Proceedings of the Fifteenth Midwest Artificial Intelligence and Cognitive Science Conference MAICS 2004

Roosevelt University
School of Computer Science and Telecommunications
Albert A. Robin Campus
Schaumburg, Illinois
April 17-18, 2004

Eric G. Berkowitz, Editor

MAICS 2004 Program Committee

Conference Chair and Program Chair: Eric Berkowitz *Roosevelt University*

Committee members:
Hani Abu-Salem *DePaul University*
Raj Bhatnagar *University of Cincinnati*
Douglas S. Blank *Bryn Mawr College*
Doina Caragea *Iowa State University*
Sumali Conlon *University of Mississippi*
Michael T. Cox *Wright State University*
Scott A. DeLoach *Kansas State University*
Martha W. Evens *Illinois Institute of Technology*
Michael Glass *Valparaiso University*
Art Graesser *University of Memphis*
Vasant Honavar *Iowa State University*
Boris Kerkez *Wright State University*
Beomjin Kim *Indiana University-Purdue University*
Jung Hee Kim *North Carolina A&T State University*
Yeongkwun Kim *Western Illinois University*
Dimitris Margaritis *Iowa State University*
Frank W. Moore *Miami University*
Patrick Paulson *Miami University*
Anca Ralescu *University of Cincinnati*
Jennifer Seitzer *University of Dayton*
John Schlipf *University of Cincinnati*
Adrian Silvescu *Iowa State University*
Robert Stufflebeam *University of New Orleans*
Jin Tian *Iowa State University*
Gregory D. Weber *Indiana University East*
Dawn Wilkins *University of Mississippi*

Acknowledgments:

I would like to thank the committee for the reviews they performed. Each paper was reviewed by at least two reviewers, many at the last minute. I am truly grateful for the assistance and responsiveness in getting reviews back to all of the authors

who submitted papers to the conference.

I would like express my sincere gratitude to Jennifer Costello and Zita Ceponis for their efforts in planning this conference. Zita and Jennifer took care of many of the arrangements that needed to be made to make the conference a success. I would like to thank Mr. Ray Wright, Director of the School of Computer Science and Telecommunications for the School's sponsorship of the conference and for making the resources and administrative support of the school available to me. I also want to thank Acting Provost, Dr. Lynn Weiner, and Dr. Steve Cohen, Acting Dean of the College of Arts and Sciences for their sponsorship and support in hosting this conference at Roosevelt University's Robin Campus. Special thanks also go out to Prof. Martha Evens whose advice, insight, and reassurances were, as always, invaluable. Lastly, I would like to state that any errors or omissions are solely my responsibility.

Sponsorship:
I would like to thank the Roosevelt University College of Arts and Sciences and the School of Computer Science and Telecommunications for their sponsorship of MAICS 2004.

Conference Schedule:

Saturday April 17, 2004

8:00-9:00 Breakfast
9:00-9:15 Welcome
10:00-11:00 Invited Speakers Alex Wolpert and Evgeny Dantsin *Satisfiability – Testing Algorithms and Bounds on Their Running Time*

11:00-11:15 Coffee break

Natural Language and Reasoning - Chair: Martha W. Evens

11:15-11:45 **Karen Ehrlich** Mathematics and Computer Science Department, SUNY College at Fredonia *Default Reasoning Using Monotonic Logic: Nutter's modest proposal revisited, revised and implemented*
11:45-12:15 **Chung Hee Lee and Martha W. Evens** Illinois Institute of Technology *Using Selectional Restrictions to parse and Interpret Student Answers in a Cardiovascular Tutoring System*
12:15-12:45 **Paul Bucheit** Harrold Washington College *Adding NL Technology to an Online Language Instruction Tool*

1:00-2:00 Lunch at the Hyatt

Genetic and Evolutionary Algorithms Chair: Jennifer Seitzer

2:15-2:45 **Mouin Hourani, Mufit Ozden, Frank Moore, and Pedrito Maynard-Zhang** Computer Science and Systems Analysis Department, Miami University *Genetic Algorithm Application to Clustering Problems*
2:45-3:15 **Zachary V. Hendershot** Computer Science and Systems Analysis Department, Miami University *A Differential Evolution Algorithm for Automatically Discovering Multiple Global Optima in Multidimensional, Discontinuous Spaces*

3:15-3:45 **Dana Vrajitoru** Intelligent Systems Laboratory, Indiana University South Bend *Intra and Extra-Generational Schemes for Combining Crossover Operators*
3:45-4:15 **Patrick Berarducci, Demetrius Jordan, David Martin, and Jennifer Seitzer** Computer Science Department, University of Dayton *GEVOSH: Using Grammatical Evolution to Generate Hashing Functions*

4:15-4:45 Coffee Break

Agents Chair: Michael T. Cox

4:45-5:15 **Adam C. Langdon and Michael T. Cox** Department of Computer Science and Engineering, Wright State College *The Effect of Agent Topologies on Multiagent Planning Performace*
5:15- 5:45 **Kevin Berridge, Benjamin Lee, Andrew Schworer, and Jennifer Seitzer** Learning with Reasoning Research Group, University of Dayton *Virus Fighting with Mobile Autonomous Agents*

6:00 - 8:00 Dinner at the Hyatt

Sunday April 18, 2004

8:00-9:00 Breakfast

Indexing and Classification Chair: Michael Glass

9:00-9:30 **Dan Vance and Anca L. Ralescu** ECECS Department, University of Cincinnati *Sifting the Local Margins -An Iterative Emperical Classification Scheme-*
9:30-10:00 **Eric G. Berkowitz, Mohamed Reda Elkhadiri, Tim Sahouri and Michel Abraham** School of Computer Science and Telecommunications, Roosevelt University *Intelligent Content Based Title and Author Name Extraction from Formatted Documents*
10:00-10:30 **Louis Green and Injoo Jeong** East-West University**, Yeongkwun Kim** Western Illinois University **and Martha W. Evens** Illinois Institute of Technology *Using an Ontology to Improve the Relevance of Information Retrieval Results*

10:30-11:00 **Eric G. Berkowitz and Mohamed Reda Elkhadiri** School of Computer Science and Telecommunications, Roosevelt University *Creation of a Style Independent Intelligent Autonomous Citation Indexer to Support Academic Research*

11:00-11:15 Coffee Break

Algorithms Chair: Frank Moore

11:15-11:45 **Kasthurirangan Gopalakrishnan and Anshu Manik** Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champagne *ILLI-PAVE Based Pavement Moduli Backcalculation Using Artificial Neural Networks*
11:45-12:15 **Sofia Visa and Anca Ralescu** ECECS Department, University of Cincinnati *Experiments in Guided Class Rebalance Based on Class Structure*
12:15-12:45 **Anshu Manik, Abhishek Singh and Shengquan Yan** Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champagne *Developing Surrogate Simulation Models for a Computationally Intensive Real World Application*

1:00-2:00 Lunch at the Hyatt

Learning and Planning Chair: Beomjin Kim

2:00-2:30 **Kevin Cleereman and Michael T. Cox** Department of Computer Science and Engineering, Wright State University *Linear Inequality Control Rules in State-Space Planning: Beyond the first order predicate calculus*
2:30-3:00 **Vasile Rus** Intelligent Systems Laboratory, Department of Computer and Information Sciences, Indiana University *Experiments with Machine Learning for Logic Arguments Identification*

3:00-3:15 Coffee Break

Image Recognition Chair: Anca Relescu

3:15-3:45 **Waibhav Tembe and Anca Ralescu** Machine Learning and

Computational Intelligence Laboratory, Department of ECECS, University of Cincinnati *A Context-dependent Metric for Similarity -Application to Supervised Learning of Handwritten Characters-*

3:45-4:15 **Jing Xie and Beomjin Kim** Department of Computer Science Indiana University–Purdue University *Templating Optimal Orthopedic Implant Using a Decision-Support System*

4:15- 4:45 **Mircea M. Ionescu and Anca L. Ralescu** ECECS Department, University of Cincinnati *The Impact of Image Partition Granularity Using Fuzzy Hamming Distance as Image Similarity Measure*

4:45-5:15 **Prasanna Karunanayaka and Anca Ralescu** ECES Department, University of Cincinnati *Directional Position of Objects in Image Understanding - a fuzzy landscape approach-*

5:15 Conference Ends

Table of Contents

Virus Fighting with Mobile Autonomous Agents

Kevin Berridge, Benjamin Lee, Andrew Schworer, Jennifer Seitzer

University of Dayton Learning with Reason Research Group
Dayton, OH 45469
Email: {berridkw, leebenjp, schworap, seitzer}@notes.udayton.edu

Abstract

In this work we examine mobile autonomous agent technology and its ability to effectively fight a computer virus attack on a network. A mobile agent is an autonomous agent that has the ability to transfer and reproduce itself on other computers within a network [Kotz, 99]. A computer virus is a malicious computer program that infects host systems and replicates itself to other host systems [Hoffman,90]. Using mobile agents, we simulate a virus attack on a small, isolated network test-bed. We then simulate a virus fighter that successfully detects the attack and distributes a patch across the network. We employ the Java Aglet environment in our implementation of mobile agents [Lange, 98]. In this paper, we present both virus propagation and virus detection algorithms, simulation details, a discussion of the aglet test environment architecture, and detailed analysis of test infection results.

Introduction

Hoffman quotes a definition by John Inglis which states the accepted definition of a computer virus:

"He defines a virus as a piece of code with two characteristics:

1. At least a partially automated capability to reproduce.
2. A method of transfer which is dependent on its ability to attach itself to other computer entities (programs, disk sectors, data files, etc.) that move between these systems."[Hoffman, 90]

Thus, the abilities to automatically reproduce and move to other hosts are insidious properties of all viruses. This ability to transfer and reproduce manifests as the rapid spread of a virus through a computer network. A virus tends to be inefficient and wasteful of host computer and network resources. The more inefficiently the virus uses the network the more damage the virus will do to the target. Damage done to a target can be measured in network bandwidth used by a virus' infection. Therefore, the greater inefficiency with which a virus uses network resources the greater amount of bandwidth will be used and consequently more damage will be done.

