	SDM1	0.1	0
User IV	SDM2	0.016666667	0
User IV	SDM3	0.0125	0
User IV	SDM4	0.072916667	0
User IV	SDM5	0.058333333	0

Table 7.6
Decision Variable - Two Subbasin

User Name	SDM	Error(%)	Subbasin I	Subbasin II
User I	SDM1	0.083333333	17	122
User I	SDM2	0.033333333	3	97
User I	SDM3	0	0	106
User I	SDM4	0.079166667	11	91
User I	SDM5	0.066666667	16	106
User II	SDM1	0.045833333	45	106
User II	SDM2	0.029166667	11	15
User II	SDM2	0.029166667	14	15
User II	SDM3	0.041666667	4	103
User II	SDM3	0.05	10	32
User II	SDM4	0.060416667	17	106
User II	SDM5	0.079166667	10	15
User III	SDM1	0.133333333	97	123
User III	SDM2	0.079166667	38	63
User III	SDM3	0.045833333	3	59
User III	SDM4	0.141666667	47	60
User III	SDM4	0.14375	59	61
User III	SDM5	0.143055556	59	104
User III	SDM5	0.144444444	59	106

User III mostly used subbasin 59. User IV used subbasin 0 i.e. entire watershed to give the feedback, which is accurate as the user was not assigned any other subbasin.

7.6.2 Decision Variable - Two Subbasins

A pair of two subbasins are used to formulate the rating. A deep network is created from the data from a pair of two subbasins. All possible combinations of subbasins are tried on Big Red II Supercomputer.

As shown in Table 7.6, we can see that rating criteria for User I matches to subbasin 0-106. For User II, there are a number of subbasin pairs which match the criteria viz 11-15 and 10-15. So User II has used a pair of subbasins to formulate the rating criteria. For User III, pair 59-61 is the subbasin pair for rating.

7.6.3 Decision Variable - Three Subbasins

A pair of three subbasins are used to formulate the rating. A deep network is created from the data from a pair of three subbasins. All possible combination of subbasins are tried on Big Red II Supercomputer.

As shown in Table 7.7, we can see that rating criteria for User I matches for 0-106-122 subbasins pair. This means that User I has used this pair to formulate the rating. As we can see, for the other users the pairs do not match for their local subbasins. That means the user has not used a pair of three subbasins, but a smaller number of subbasins, to formulate their rating criteria as given in the previous tables.

Overall we can see that the decision variable analysis test shows how effective the user was in formulating their rating criteria for specific set of subbasins. When we asked the users what particular subbasins they were interested in

- User I told us that the user was mostly interested in subbasin 106 to formulate the rating criteria while sometimes subbasin 122 was also used.
- User II told us that the user gave equal consideration to all the subbasins however subbasin 15 played a big role in formulating the rating criteria.
- User III told us that the user formulated the rating criteria based on subbasin 59 as proved in this experiment.

Table 7.7
Decision Variable - Three Subbasin

User Name	SDM	Error(%)	Subbasin I	Subbasin II	Subbasin III
User I	SDM1	0.05	9	34	103
User I	SDM2	0.020833333	0	37	97
User I	SDM2	0.025	1	65	103
User I	SDM3	0	0	106	122
User I	SDM3	0	103	105	106
User I	SDM3	0	103	106	121
User I	SDM4	0.05625	47	106	121
User I	SDM5	0.044444444	32	104	121
User II	SDM1	0.033333333	16	103	125
User II	SDM2	0.016666667	9	32	34
User II	SDM2	0.016666667	14	16	112
User II	SDM3	0.025	10	43	91
User II	SDM3	0.025	14	26	92
User II	SDM4	0.045833333	8	16	31
User II	SDM5	0.056944444	17	37	126
User II	SDM5	0.059722222	2	10	15
User III	SDM1	0.095833333	5	36	91
User III	SDM2	0.058333333	0	20	63
User III	SDM3	0.033333333	30	37	59
User III	SDM3	0.033333333	56	60	63
User III	SDM4	0.10625	17	61	125
User III	SDM5	0.116666667	30	59	123
User III	SDM5	0.119444444	59	64	65

- User IV used entire watershed to formulate the rating criteria because the user was not assigned any other local subbasins and this experiment proves that the user had used entire watershed only to formulate the criteria for rating.

8 FUTURE WORK

During the design and development of the system, I found a number of ideas that can be implemented to perform different tests and make improvement in the system.