The properties of reproduction and transferability are not unique to viruses. Agents, namely mobile autonomous agents, also have these capabilities. An agent can be defined as an entity that can receive percepts through sensors from its environment, and perform actions on its environment through affecters [Russell, 03]. An autonomous agent is a species of agent that senses its environment and acts in pursuit of its own agenda and so as to affect what it senses in the future [Franklin, 96]. A mobile agent is yet another species of agent that is capable of transporting itself from one machine to another [Kotz, 99]. The Java Aglet implementation of mobile agents adds the ability to clone itself as well as transfer itself to another machine [Lange, 98]. Therefore, the combination of these two agents yields a mobile autonomous agent and is capable of behaving much like a virus because it possesses the capability to autonomously reproduce and transfer itself to other machines.

In this paper, we describe our work that merges these two entities by using one to simulate the other. Our system uses mobile agents to emulate both a propagating virus and a self-directed virus detecting and fighting agent. We hypothesized that since a typical virus is inefficient, an efficient mobile autonomous agent has an inherent advantage over the virus. That is, once a virus is detected, the fighting agent could quickly repair all infected nodes as well as inoculate those yet to be visited by the virus. The mobile autonomous agent, or virus fighter, should be able to act much like a virus in that it can reproduce and transfer itself, but instead of infecting a host it could inoculate it. This research seeks to validate this claim by using mobile autonomous agents to simulate a virus infection on a network and use mobile autonomous agents to fight the infection in real-time.

Environment

Java Aglets are used as the mobile agent architecture of choice. Aglets were developed by IBM Japan, and the software development kit is available free in open source [Lange, 98]. Each ***aglet*** is a mobile agent. Our test-bed system consisted of five computers connected on a 100mbps network with an Ethernet hub. This network topology is typical of any small office or home network. Each computer in the network was installed with the latest

version of the aglet runtime environment, Tahiti v2.0.2 [Aglets.org, 03]. Running Tahiti environment on every computer allowed for the creation, movement, disposal, and reproduction of the aglets. Thus, this environment allowed the mobile agents to behave like viruses.

Viruses and virus fighters are each represented as individual aglets. These aglets then have the ability to clone (reproduce) and dispatch (transfer) to other nodes. Tahiti also provides the ability to send messages from one aglet to another, or to broadcast a message to all aglets on a node. We use messaging capability with both the viruses and the virus fighter.

For our viruses to traverse a network, it must be able to determine the presence of all adjacent nodes. Knowing the adjacent nodes, allows the virus to infect the entire network. To simplify this task for the network simulation, text files are used on every machine that explicitly denote adjacencies to the host node. Moreover, by using this text file architecture, revising node adjacencies becomes trivial. This gives the simulated network the ability to be reconfigured into many different network topologies very quickly and easily, and allow for testing of the virus fighting algorithm on multiple network configurations. In addition, the ability to run multiple instances of Tahiti on one machine, each operating on a different port, allows for further expansion of the original five node network.

Virus Algorithm

Our propagation algorithm used for the virus mobile autonomous agent is simple and was designed to emulate the inefficient nature of a virus as it infects a network to which we alluded in the introduction. To begin an infection one virus is created, and then manually dispatched, using Tahiti, to infect a target node. Upon the virus entering the node, the virus announces its arrival and waits a set duration for a reply.

A reply could come from the virus when it is in one of two different stages of its life. If another instance of the virus was currently infecting that node, that instance, already a resident of the host, would tell the arriving virus to die. The arriving virus would comply and die. This method of arrival was done to keep the amount of viruses on one node at a minimum to keep from overloading the fragile Tahiti environment. The second type of reply would come from a virus that had been cleaned by a virus fighter. This mode of the virus now allows the virus to act as a simulated patch on the node. The simulated patch virus would also tell any arriving virus to die.

If the arriving virus did not receive any replies to its arrival announcement it could assume that the node is neither infected nor cleaned. Consequently, the virus would begin to reproduce and send the infections to all adjacent nodes. A virus clone would even be sent back to the node from which the original virus had just come. After it had sent out viruses to all of its adjacent nodes, it would wait for a brief period of time, and then repeat the process. As alluded to earlier, when a virus fighter sends a clean message to a virus, the virus immediately stops sending out infections, and acts as a patch on the previously infected node. In general, again, this virus infection algorithm was designed to be inefficient and to be the most caustic to the environment and the network.

Virus Fighter Algorithm

The algorithm used by the virus fighters is based on the breadth first search algorithm for finding the shortest path to a goal in a graph without weights. One implementation of the breadth first search algorithm uses a table which keeps track of the nodes visited, whether the node is known, the distance of the node from the start node denoted d_v, and the parent of the node denoted p_v.[Weiss, 99]. Table 1 shows an example of what this table might look like after the algorithm has executed assuming five nodes in the network.

v	known	d_v	p_v
1	T	0	0
2	T	1	1
3	T	2	2
4	T	3	3
5	T	3	3

Table 1: Breadth First Search Table

This same table structure was used in the virus fighters as a way to guarantee that the virus fighters would always move through the entire graph. When a virus fighter discovers a node, it adds it to its breadth table. If the node has already been found it simply updates that node's data in the table. The Virus fighter marks a node known in the table after it has visited that node. The Virus fighter determines what node to visit next by finding the first unknown node in the breadth table. Using this strategy, the virus fighter is guaranteed to visit every connected node in the graph exactly once. When the virus fighter has visited all the nodes in the graph and there are no more unknown nodes in its breadth table, it starts over, at which ever node happens to be its current location, by deleting all the data in its table and starting as if it had just been created for the first time.

To this algorithm the concept of cloning was added. If, during a virus fighter's traversal, it discovers a virus on a node, it immediately clones to all of that node's children which it hasn't already visited and marked known

in its breadth table. Each clone is sent to the child node with a modified breadth table from its parent virus fighter. This modified table includes the destination node and the path from the start node to that destination node. Passing this table prevents the clones from moving in cycles and from moving back along paths that their parent virus fighter already traveled. Meanwhile the parent node marks the nodes to which it has sent clones as known in its own breadth table and continues to search. Lastly, like the virus agent, if a virus fighter agent finds another virus fighter at a node it is visiting, it dies.

To summarize the general behavior of this algorithm, we note that when a virus fighter is released on a graph it will move to every node in the graph continually searching for the presence of any viruses. When it finds a virus it will clone itself causing the number of virus fighters in the graph to increase if a virus is present. Then if the virus fighter encounters another virus fighter it will die, causing the number of virus fighters to diminish as they encounter each other in the network. This means that a single virus fighter moving through a network will dynamically spawn many virus fighters to combat viruses and then the many virus fighters will die off until there are just a few left.

Data Collection

In order to analyze the effectiveness and life cycles of both the viruses as well as the virus fighters we collected large amounts of data during each trial. Data collection was essential due to the mobile nature of the aglets. Without detailed data collection, analysis of agent movement and virus fighting effectiveness would have been impossible ascertain. Furthermore, to understand the effectiveness of different virus fighting techniques, two independent data sets were collected for every graph. One data set represented a trial where a single virus fighter was introduced into the environment and allowed to traverse the network once, before a single virus was introduced to a random node. This was described as the "virus-fighter-first" condition. The other data set had viruses being introduced first, "virus-first" condition, and allowed to propagate and infect all nodes before a virus fighter was released to fix the infection. This was done to mimic current virus removal methods that are not invoked until a network is already experiencing massive problems due to virus infections.

In order to collect movement and creation/death statistics of all aglets on the system we created a centralized logging server. The logging software was created from the ground up in Java and built to handle numerous simultaneous connections. Each connection from either a virus or virus fighter consisted of the agent's unique id, a message code, the DNS name of the machine on which it was currently running, and the local time and date. This information was recorded into a MYSQL relational database located on the logging machine [Kortenkamp, 04]. After each trial was completed the data in the database was run against a compilation script that executed SQL statements to formulate and compile many statistics on a trial by trial basis. This data was useful in determining the number of virus fighters and/or viruses created over the course of a trial and the relative infection times. Examples of data collected using this method can be found in table 2.

Network usage statistics were gathered using *tcpstat* on an independent machine running RedHat Linux. Data was collected during one-second intervals gathering data on the number of packets broadcast, the total bytes of data broadcast, and the average packet size. This data was used primarily in our graph creations and demonstrated the evolution of the trials and the difference in network usage from one starting condition to another. Upon running each trial, *tcpstat* was started at the same time as the simulating environment was initialized and was not closed until normal bandwidth use was achieved once again.

Data

Data was collected using five network graphs, each testing different aspects of the virus fighters and the viruses as well as different graph structures. Figure 1 shows each of these graphs in visual form. Each circle represents a node or computer on our simulated network. Each node acts as an independent entity which may or may not contain any number of viruses or virus fighters at a given moment. The bold circles represent the start node from which virus fighters were released. The edges of these graphs represent bidirectional connections between the nodes of the simulated network. This means that viruses and virus fighters can travel in either direction along these paths. As was stated previously, data was collected on network usage as well as numerical and time based data on virus fighter and virus activity.

Graph	Virus fighters Created	Viruses Created	#Virus fighters left alive
1-VF	4	27	2
1-Virus	5	65	1
2-VF	9	33	1
2-Virus	14	55	2
3-VF	14	22	2
3-Virus	18	67	2
4-VF	4	28	2
4-Virus	5	99	3
5-VF	18	145	2
5-Virus	17	395	1

Table 2: logging data

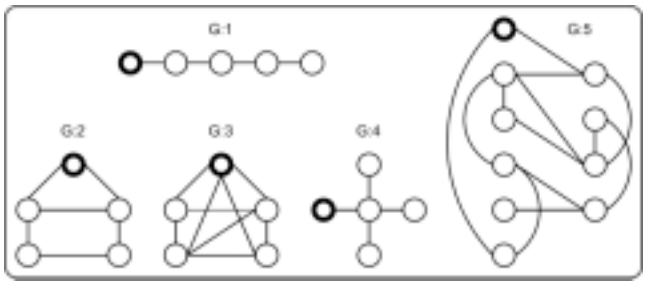


Figure 1: Graph Layouts

Graph 1

This graph was the simplest tested, consisting of a straight line with only one connection between the nodes. The purpose of this graph was to demonstrate how the virus fighter's and virus' efficiency were equally lowered when navigating such a simple and relatively unconnected graph. The cloning nature of both the viruses and the virus fighters is negated by the lack of connecting edges, causing them to only be able to move linearly.

Figure 2 shows the packets vs. time graphs of both the virus-fighter-first and virus-first trials. As was predicted, the peak packet intervals were similar and did not give a distinct advantage to either technique. Slight but distinct differences were found in the *infection time*, the time period between when the first computer was infected and the last virus was cleaned, among the trials. While the virus-fighter-first trial seems to spike with a high rate of climb and then quickly fall off again to normal load, the virus-first approach gave a longer infection time which rose and fell less suddenly. The higher normal load found after the virus fighter spike is attributed to the fact that this particular graph allowed two virus fighters to live on after the virus infection in the virus-fighter-first condition as seen in Table 2. The packet figures for Graph 1 were only slightly lower than other trials.

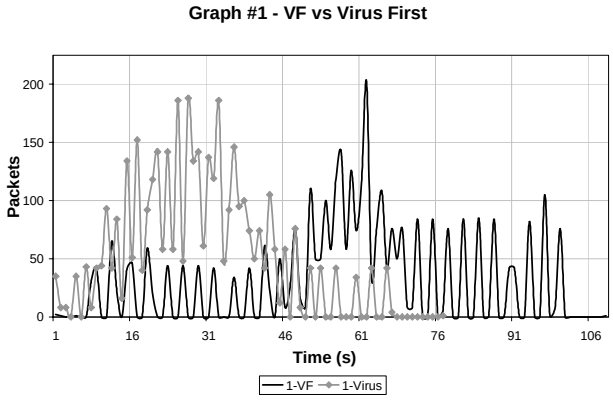


Figure 2: Graph 3 Network Usage

Peak network usage was similar between the two trials as we believed would happen given the graph design. Significantly fewer virus fighters were created under this graph as compared to others in order to handle the virus infection under both start conditions. This can be attributed to the inability of the virus fighters to clone to many nodes at once. As was suspected, the number of viruses created under the virus-first condition was significantly greater than the virus-fighter-first condition.

It should be noted that network load statistics were gathered per second across the network and that average idle load, that is load without intentional network usage, was found to be only one or two packets per second. When normal testing load was reached, consistent dips in usage can be seen. This is due to time periods where virus fighters are analyzing their current node and not in transit. During these periods, network load drops to its idle levels. Finally, offsets in graph spikes were due to the fact that we allowed the virus fighter to traverse the entire graph once before introducing the viruses to ensure that the entire graph was being covered. On the contrary, during virus-first trials, we immediately introduced viruses to the system which caused their network usage graphs to begin much sooner.

Graph 2

Graph 2 was designed to be a small semi-connected graph that would allow for more cloning than Graph 1 but was not fully connected. This graph is more representative of a medium network configuration. Here the multiple paths allow both the virus fighters and viruses more access to the entire graph from each node.

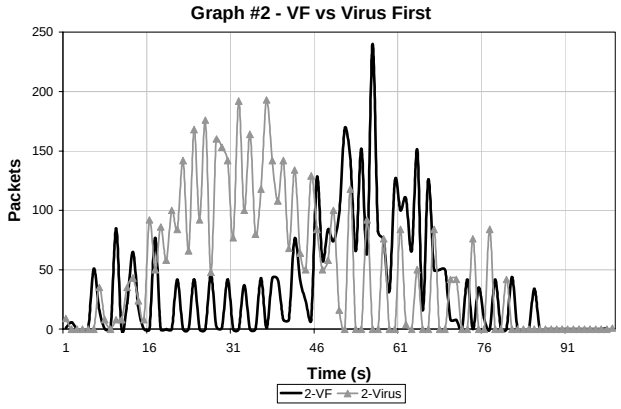


Figure 3: Graph 2 Network Usage

Figure 3 illustrates the results of running our experiments on Graph 2. It shows a significantly greater infection time for the virus-first condition with a drastic fall off at the end. Contrasting this is the much slimmer virus-fighter-first spike. The spike is more pronounced with this graph due to the higher connectivity of the layout. Increased connections did not greatly help or hurt the viruses' ability to propagate but did in fact greatly aide in the virus fighter's ability to spread quickly to all infected

nodes and then recede to a nominal level. The greater number of connections allows the virus fighter to spread faster when in cloning mode as well as helped to promote their recession after the virus threat had been neutralized. Graph 2 showed higher virus fighter creations under both conditions as compared to Graph 1 where connections were limited.

Graph 3

Graph 3 is an evolution of our graph in the second trial. Here we increased the graph's complexity and shifted to an almost fully connected graph. A flaw in the simulating environment was the reason for the graph not being fully connected. The increase in connectivity decreased the amount of time it took the viruses to propagate the entire graph as compared to many of the others trials. Paralleling this aspect was still greater efficiency on the part of the virus fighters. As complexity rose in each of the first three graphs, the virus fighters enjoyed continued gains in efficiency and lowered amounts of infection time under both starting conditions.

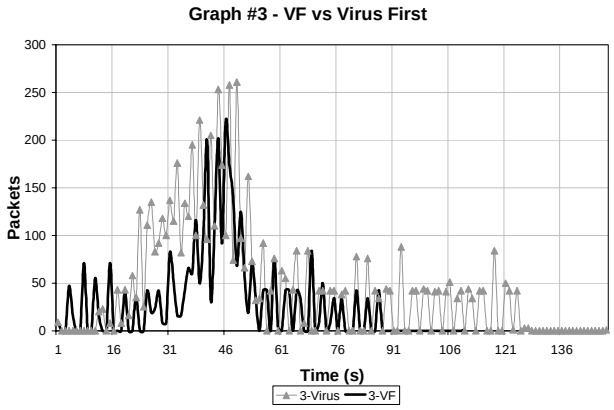


Figure 4: Graph 3 Network Usage

Both starting conditions finalized with a single virus fighter traversing the now cleaned network and enjoyed almost identical normal network load. While infection times were not so different between the starting conditions of this graph, the number of packets at their peek was noticeably greater for the virus-first condition. This was caused by the interconnectivity of the graph design. The virus fighters continually were aware of the entire graph and when viruses were introduced they were able to quickly clone to most if not all other nodes in a single iteration as opposed to many in the case of graph 1. This means a sharper spike at first but a fast decline as machines are cleaned and virus fighters find each other and die off

Graph 4

Trial 4 used Graph 4, seen in Figure 1, to test another interesting network arrangement. All the nodes in Graph 4 are only connected to the one center node. This effectively creates a bottle neck situation in which all viruses and virus fighters must at some point go through the center node if they are to detect the other nodes in the graph.

When the virus fighter is operating in search mode it will avoid this problem because it will only have to visit the center node once to learn about all the other nodes. Whenever it moves after that it will move directly to one of the fringe nodes without moving through the center node. However, when the virus fighter detects a virus and clones there are two cases. If the virus fighter is located at any of the fringe nodes it will immediately clone to the center node. If this node is infected then the virus fighter will clone to all of the fringe nodes. The second case is if the virus fighter is located at the center node when it first detects a virus. In this case it will clone to all of the fringe nodes immediately. The effect of this can be seen in figure five which displays the virus-fighter-first and virus-first network usage graph.

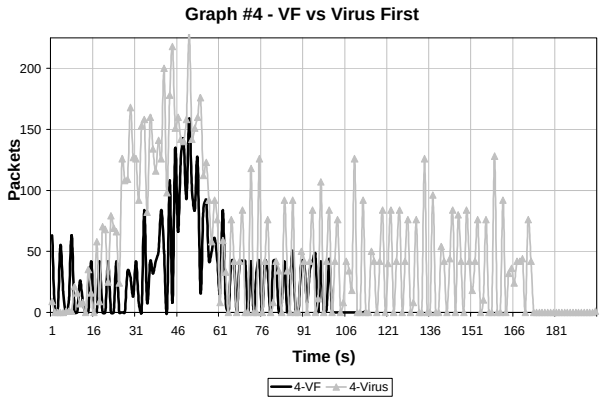


Figure 4: Graph 4 Network Usage

Comparing this graph to the other network usage graphs shows that the graph for this trial has a large difference. Because of how quickly the virus fighter's clone to all the nodes in this graph the peak drops back to normal network usage far faster than in any of the other trials. The drop is not immediate because it takes a little while for the many virus fighter clones to meet each other and die off to a low level. Interestingly, in this graph there were two and three Virus fighters left after the trial for virus-fighter-first and virus-first respectively. This is due to the nature of the movement algorithm of the virus fighters. Because they do not have to stop at the center node to get to their destination fringe nodes they did not meet each other die off. The low number of virus fighters created in both these trials, seen in table 2, is also due to how quickly they spread across the whole network.

Graph 5

Trial 5 used Graph 5 seen in Figure 1, to test a larger network of ten nodes instead of the five that were used in all other trials. As has been previously described, five nodes were used because there were five computers available in our test bed network. To simulate ten computers two Tahiti aglet servers were run on each computer in our network. To attempt to keep the network stats comparable with the previous trials, the graph was designed to make certain that no two Tahiti servers on the same computer had an edge to each other. This means that when a virus or virus fighter moves it has to move between computers, it can never move within a computer.

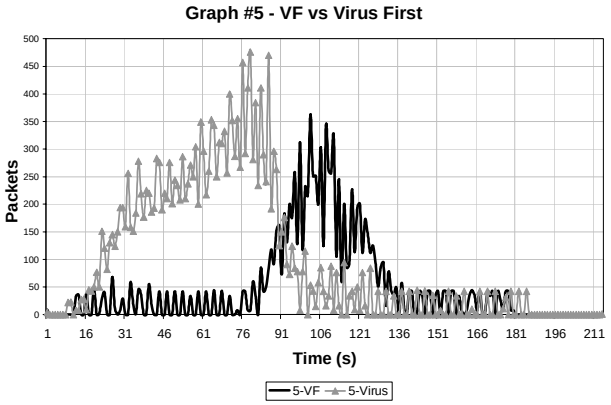


Figure 5: Graph 5 Network Usage

The results from this trial backed up all the results found with the smaller graphs. The most impressive data found was that the number of virus fighters created was eighteen and seventeen for the virus-fighter-first and virus-first trials. These numbers are not significantly different from the number of creations found in the other trials. In fact the numbers are nearly the same as those from Trial 3. This indicates that even though the number of nodes was doubled the number of virus fighters which spread through the network removing viruses was not terribly increased. However, our data did show that it took considerably longer for the virus fighters to eliminate the viruses, as seen in figure 6 which shows the network usage graph for both the virus-fighter-first and virus-first trials.