8.1 Future Work for Batch Mode Learning

1. To overcome the limitation of using RLBM for a problem in continuous domain, the continuous domain can be transformed to discrete domain. The continuous set of actions is changed to a finite set of discrete sets. In case of Environmental Science the discrete set would not affect much accuracy as the data itself is generated in stochastic manner. Using this approach, the batch mode technique can be used for continuous domain problem.

2. Time based adaptive algorithm can be tried with the Batch Mode. As the learning rate and the batch of designs being selected for learning can vary. An adaptive algorithm can be tried to do the optimization.

3. We can continue the implementation of Batch Mode learning techniques in other different ways. It's multi-objective version can be applied on more than two objectives. Based on availability of computational resources the slicing of weights can be increased to get more accuracy. We can run these stochastic algorithms on powerful machines like Supercomputers or GPUs to attain more precision at very high speed and accuracy.

8.2 Future Work User Modeling

There are a number of user modeling techniques available. But finding the correct user modeling in itself is a hard problem. In this research I mostly tried ANN to model user's rating preferences. But there are many other techniques which can tried. A number of supervised learning techniques like Support Vector Machines, Naive Bayes, decision tree, logistic regression, etc. can be tried.

8.3 Future Work Collaborative Search

Human computer collaborative search, where the human group do the collaboration with computer. A number of collaborative problem solving approaches can be tried. Similarity-based approach and the aggregation function-based approach of multi-criteria rating system can be tried [58].

8.3.1 Different collective judgment approaches are

- Based on crowd wisdom (Independent): Every person is given the same job. The final solution is derived as intersection of all the solutions.
- Based on Collaboration (Citizen Science): The total search space is divided into N (no of people) parts. Each person is assigned one part to search for best set of designs. After everybody solved their parts, the solutions are joined to find the best from all.

8.3.2 Implementation of collective judgment approaches

- Crowd wisdom approach: The same problem given to all humans independently.
  - Democracy Approach: In this approach correct rating for a design in found in a democratic manner. A single search is performed and all the stakeholders work on their terminals simultaneously. The search continues when

all the stakeholders give their feedback and correct rating is determined in a democratic manner. Which can be considered as a majority opinion. The SDM being trained is based on majority opinion.

  - Recommendation Approach: In this approach all the stakeholders run their experiments independently, although they have to run their work simultaneously. In this approach, the stakeholders will communicate with other stakeholders and recommend good designs to other stakeholders for their opinion.

- Based on Collaboration (Citizen science): Different parts of the same problem given to all the users, because the problem is too big for either human or computer or group. Just like in online citizen science projects like zoouniverse, The Milky Way Project [59] where a number volunteer citizens work together to classify galaxies which is impossible even for supercomputers to classify them alone.

  - The final solution cab be obtained by union of all solutions.
  - The final solution can be obtained by finding a global optimal from those solution.

8.3.3 Collaborative Filtering Technique

Just like in information science, filtering technique is one way of discarding unnecessary data, the same technique can be tried here as well. The user is allowed to add constraints based on his/her preferences. The search begins with a large population size, say 100. Although the user is not capable of rating 100 designs, but the filter is used to discard those designs which the user is not interested. In this way more accuracy can be obtained because a large set of population is considered for search.

9 SUMMARY

I would like to conclude this thesis with a short summary of the work I did and some future prospects.

First of all the thesis defines the problem of environmental planning using interactive optimization in which the human users collaborate with computers and they search for better answers. The designing an environmental planning system is very cumbersome work, and there are a number of issues involved in designing such a system.

In this thesis, I discussed what are the challenges we faced and proposed different solutions to solve those challenges. One of the biggest challenge that I faced was running a number of environmental simulations simultaneously. To solve that problem, I worked in making the parallel version of NSGA2 called dist. NSGA2 based on the work of [44]. So that, a number of swat evaluations can be run in parallel on various cluster computers in an efficient manner.

The development of the IGAMI2 System in itself was very hard work. I faced a lot of challenges found in a distributed system as handling data, managing clusters, managing nodes, etc. Managing multi-user data was also not an easy task. I used the open source MySql database to manage data, which gave us a lot of advantages in managing users' data efficiently.

In this thesis, I also described different components of the IGAMI2 distributed system that we developed that can be used to provide Best Management Practices for Environmental Planning.