Conclusion

The data presented here makes a strong case for the effectiveness of a mobile autonomous agent to fight viruses. It has been made clear that the algorithm used by the virus fighters is far more efficient than the sample virus algorithm. The virus fighter's default mode of moving through the entire network uses little network bandwidth and the virus fighter's method of cloning when it detects viruses proved effective as it did not create a myriad of virus fighters. The virus fighter's footprint on the network is so small; it is capable of sneaking in behind the viruses and cleaning them even though the viruses may be using the majority of the network bandwidth available.

Also, comparing the virus-fighter-first and virus-first trials demonstrates that having the virus fighter constantly moving through the network is a more effective way of fighting off a virus infection than introducing the virus fighter after the infection has taken control. In the trials in which the virus was started first the network usage graphs peaked at a higher number of packets and remained at a high packet value for a longer duration than in the trials in which the virus fighter was started first. Clearly this is due to the fact that having the virus fighter constantly moving through the network causes the virus fighter to detect the viruses sooner and therefore clean them faster. Due to the fact that it detects the viruses faster the viruses are not capable of consuming much of the network usage. This can clearly be seen by the fact that in all the network usage graphs the peak of the virus-fighter-first graph is lower, or roughly the same, as the peak of the virus-first graph. This means that the most effective way to fight viruses with a mobile autonomous agent is to have that agent constantly moving through the network, because the virus fighter uses a negligible amount of network bandwidth in its searching mode.

Future Work

This system has been designed to test whether an autonomous agent could be an effective tool to fight viruses. There are two issues which would have to be addressed in future work into the concept of a virus fighting mobile autonomous agent. Namely, it is not possible for a virus fighter to know how to detect and clean all possible viruses which may some day be written and introduced to the internet as has been assumed in this work. In real life, patches must be written as new viruses are found on the internet. Therefore, the ability to teach the virus fighters moving through the network about new viruses and how to patch them would have to be added to the system developed in this work. We would hypothesize that this would still be an effective way of fighting viruses because it would not be necessary for each individual computer user to run a scan after the scan and patch were released. Instead the scan and patch would only have to be released to the virus fighters who would immediately begin to do the rest of the work. However this hypothesis would have to be tested in future work before it could be validated.

The second issue involves the fact that a virus scan is in fact a large drain on the resources of a computer, so it can not be run at any time as was assumed in this work. One possible way to solve this problem might be to break the virus fighter into two parts. One part would be

like the agent developed in this work which would guarantee full coverage of any network. The other part would be an agent which is left behind by the traversal agent to perform a virus scan during the idle processing time of a computer. This way the virus scan would not directly affect the user's experience on the system.. If this agent found a virus, it would send traversal clones to all adjacent computers just as this work has done. We would hypothesize that this method would still prove effective in fighting viruses, but it would cause the process of removing all the viruses to take longer. Again, this would have to be tested in future work.

Acknowledgments

A big thank you goes to the Computer Science Department and the College of Arts and Science at the University of Dayton for resources and support of this work.

References

Lance J. Hoffman, editor. 1990. Rogue Programs: Viruses,Worms, and Trojan Horses. Van Nostrand Reinhold. pages143–157.

Stan Franklin and Art Graesser, 1996, Is it an Agent, or just a Program?: Taxonomy for Autonomous Agents, *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*, Springer-Verlag, page 5.

Mark Allen Weiss, 1999, Data Structures and Algorithm Analysis in C++ 2nd Edition, Addison-Wesley, page 335.

Danny B. Lange and Mitsuro Oshima, 1998, Programming and Deploying Java Mobile Agents with Aglets. Addison-Wesley.

David Kotz and Robert S. Gray, August 1999, Mobile Agents and the Future of the Internet. *ACM Operating Systems Review 33(3)*. pages 7-13.

Stuart Russell and Peter Norvig. 2003. Artificial Intelligence A Modern Approach.Pearson Education. Page 32.

David Kortenkamp, Reid Simmons, Tod Milam, and Joaquín L. Fernández. March 2004. A Suite of Tools for Debugging Distributed Autonomous Systems. *Formal Methods in System Design Volume 24 , Issue 2.* pages 157-188.

Aglets.org. March 2003. The Aglets Portal. December 2003. <http://aglets.sourceforge.net/>.

Experiments in Guided Class Rebalance based on Class Structure

Sofia Visa

ECECS Department
ML 0030
University of Cincinnati
Cincinnati, OH 45221, USA
svisa@ececs.uc.edu

Anca Ralescu

ECECS Department
ML 0030
University of Cincinnati
Cincinnati, OH 45221, USA
Anca.Ralescu@uc.edu

Abstract

Up-sampling and *down-sampling* are the two most used methods in balancing the data when dealing with two class imbalance problem. However, none of the existing approaches to class rebalance take into account class information (e.g. distribution, within and between class distances, imbalance factor).

Rebalance of training data is considered necessary, since it was observed that usual classifiers do not perform well on imbalanced data and since the small class is the class of interest.

This study presents initial results of up-sampling methods based on various approaches to aggregation of class information such as spread, imbalance factor and the distance between the classes. Artificially generated data are used for experiments. The performance of each up-sampling method is evaluated with respect to how well the resulting data set reflects the underlying original class distribution, in terms of mean, standard deviation and (empirical) distribution function.

Introduction

Recent interest in learning classifiers for imbalanced data sets has brought about various proposals for new classifiers (Visa & Ralescu 2003), or for adopting known ones after a rebalancing of the training data. Most research in rebalancing the training set investigates the relative advantage, if any, of down-sampling, or up-sampling techniques (Drummond & Holte 2003). A guided resampling study is presented in (Nickerson & Milios 2001): the sampling method takes into account within class imbalance, assuming that elements of the imbalanced (small, or positive) class, are grouped in subclusters. Up-sampling is guided towards enlarging the under-represented clusters. Since the up-sampling step assumes that the number of (sub)clusters is known in advance, unsupervised clustering methods (e.g. k-means algorithm or self-organizing maps) have to be used prior to up-sampling, in order to obtain such an information about the unbalanced classes.

In (Zhang & Mani 2003) five different under-sampling methods are presented, each based on a k-nearest neighbors

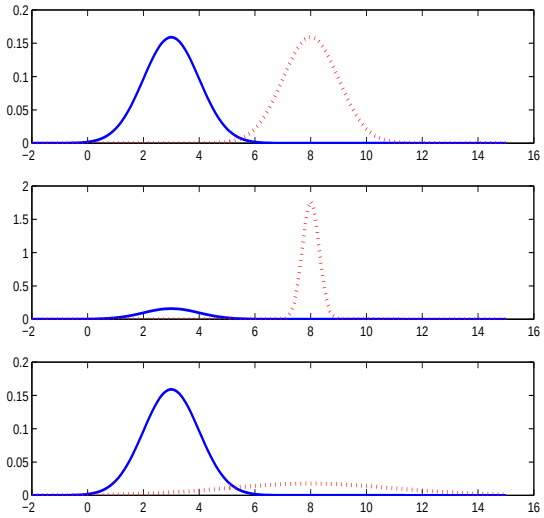


Figure 1: Mean 8 and standard deviations 1, 0.3 and 3 for the positive class.

technique. The experiments carried in (Maloof 2003) suggest that sampling, as well as adjusting the cost matrix, have the same effect as moving the decision threshold. An effective ratio of up/down sampling was recently investigated in (Estabrooks & Japkowicz 2004).

Evaluation of rebalancing is done in each study with respect to the performance of a selected classifier (e.g. ID3/C4.5, neural network) trained with the balanced training data. Therefore, in some sense such studies answer the question, "What is the best rebalancing approach to achieve best performance for the classifier C?".

Current study

The current study poses a different question. The starting point of the study is the remark that limited (poor) performance of a classifier for imbalanced data may due to the classifier itself and/or to the underlying data distribution. Rebalance is done by iterative up-sampling guided, at each

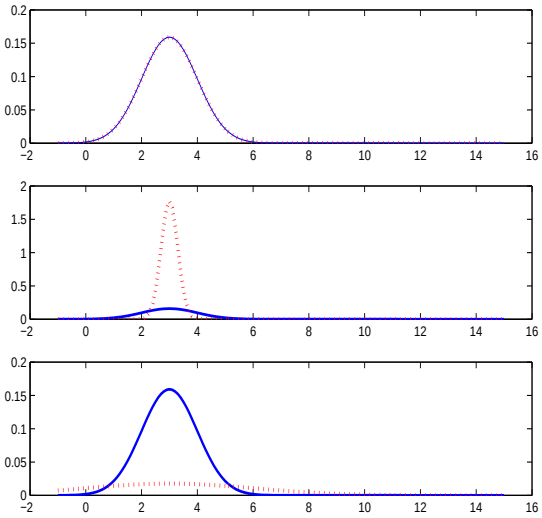


Figure 2: Mean 3 and standard deviations 1, 0.3 and 3 for the positive class.

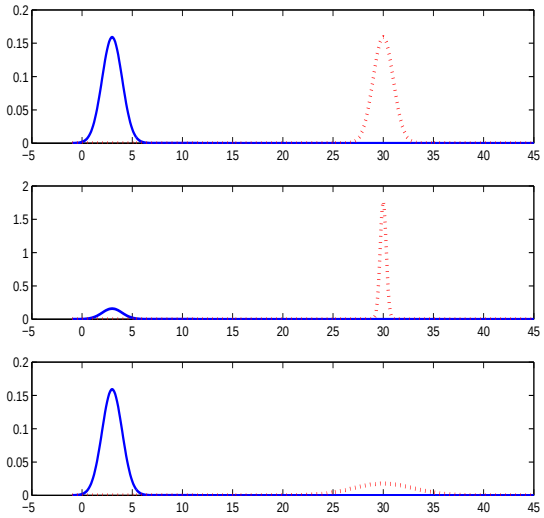


Figure 3: Mean 30 and standard deviations 1, 0.3 and 3 for the positive class.

iteration, by the characteristics (within class distance, between class distance, imbalance index. etc.) of the current data set.

Notation and Terminology. Throughout this study, C_+ and C_- denote the training data for the two classes of interest; n_+ and n_- denote the number of elements for C_+ and C_- respectively. It is assumed that to begin with $n_+ << n_-$, that is, that initially, the number of training points for C_+ is much smaller than those for C_-. For a given point in C_+ up-sampling generates a new data point, whose coordinates are determined by a parameter ϵ, d_+ and d_-, the average within class distance for the positive and negative class respectively, $d_{+,-}$, the minimum distance between C_+ and C_-, and the imbalance index, I. More precisely, the following relations hold:

$$n_+ = |C_+|;\ n_- = |C_-| \quad (1)$$

$$d_l = \frac{\sum_{x,x' \in C_l} dist(x,x')}{n_l}; l \in \{+,-\} \quad (2)$$

$$d_{+,-} = min\{d(x,y);\ x \in C_+; y \in C_-\} \quad (3)$$

$$I = \frac{n_+}{n_-} \quad (4)$$

$$MaxD_+(\mathbf{x}) = max_{\mathbf{x}' \in C_+} d(\mathbf{x},\mathbf{x}'),\ \mathbf{x} \in C_+ \quad (5)$$

where d denotes any distance function (here the Euclidean distance is used). Furthermore, μ_+, μ_- denote the means, and σ_+, σ_- the standard deviations of the (Gaussian) distributions underlying the positive and negative classes respectively.

The data set. The experiments carried out for the current study use an artificial data set. Training data points are generated for each class from the Gaussian distribution as follows:

- **The negative class:** Throughout the experiments, the data for the negative class is generated from Gaussian with mean $\mu_- = 3$ and standard deviation $\sigma_- = 1$;
- **The positive class:** The data for the positive class is generated from the Gaussian distribution with parameters $\mu_+ \in \{3, 8, 30\}$, and $\sigma_+ \in \{1, 0.3, 3\}$, for a total of nine different scenarios.

For each scenario, 160 two-dimensional training points are generated for each of the two classes using a Gaussian distribution along both (x and y) coordinates. Figures 1, 2 and 3 show the distribution underlying each class for each scenario. The selection of various means and standard deviations for C_+ is intended to cover the cases when C_- and C_+ are (a) overlapping (same mean same standard deviation), (b) different but *not very far and not very close* ($\mu_+ = 8$), and (c) very far away ($\mu_+ = 30$), for different cases of data spread around the respective means.

Data rebalancing

The rebalancing algorithm takes as input n_- for C_-, and a very small number n_+^0 for C_+. In the experiments discussed here, n_+^0 data points are selected as *seeds* for the up-sampling algorithm from n_+ $(= n_-)$ data for C_+.

For a generic data point $\mathbf{x} \in C_+$ a new value is generated as:

$$\mathbf{x} \leftarrow \mathbf{x} + \mathbf{A} \quad (6)$$

where $\mathbf{A}$ is generated randomly according to the uniform in an interval $[-\epsilon, \epsilon]$. The parameter ϵ is defined so as to take into account current characteristics of C_+ and relation between C_- and C_+. The idea is to allow these characteristics to control the sampling of a new point. Three definitions are considered for the parameter ϵ:

The sampling parameter and the rebalancing algorithm

The first definition for the sampling parameter is given in (7).

$$\epsilon_{index} = \frac{d_+}{d_-} \cdot \frac{1}{1+d_{+-}} + I \cdot d_{+-} \cdot \frac{d_+ \wedge d_-}{d_+ \vee d_-} \qquad (7)$$

where $\wedge$ and $\vee$ denote minimum and maximum respectively.

It can be seen from (7) that ϵ varies with I, d_+, d_-, and $d_{+,-}$. For example, to begin with, when I is smallest, $\epsilon_{index} \simeq \frac{d_+}{d_-} \cdot \frac{1}{1+d_{+-}}$, that is, it depends indirectly on the minimum distance between classes; as I increases (to its maximum value of $\simeq 1$), this dependence shifts. For the experiments carried out in this study, with the initial values $n_+^0 = 5$ and $n_- = 160$, the imbalance index takes, successively, the values $0.0313, 0.0625, 0.125, 0.25$ and 0.5 before becoming 1 in the last step of updating. On the other hand, as $d_{+,-}$ increases, the contribution of the second term towards the final value of ϵ_{index} increases in two ways, first indirectly, by decreasing the magnitude of the first term, and secondly directly, by increasing the magnitude of the second term. In the opposite situation, when the classes are very close (or totally overlapping, see figure 2), the second term of the equation has a lower weight in computing the index and the behavior of the index is given mainly by the first term.

The detailed investigation of the contribution/necessity of each term of the equation (7) is undergoing. Still some observations can be made as follows: the imbalance index acts like a regularization term to avoid initial big jumps (from the current seed) in sampling another value for the imbalanced class, (avoiding to generate a *sparse* positive class).

The second proposal for the sampling parameter is given by equation (8):

$$\epsilon_{ave} = I \cdot d_+ \qquad (8)$$

With either of these definitions, the value of the sampling parameter ϵ is updated after each sampling step as Algorithm 1 shows.

Algorithm 1 *While I != 1 do*

1. *Compute ϵ_{index} (or ϵ_{ave}) according to the equation (7) (or (8)).*
2. **Up-sampling:** *For each seed, $\mathbf{x} \in C_+$, compute*

$$\mathbf{x} \leftarrow \mathbf{x} + \mathbf{A}$$

where $\mathbf{A} \in [-\epsilon, \epsilon]$ are generated randomly using a uniform distribution.

The final proposal for the sampling parameter is similar to the ϵ_{ave} with d_+ replaced by $MaxD_+(\mathbf{x})$, for each $\mathbf{x} \in C_+$ as shown in equation (9).

$$\epsilon_{Max(\mathbf{x})} = I \cdot \frac{MaxD_+(\mathbf{x})}{2} \qquad (9)$$

	$\sigma = 1$		$\sigma = 0.3$		$\sigma = 3$	
	μ	σ	μ	σ	μ	σ
ϵ_{index}	0.034	**0.06**	**0.006**	0.152	0.185	**0.399**
ϵ_{Max}	0.037	0.12	0.012	**0.021**	0.190	0.454
ϵ_{ave}	**0.027**	0.15	0.013	0.033	**0.182**	0.517

Table 1: Results for $\mu_+ = 8$: averaged (over 100 runs) distances between the mean (standard deviation) of obtained classes and the mean (standard deviation) of original data.

	$\sigma = 1$		$\sigma = 0.3$		$\sigma = 3$	
	μ	σ	μ	σ	μ	σ
ϵ_{index}	0.095	0.374	0.021	0.061	0.260	0.483
ϵ_{ave}	**0.070**	**0.064**	**0.018**	**0.034**	**0.225**	**0.374**
ϵ_{ave}	0.070	0.089	0.019	0.043	0.230	0.425

Table 2: Results for $\mu_+ = 3$: averaged (over 100 runs) distances between the mean and standard deviation of obtained classes and the mean and standard deviation of original data.

In this case, the rebalancing algorithm is slightly different: as it can be seen from equation (9) the value of the sampling parameter depends now of the current seed. This is reflected in the rebalancing algorithm, Algorithm 2, which is Algorithm 1 with and intermediate step to compute $\epsilon_{Max(\mathbf{x})}$ for each seed $\mathbf{x}$.

Algorithm 2 *While I != 1 do*

For each seed $\mathbf{x} \in C_+$enumerate,

1. *Compute $MaxD_+(\mathbf{x})$ according to equation (5).*
2. **Compute sampling parameter** *$\epsilon_{Max\mathbf{x}}$ according to the equation (9).*
3. **Up-sampling:** *Generate one instance for the positive class,*

$$\mathbf{x} \leftarrow \mathbf{x} + \mathbf{A}$$

where $\mathbf{A}$ generated randomly according to the uniform distribution in the $[-\epsilon, \epsilon]$.

Results

The approach described above is applied to a two dimensional data set, with $n_- = 160$, $n_+^0 = 5$ the latter generated randomly from 160 data points for the positive class. All results are averaged over 100 runs: for the same positive class, at each of the 100 runs, five seeds are selected randomly and

	$\sigma = 1$		$\sigma = 0.3$		$\sigma = 3$	
	μ	σ	μ	σ	μ	σ
ϵ_{index}	0.110	3.346	0.036	0.152	5.187	**1.103**
ϵ_{ave}	0.102	**0.066**	0.008	**0.025**	**0.104**	0.277
ϵ_{ave}	**0.100**	0.108	**0.007**	0.031	0.113	0.327

Table 3: Results for $\mu_+ = 30$: averaged (over 100 runs) distances between the mean (standard deviation) of obtained classes and the mean (standard deviation) of original data.

		$\sigma = 1$	$\sigma = 0.3$	$\sigma = 3$
$\mu = 8$	$x < 7$	ϵ_{Max}	ϵ_{ave}	ϵ_{Max}
	$x < 8$	$\epsilon_{ave}(\epsilon_{Max})$	ϵ_{index}	ϵ_{Max}
	$x < 9$	ϵ_{Max}	$\epsilon_{ave}(\epsilon_{Max})$	$\epsilon_{ave}(\epsilon_{Max})$
$\mu = 3$	$x < 2$	ϵ_{Max}	$\epsilon_{ave}(\epsilon_{Max})$	ϵ_{Max}
	$x < 3$	ϵ_{Max}	ϵ_{Max}	ϵ_{Max}
	$x < 4$	$\epsilon_{ave}(\epsilon_{Max})$	$\epsilon_{ave}(\epsilon_{Max})$	$\epsilon_{ave}(\epsilon_{Max})$
$\mu = 30$	$x < 28$	$\epsilon_{ave}(\epsilon_{Max})$	$\epsilon_{Max}(\epsilon_{ave})$	ϵ_{ave}
	$x < 30$	ϵ_{index}	ϵ_{index}	ϵ_{ave}
	$x < 32$	ϵ_{ave}	$\epsilon_{Max}(\epsilon_{ave})$	$\epsilon_{Max}(\epsilon_{ave})$

Table 4: Results for comparing the probability functions: for each of the nine scenarios, three probabilities functions are computed. The index which gives the closest probability function to the original sets probability function are shown. The entries in parenthesis are the second runner up (its value differs from the winner, only by 0.001).

the classes are rebalanced according with the algorithms 1 and 2 using the three sampling parameters described previously. Figures 4-12 illustrate the results obtained for the positive class after rebalancing. There is one figure for each of the nine scenarios investigated in this study and in each figure the negative data are plotted as circles (o) and the positive data as pluses ($+$).

In each figure, the plot in the upper left corner shows the obtained rebalanced class using ϵ_{index}, the plot in the upper right corner illustrates the rebalanced class using ϵ_{Max}, the plot in lower left corner shows the rebalance results using ϵ_{ave}. The lower right corner shows the original data. The results are evaluated by (1) comparing the distribution parameters, (2) comparing the empirical distribution functions for the original positive class and the data generated by up-sampling, and finally (3) by comparing the original data for the positive class C_+ and the generated data set C'_+ as sets.