We created a Batch Mode Learning algorithm called RLBM which is an extension of Decentralized Pursuit Learning Algorithm developed by [4] to implement Batch Mode technique, as commonly used in Neural Network, in the Reinforcement Learning based optimization. The RLBM optimization can be used if a faster optimization is

needed. Although, RLBM works only if the number of actions is discrete and limited. For those problems whose actions are in a continuous domain, some mechanism of transforming actions from continuous to discrete sets would be needed to implement the RLBM.

We also proposed the idea of fine tuning the Artificial Neural Network (ANN) in situations where the number of learning data points are very less, and if some linear or non-linear technique can be used to generate additional data. This helps in increasing the accuracy of ANN in situation of limited data. If the learning data is small and is skewed to some specific rating class, the ANN training becomes biased, and it starts giving the same rating for any input. In such scenario adding a random noise, can prevent the ANN from giving biased ratings. In the experiments, we used the help of other linear/non-linear machine learning techniques to fine tune the ANN. The simulated users experiment and the real stakeholder's experiments both showed promising results.

We tried Deep Learning as an additional machine learning technique in which the Network was pre-trained before training the actual ANN. The pre-training was done using SOM aka Kohonan Map. We showed that use of Deep Learning to model user's rating criteria performs much better than other linear/non-linear, ANFIS and Neural Network machine learning techniques. Deep Learning can be very useful in user modeling especially in the case of Environmental Planning.

We showed that SDM created using the data from local-subbasin worked better than the global as the stakeholders were mostly interested in optimization of local subbasins.

We also found that the stakeholders formulated their rating criteria using a particular subbasin or a pair of subbasins. The experiments showed which stakeholder used what particular subbasin or pair of subbasins to formulate their rating criteria.

LIST OF REFERENCES

LIST OF REFERENCES

[1] Meghna Babbar-Sebens and Barbara S. Minsker. Interactive genetic algorithm with mixed initiative interaction for multi-criteria ground water monitoring design. *Applied Soft Computing*, 12(1):182 – 195, 2012.

[2] J. G. Arnold, R. Srinivasan, R. S. Muttiah, and J. R. Williams. Large area hydrologic modeling and assessment part i: Model development1. *JAWRA Journal of the American Water Resources Association*, 34(1):73–89, 1998.

[3] J. G. Arnold and N. Fohrer. Swat2000: current capabilities and research opportunities in applied watershed modelling. *Hydrological Processes*, 19(3):563–572, 2005.

[4] O. Tilak, M. Babbar-Sebens, and S. Mukhopadhyay. Decentralized and partially decentralized reinforcement learning for designing a distributed wetland system in watersheds. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pages 271–276, 2011.

[5] M. Tory and T. Moller. Human factors in visualization research. *Visualization and Computer Graphics, IEEE Transactions on*, 10(1):72–84, 2004.

[6] H. Takagi. Interactive evolutionary computation: fusion of the capabilities of ec optimization and human evaluation. *Proceedings of the IEEE*, 89(9):1275–1296, 2001.

[7] Marta Herva and Enrique Roca. Review of combined approaches and multi-criteria analysis for corporate environmental evaluation. *Journal of Cleaner Production*, 2012.

[8] Xavier Llorà, Kumara Sastry, David E. Goldberg, Abhimanyu Gupta, and Lalitha Lakshmi. Combating user fatigue in igas: partial ordering, support vector machines, and synthetic fitness. In *Proceedings of the 2005 conference on Genetic and evolutionary computation*, GECCO '05, pages 1363–1370, New York, NY, USA, 2005. ACM.

[9] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, UAI'98, pages 43–52, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[10] V.B. Singh, S. Mukhopadhyay, and M. Babbar-Sebens. Decentralized pursuit learning automata in batch mode. In *Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS), 2012 Joint 6th International Conference on*, pages 1567–1572, 2012.

[11] Vidya Bhushan Singh, Snehasis Mukhopadhyay, and Meghna Babbar-Sebens. User modelling for interactive optimization using neural network. In *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, pages 3288–3293, Oct 2013.

[12] David Anderson, Emily Anderson, Neal Lesh, Joe Marks, Ken Perlin, David Ratajczak, and Kathy Ryall. Human-guided simple search: combining information visualization and heuristic search. In *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management*, NPIVM '99, pages 21–25, New York, NY, USA, 1999. ACM.