Comparing means and standard deviations

- The means and standard deviations are computed for the newly obtained positive classes and averaged over 100 runs. The Euclidean distances between these values and the real mean and standard deviations are listed in the Tables 1, 2 and 3. The closest distances to the original positive class are highlighted. As it can be observed,
  1. For the case when classes are very close (overlapped), the best performance is given by the ϵ_{Max}, see table 2.
  2. Overall, ϵ_{Max} has the best behavior: four closest means and six closest standard deviations over all nine possible scenarios.
  3. ϵ_{ave} has the worst behavior in respect with standard deviation but preserves well the mean of the positive class. In conclusion, ϵ_{ave} expands the class without capturing much the standard deviation but preserves the expected value (preserves a certain symmetry in up-sampling).
  4. ϵ_{index} approximates well the original standard deviation for reasonable close classes when positive mean is eight) and bad for high distance between classes (when positive mean is 30). It does not preserve well the mean.
- ϵ_{index} approximates the best the mean and the standard deviation of the underlying distribution when the evaluation of closeness is as follows: for each of the 100 runs the mean and standard deviation are computed for each of the three instances of the positive classes (obtained using the three sampling parameters). Next, the Euclidean distance to the actual mean and standard deviation are computed and the closest one is the winner of the current run. In this case *the amount* by which the winner is closest to the actual values is lost for the next run (it is not accumulated over the runs as in the previous paragraph). The results are averaged over 100 runs. In this setup, ϵ_{index} preserves the mean the best (four out of the nine scenarios) and approximates the standard deviation the best (six out of the nine scenarios).

  The two apparently contradictory results can be explained as follows: when ϵ_{index} gives the best approximation of the mean, it does so by a narrower margin than when it fails to give the best approximation.

Comparing the probability distribution functions

The second way of evaluating the up-sampling algorithms is to compare the distribution underlying the original data set C_+ and the empirical distribution of the data set C'_+ generated by up-sampling. Note that in the experiments presented here, the theoretical distribution of the original C_+ is known. However, for comparison purpose, the empirical rather than the theoretical distribution function is used for C_+ as well. The *empirical distribution* $F(x)$ corresponding to a collection of observed values, is given by equation (10).

$$F(x) = \frac{\#\ of\ data\ < x}{total\ \#\ of\ data} \tag{10}$$

For illustration purposes, for each of the nine scenarios, three values for the distribution functions are calculated (for the x values shown in the second column of the table 4). The table 4 shows the winner index in comparing $F(x)$, the probability distributions of the original data, with $F'(x)$ obtained by resampling. Only the results with respect to the x coordinate (obtained by projection of the data set on the $x-$axis are shown, the results along the y coordinate are of the same quality).

Discussion of results

- **Standard deviation discussion:** For larger standard deviation ($\sigma = 1$ and $\sigma = 30$) ϵ_{Max} performs the best. Also ϵ_{index} gives good results for these cases but does not approximate well the original distribution for large standard deviation (for $\sigma = 30$ it expands the positive class too much). For small standard deviation, both ϵ_{Max} and $\epsilon_{average}$ give good approximation of the original class.
- **Mean class discussion:** For mean $\mu_+ = 3$ (total overlap of C_+ and C_-), and for $\mu = 8$, the best positive class is achieved by using ϵ_{Max}; for $\mu = 30$ (C_+ and C_- are far apart) the best performance is given by $\epsilon_{average}$ for almost any standard deviation.

Comparing the distance between the original and up-sampled class

The final criterion for evaluating the up-sampling algorithm is taking the minimum of the (geometric) distance (here in the two-dimensional plane) between C_+ and $C'_{+,\epsilon}$, calculated for each sampling parameter ϵ as shown in equation (11).

$$d_\epsilon(C_+, C'_+) = \sum_{\mathbf{x}\in C_+, \mathbf{x}'\in C'_{+,\epsilon}} d(\mathbf{x}, \mathbf{x}') \qquad (11)$$

It is verified that at least, for the artificial data set generated for these experiments, taken across all nine scenarios, ϵ_{ave} is the best, that is, it satisfies equation (12):

$$d_{\epsilon_{ave}}(C_+, C'_+) = min_{\epsilon\in\{\epsilon_{index}, \epsilon_{ave}, \epsilon_{Max}\}} d_\epsilon(C_+, C'_+) \qquad (12)$$

That is, this case, by up-sampling with the parameter ϵ_{ave} the resulting (up-sampled) positive class is closest to the original class in all nine scenarios. This result is not surprising since the equation of ϵ_{ave} involves an average distance in the positive class, hence at each stage of up-sampling the new points are closer to the seeds (distance between classes does not affect this index).

Conclusions

This study proposes an alternative criterion of the up-sampling/down-sampling operations for rebalancing training data for a two class classification problem. Rather than to evaluate the up-sampling with respect to the performance of a particular classifier (or family of classifiers) it evaluates it with respect to the characteristics underlying the classification problem (imbalance factor, distance within and between classes, etc.). Three options for selecting the sampling parameter for up-sampling are proposed and results for experiments for nine different scenarios of distribution data for each of these parameters are obtained. The results are promising in several ways: each of the proposed sampling parameters seems to perform quite well, often the difference in the ranking of their performance is due to differences in the third decimal of the criterion used for comparison; the performance is consistent across three criteria of comparison; the comparison criteria are particular instances of more general statistical problems (e.g. the first criterion corresponds to estimation of parameters of a known distribution, the second, to the approximation of the - unknown - underlying distribution).

Although it appears that ϵ_{Max} and ϵ_{ave} perform better, this seems to be due mainly to the fact that they measure only how well the rebalanced positive class captures the underlying distribution of the original class. But in an imbalanced classification problem involving more classes, the rebalance process has to be guided by the whole set-up of the data: the distance between the classes must be considered and properties of the big class(es) must be used as well. For such problems, ϵ_{index} appears to be more comprehensive and hence more promising.

Finally, the artificial data set provides the important information on the actual positive class which used in evaluating the performance of the up-sampling algorithms (although, it is important to note, not in the algorithms themselves). For real data sets, where the original positive class is not known, and for which to begin with there are very few instances, the algorithms must be adapted so as to include a stopping condition (in conjunction with that checking the imbalance index I) which tracks the change in the parameter that are estimated. In a way, the extension to such cases is straightforward but an analysis of the convergence of such algorithms is also be needed in addition to extensive experimental results.

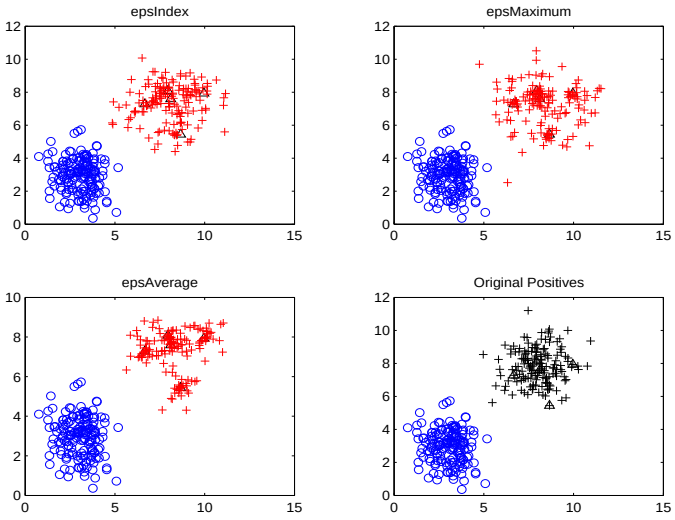


Figure 4: Rebalance of the positive class for mean 8 and standard deviation 1.

Acknowledgments

Sofia Visa's work for this study was partially supported by a Graduate Fellowship from Ohio Board of Regents. Anca Ralescu's work for this study was partially supported by the Grant N000140310706 from the Department of the Navy.

References

Drummond, C., and Holte, C. 2003. C4.5, class imbalance, and cost sensitivity: Why undersampling beats oversampling. Proc. of the ICML-2003 Workshop: Learning with Imbalanced Data Sets II,1-8.

Estabrooks, A.; Jo, T., and Japkowicz, N. 2004. A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence* 20(1):18–36.

Maloof, M. 2003. Learning when data sets are imbalanced and when costs are unequal and unknown. Proc. of the ICML-2003 Workshop:Learning with Imbalanced Data Sets II, 73-80.

Nickerson, A.S.; Japkowicz, N., and Milios, E. 2001. Using unsupervised learning to guide resampling in imbalanced data sets. 261–265.

Visa, S., and Ralescu, A. 2003. Learning imbalanced and overlapping classes using fuzzy sets. Proc. of the ICML-2003 Workshop: Learning with Imbalanced Data Sets II, 97-104.

Zhang, J., and Mani, I. 2003. knn approach to unbalanced data distributions: A case study involving information extraction. Proc. of the ICML-2003 Workshop: Learning with Imbalanced Data Sets II, 42-48.

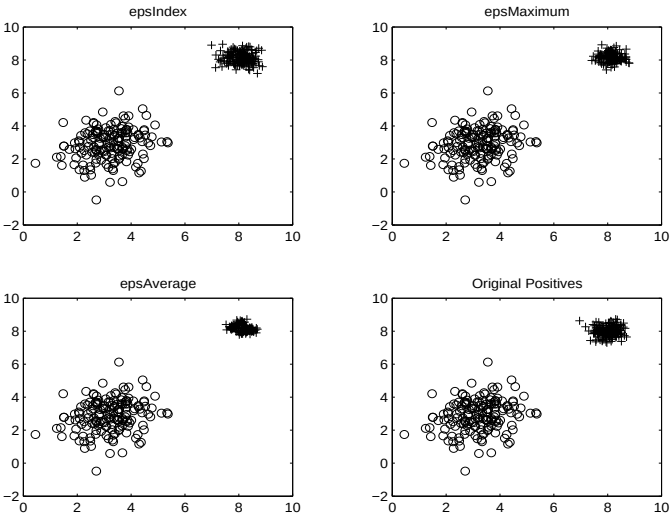


Figure 5: Rebalance of the positive class for mean 8 and standard deviation 0.3.

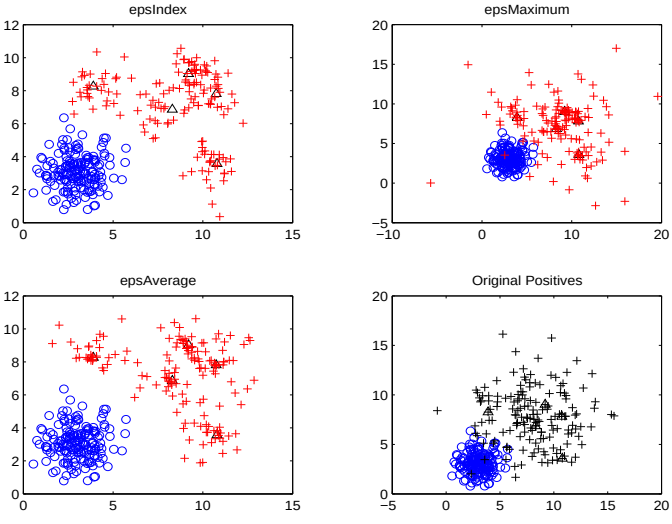


Figure 6: Rebalance of the positive class for mean 8 and standard deviation 3.

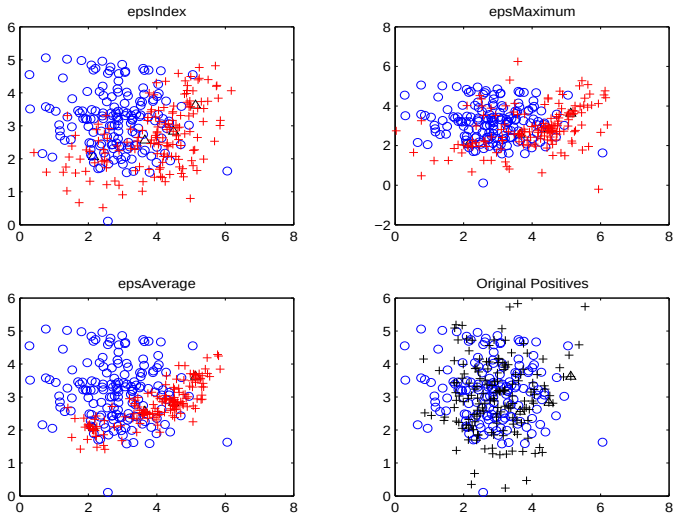


Figure 7: Rebalance of the positive class for mean 3 and standard deviation 1.

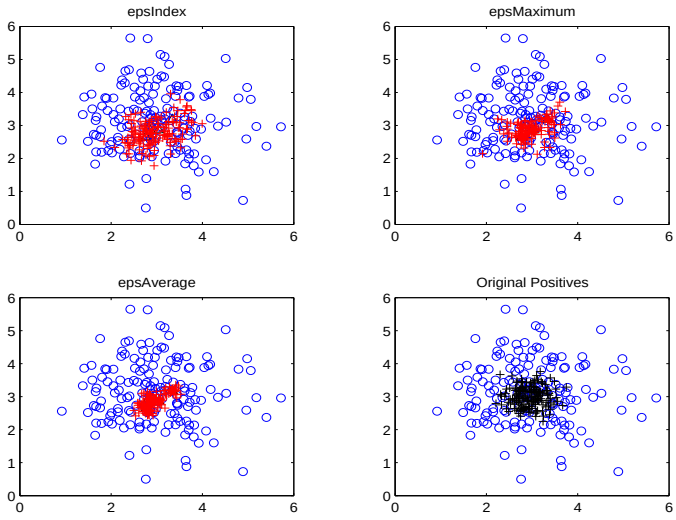


Figure 8: Rebalance of the positive class for mean 3 and standard deviation 0.3.

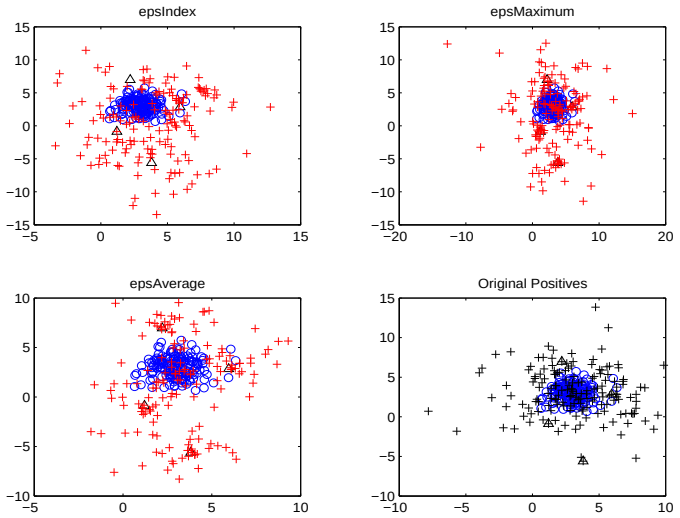


Figure 9: Rebalance of the positive class for mean 3 and standard deviation 3.

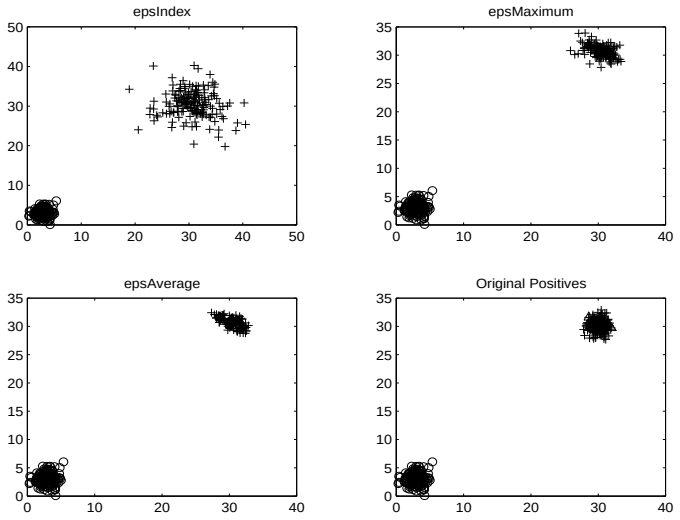


Figure 10: Rebalance of the positive class for mean 30 and standard deviation 1.

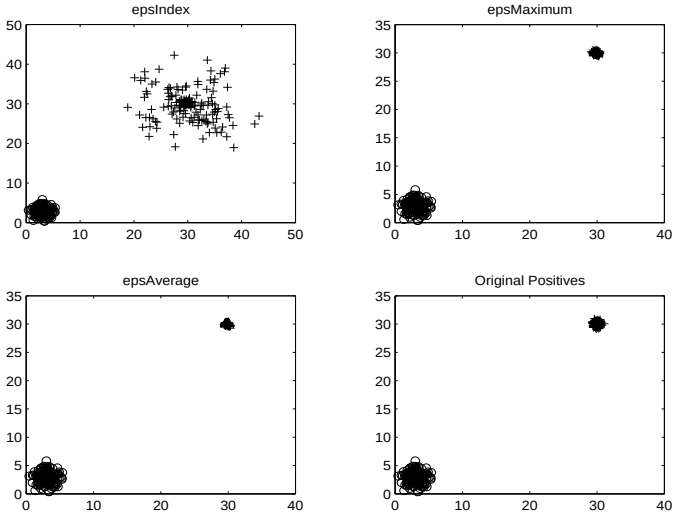


Figure 11: Rebalance of the positive class for mean 30 and standard deviation 0.3.

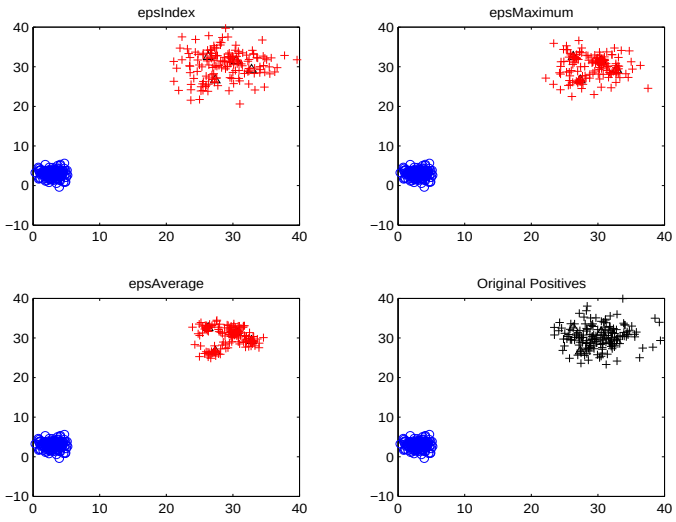


Figure 12: Rebalance of the positive class for mean 30 and standard deviation 3.

Adding NL Technology to an Online Language Instructional Tool

Paul Buchheit
Harold Washington College
pbuchheit@ccc.edu

Abstract

The Online Language Assistant is being developed to allow students of different languages and cultures to conduct online conversations with translation help from the computer. In addition to person-to-person communication, a person-to-computer option is included. The propositional approach employed by the Infant natural language processing system drives this part of the language assistant. This paper will provide an overview of the OLA and relevant parts of Infant, and then describe the integration of the two methodologies into a human/computer communication tool.

Introduction

Work on the Infant System has focused on the understanding of sentence input in conversations with a small child [1]. It incorporates many of the principles of the Image-Propositional-Syntactic (IPS) hypothesis [2,3], which suggests that language is derived from the propositions formed by the images processed by the brain.

In 2002 the One Globe Language Group, a consortium of researchers from Harold Washington College, the Chicago Public Schools, and private businesses, was formed to develop an improved approach to language learning using the principles of natural language research, the interactivity of the Internet, and multimedia technology. The result was the Online Language Assistant (OLA), which provides synchronous two-way interaction between students of different languages and cultures. A slide show of the methodology can be viewed at http://www.oneglobe.com/demo .

A major goal of the OLA effort was to combine the natural language capabilities of the Infant System with the instructional interface of the language learning system. Thus, students would not only be able to communicate with each other, but in the absence of a partner they could speak directly to the computer for the purpose of working with a second language. This paper will describe the means by which this integration was accomplished.