[13] Kumpati S Narendra and Mandayam AL Thathachar. *Learning automata: an introduction*. DoverPublications. com, 2012.

[14] O. Tilak, S. Mukhopadhyay, M. Tuceryan, and R. Raje. A novel reinforcement learning framework for sensor subset selection. In *Networking, Sensing and Control (ICNSC), 2010 International Conference on*, pages 95–100, 2010.

[15] J. G. Arnold and N. Fohrer. Swat2000: current capabilities and research opportunities in applied watershed modelling. *Hydrological Processes*, 19(3):563–572, 2005.

[16] T. Heskes and W. Wiegerinck. A theoretical comparison of batch-mode, online, cyclic, and almost-cyclic learning. *Neural Networks, IEEE Transactions on*, 7(4):919–925, 1996.

[17] Steven C. H. Hoi, Rong Jin, and Michael R. Lyu. Large-scale text categorization by batch mode active learning. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 633–642, New York, NY, USA, 2006. ACM.

[18] Steven C. H. Hoi, Rong Jin, Jianke Zhu, and Michael R. Lyu. Batch mode active learning and its application to medical image classification. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 417–424, New York, NY, USA, 2006. ACM.

[19] S.C.H. Hoi, Rong Jin, Jianke Zhu, and M.R. Lyu. Semi-supervised svm batch mode active learning for image retrieval. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7, 2008.

[20] Yuhong Guo and Dale Schuurmans. Discriminative batch mode active learning. In *Advances in neural information processing systems*, pages 593–600, 2007.

[21] M. N. Howell, T. J. Gordon, and F. V. Brandao. Genetic learning automata for function optimization. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 32(6):804–815, 2002.

[22] Masaharu Munetomi, Yoshiaki Takai, and Yoshiharu Sato. Stga: An application of a genetic algorithm to stochastic learning automata. *Systems and Computers in Japan*, 27(10):68–78, 1996.

[23] Feng-Tse Lin, Cheng-Yan Kao, and Ching-Chi Hsu. Applying the genetic approach to simulated annealing in solving some np-hard problems. *Systems, Man and Cybernetics, IEEE Transactions on*, 23(6):1752–1767, 1993.

[24] Alfred Kobsa. User modeling: Recent work, prospects and hazards. *Human Factors in Information Technology*, 10:111–111, 1993.

[25] Q. Chen and A.F. Norcio. A neural network approach for user modeling. In *Systems, Man, and Cybernetics, 1991. 'Decision Aiding for Complex Systems, Conference Proceedings., 1991 IEEE International Conference on*, pages 1429–1434 vol.2, 1991.

[26] Andrew Jennings and Hideyuki Higuchi. A personal news service based on a user model neural network. *IEICE Transactions on Information and Systems*, 75(2):198–209, 1992.

[27] Rui Zou, Wu-Seng Lung, and Jing Wu. An adaptive neural network embedded genetic algorithm approach for inverse water quality modeling. *Water Resources Research*, 43(8):n/a–n/a, 2007.

[28] Ozgur Kisi. Evolutionary fuzzy models for river suspended sediment concentration estimation. *Journal of Hydrology*, 372(14):68 – 79, 2009.

[29] F. Chiari, M. Delhom, J. F Santucci, and J.-B. Filippi. Prediction of the hydrologic behavior of a watershed using artificial neural networks and geographic information systems. In *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, volume 1, pages 382–386 vol.1, 2000.

[30] Gurjeet Singh and Rabindra K Panda. Daily sediment yield modeling with artificial neural network using 10-fold cross validation method: a small agricultural watershed, kapgari, india. *International Journal of Earth Sciences and Engineering*, 4(6):443–450, 2011.

[31] E. Frias-Martinez, G. Magoulas, S. Chen, and R. Macredie. Modeling human behavior in user-adaptive systems: Recent advances using soft computing techniques. *Expert Systems with Applications*, 29(2):320 – 329, 2005.

[32] Hong-Zhong Huang, Zhigang Tian, and Ming J. Zuo. Intelligent interactive multiobjective optimization method and its application to reliability optimization. *IIE Transactions*, 37(11):983–993, 2005.

[33] Ingrid Zukerman and DavidW. Albrecht. Predictive statistical models for user modeling. *User Modeling and User-Adapted Interaction*, 11(1-2):5–18, 2001.

[34] Christos Papatheodorou. Machine learning in user modeling. In Georgios Paliouras, Vangelis Karkaletsis, and ConstantineD. Spyropoulos, editors, *Machine Learning and Its Applications*, volume 2049 of *Lecture Notes in Computer Science*, pages 286–294. Springer Berlin Heidelberg, 2001.

[35] Alfred Kobsa. Generic user modeling systems. *User Modeling and User-Adapted Interaction*, 11(1-2):49–63, 2001.

[36] Judy Kay. The um toolkit for cooperative user modelling. *User Modeling and User-Adapted Interaction*, 4(3):149–196, 1994.

[37] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2014/01/08 2006.

[38] David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147 – 169, 1985.

[39] Rajat Raina, Anand Madhavan, and Andrew Y Ng. Large-scale deep unsupervised learning using graphics processors. In *ICML*, volume 9, pages 873–880, 2009.

[40] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.

[41] A. Samuels and M. Babbar-Sebens. Field scale optimization for longterm sustainability of best management practices in watersheds, 2012.

[42] M. Thathachar and M. T. Arvind. Parallel algorithms for modules of learning automata. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 28(1):24–33, 1998.

[43] M. Babbar-Sebens and S. Mukhopadhyay. Reinforcement learning for human-machine collaborative optimization: Application in ground water monitoring. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 3563–3568, 2009.

[44] J.J. Durillo, A.J. Nebro, F. Luna, and E. Alba. A study of master-slave approaches to parallelize nsga-ii. In *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, pages 1–8, 2008.

[45] Balázs Csanád Csáji. Approximation with artificial neural networks. *Faculty of Sciences, Etvs Lornd University, Hungary*, page 24, 2001.

[46] Hava T. Siegelmann and Eduardo D. Sontag. Turing computability with neural nets. *Applied Mathematics Letters*, 4(6):77 – 80, 1991.

[47] Jane M. Binner *, Rakesh K. Bissoondeeal, Thomas Elger, Alicia M. Gazely, and Andrew W. Mullineux. A comparison of linear forecasting models and neural networks: an application to euro inflation and euro divisia. *Applied Economics*, 37(6):665–680, 2005.

[48] Hsiao-Tien Pao. A comparison of neural network and multiple regression analysis in modeling capital structure. *Expert Systems with Applications*, 35(3):720 – 727, 2008.

[49] R. Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural Networks, 1989. IJCNN., International Joint Conference on*, pages 593–605 vol.1, 1989.

[50] Wolfgang Pohl. Learning about the user - user modeling and machine learning. In *ICML'96 Workshop Machine Learning meets Human-Computer Interaction*, pages 29–40, 1996.

[51] Stefano Nolfi and Domenico Parisi. Learning to adapt to changing environments in evolving neural networks. *Adaptive behavior*, 5(1):75–98, 1996.

[52] Jyh-Shing R. Jang. Fuzzy modeling using generalized neural networks and kalman filter algorithm. In *Proceedings of the ninth National conference on Artificial intelligence - Volume 2*, AAAI'91, pages 762–767. AAAI Press, 1991.

[53] J.-S.R. Jang. Anfis: adaptive-network-based fuzzy inference system. *Systems, Man and Cybernetics, IEEE Transactions on*, 23(3):665–685, 1993.

[54] Chia-Feng Juang and Chin-Teng Lin. An online self-constructing neural fuzzy inference network and its applications. *Fuzzy Systems, IEEE Transactions on*, 6(1):12–32, 1998.

[55] Chia-Feng Juang, Teng-Chang Chen, and Wei-Yuan Cheng. Speedup of implementing fuzzy neural networks with high-dimensional inputs through parallel processing on graphic processing units. *Fuzzy Systems, IEEE Transactions on*, 19(4):717–728, 2011.

[56] Yoshua Bengio. Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009.

[57] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[58] Gediminas Adomavicius, Nikos Manouselis, and YoungOk Kwon. Multi-criteria recommender systems. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, pages 769–803. Springer US, 2011.

[59] R. J. Simpson, M. S. Povich, S. Kendrew, C. J. Lintott, E. Bressert, K. Arvidsson, C. Cyganowski, S. Maddison, K. Schawinski, R. Sherman, A. M. Smith, and G. Wolf-Chase. The milky way project first data release: a bubblier galactic disc. *Monthly Notices of the Royal Astronomical Society*, 424(4):2442–2460, 2012.