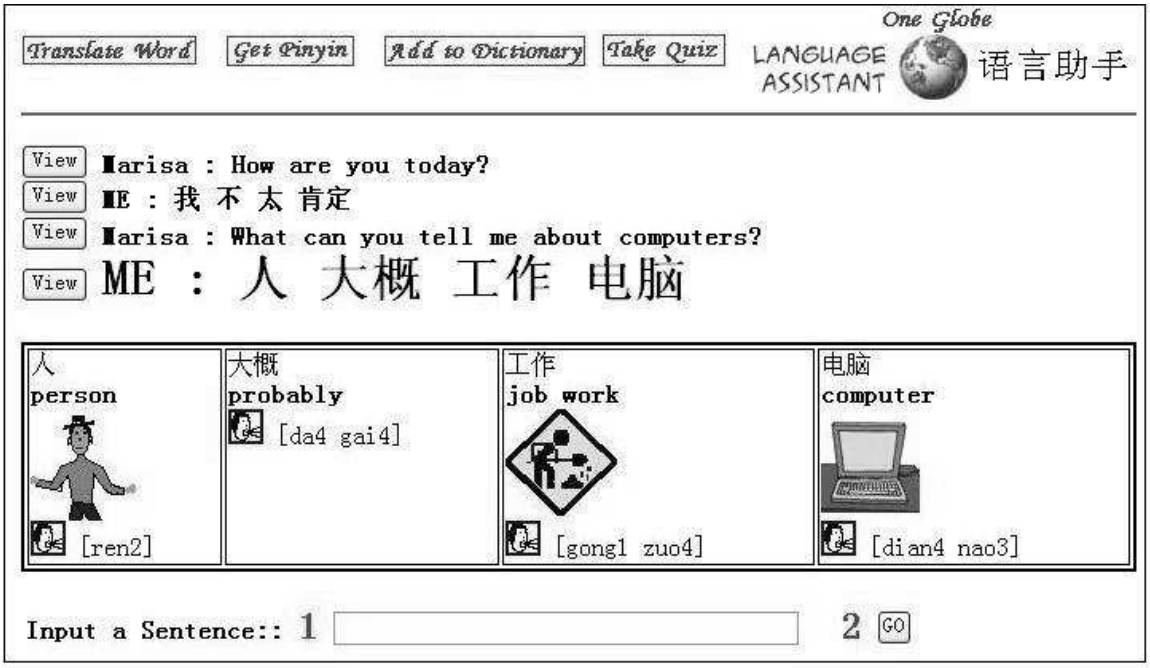


Figure 1 - The OLA Interface

The Online Language Assistant (OLA)

Figure 1 shows the basic OLA interface. A student by the name of Marisa is communicating with a partner - in this case the computer - for the purpose of improving her Chinese. OLA has been developed primarily as a classroom interface for students who wish to communicate directly with a foreign language student. The partners may communicate free-style, or they may take advantage of thematic lessons that were designed for OLA based on the principles of the National Standards for Foreign Language Learning [4]. In addition, there is an option for group communication.

There are a number of help features available to OLA users. Students may obtain alternative translations from selected Internet sites, they may update the online dictionary, and they may evaluate their progress by participating in an online quiz.

Translation between languages is currently word-to-word, although continued development of the system will employ the syntactic and semantic parsing modules of the Infant System

for the purpose of identifying parts of speech and sentence structures. Since all OLA conversations are recorded, it will also be possible to use probabilistic methods to assist with language-to-language translation. It is interesting to note, however, that students may benefit from the presentation of second language input in its original form, without structural translation. This allows them to learn the syntax of the foreign language by repeated observation of word and phrase order. Our future evaluations will address this issue.

Three pilot tests have been run with OLA, most recently a colloquium at a Chicago high school with 8 participants, 4 learning English and 4 learning Chinese. Although evaluation was not performed, informal feedback indicated that the OLA sessions were instructional and enjoyable.

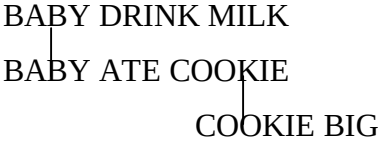
OLA has been developed as an open-source software tool on the Internet. Therefore it is easily accessible to users, although for security purposes access is carefully managed. It is written in PHP, with a MySQL back end and Java applets for audio and video features.

The Infant System – Hierarchical Logical Form

Previous work by Buchheit [1,5] has argued for a logical, semantics-based representation of sentences called Hierarchical Logical Form (HLF). For example, the sentence

The baby who ate the big cookie is drinking the milk

is translated into HLF as:



The top-level proposition in each sentence represents the main propositional thought, and subordinate clauses, while dependent on higher-level clauses, still form complete thoughts based on their Agent/Action/Object case assignments. In the HLF above,

code	phr	fuz	gen		
1	adult help child	8		Edit	Delete
2	baby crawl _	7		Edit	Delete
3	baby cry _	7		Edit	Delete
4	baby do school	2		Edit	Delete
5	baby do work	1		Edit	Delete
6	baby drink milk	7	food	Edit	Delete
7	baby eat cookie	7	food	Edit	Delete

code	eng	chi		
10	i do not know	我不知道	Edit	Delete
11	I do not know what you are talking about	我不知道你在讲什么	Edit	Delete
12	i am not sure	我不太肯定	Edit	Delete
13	what do you think about that?	你对此有什么想法?	Edit	Delete
14	what do you mean?	你是什么意思?	Edit	Delete
15	i do not think so	我不这样认为	Edit	Delete
16	that may be wrong	那可能是错误的	Edit	Delete
17	i understand	我明白	Edit	Delete

Figure 2 - Propositions and Common Responses

(BABY DRINK MILK) is the primary thought in the sentence. (BABY ATE COOKIE) is a supplementary thought that helps describe the baby, and (COOKIE BIG) in turn describes the cookie. The IPS hypothesis suggests that each of the three propositions is represented by a pattern of neural activity, and that the intensity of each activation frequency corresponds to its position in the HLF.

It was recognized during the development of OLA that since HLF is a deeper semantic representation of a sentence, it could serve as a global representation of meaning across languages. Thus simple propositions that express universal concepts could be stored in a database and translated into an appropriate language-specific syntactic form for an OLA participant.

Figure 2 shows some of the propositions stored in the database. For example, *baby eat cookie* has a high fuzzy value while *baby do work* has a low fuzzy value. References to

1. Search proposition database for phrases which contain a word or words that is/are subsumed by a more general target word or words in the current topic of conversation.

2. Assign relevance values to chosen phrases based on relationship to target word(s), degree of specificity, frequency of usage, and recency of usage.

3. Choose randomly from candidate phrases.

4. In the absence of candidate phrases or as a result of random choice, select a common (innocuous) response rather than a specific response.

5. Convert to required language.

Figure 3 - Specification Algorithm

milk and *cookie* denote general forms of *food*, and are therefore tagged as thematic elements for use in conversation. The figure also shows some of the common phrases that are used in casual conversation.

Integration

While OLA is normally employed as a one-to-one or many-to-many conversational interface among human participants, it can be used as a person-to-computer instructional tool by activating the natural language component. Some of the Infant understanding modules have been adapted to OLA for this purpose. Most significantly, a *specification* algorithm (see Figure 3) is used to focus a conversation on a specific topic. For example, a user's reference to 'school' will cause Infant OLA to seek recent mention of school-related sentence input. Lacking this, it will either seek information from the database or respond innocuously.

The Infant OLA module deals largely with simple Subject-Verb-Object sentences, and therefore is able to present the results in other languages in a syntactically correct format. Figure 4 demonstrates this capability in Chinese and English. Although the computer's responses are occasionally lacking with respect to the target language's common usage, they serve to provide the user experience with vocabulary, writing, and phonetics.

Figure 4 - A conversation with the computer

Future Work

The specification algorithm is one of 13 response forms adapted from work in speech act theory [6,7,8] and defined in [9] for the Infant System. The response forms include *Change Subject, Paraphrase, Repair, Assess, Agree/Disagree, Specify, Generalize, Refer to Past, Imply about Future*. At present, the OLA system uses only Specify and the choice of an innocuous reply. As the project develops, the NL interface will be augmented with other responses previously used by Infant.

References

[1] Buchheit, Paul (1991). *INFANT: A Connectionist-Like Knowledge Base and Natural Language Processing System*. Ph.D. Thesis, University of Illinois at Chicago.

[2] Buchheit, Paul (1999). A Neuro-Propositional Model of Language Processing, *International Journal of Intelligent Systems*, Vol 14, No. 6, 6/99.

[3] Buchheit, Paul (2000). Images to Syntax: A Neuropropositional Model of Language *Cognitive Systems Research*, Vol 1, Issue 3, April 2000: http://www.elsevier.nl/locate/cogsys?menu=cont&label=table

[4] National Standards in Foreign Language Learning Project. (1999). *Standards for Foreign Language Learning in the 21st Century*. Lawrence, KS: Allen Press.

[5] Buchheit, Paul (1993). 'INFANT: A Modular Approach to Natural Language Processing.' *Proceedings of the 21st ACM Computer Science Conference*, Indiana Convention Center, Indianapolis, IN, February 16-18, 1993, pp. 410-417.

[6] Austin, J. L. (1962). *How To Do Things With Words*. New York: Oxford University
Press.

[7] Grice, H. P. (1975). Logic and conversation. In Cole & Morgan, Eds., *Syntax and Semantics* vol 3: Speech Acts, pp 113-127. New York: Academic Press.

[8] Sacks, H., Schegloff, E., and Jefferson, G. (1974). A simplest systematics for the organization of turn-taking in conversation. *Language* 50:696-735.

[9] Buchheit, Paul (2002). *"Classifying Responses in a Natural Language System."* (unpublished).

A Context-dependent Metric for Similarity -Application to Supervised Learning of Handwritten Characters-

Waibhav Tembe and Anca Ralescu
{wtembe,aralescu}@ececs.uc.edu
Machine Learning and Computational Intelligence Laboratory
Department of ECECS, University of Cincinnati, Cincinnati, OH 45221-0030, USA

Abstract

This paper proposes a new measure of similarity called context dependent measure of similarity and defines cluster structure and associated parameters based on this measure. These ideas are integrated into a supervised learning algorithm with different classification schemes and applied to the problem of hand-written character recognition. The algorithm performance is tested on a real data set through learning curve and extensive analysis is provided. Comparisons with a popular minimum distance classifier shows that the performance of the algorithm is very promising (prediction accuracy as high as 96%) even though sophisticated image processing and mathematical techniques are not employed.

1 Introduction

Research in character recognition continues to impact the day-to-day life through numerous automated systems incorporating hundreds of character recognition algorithms. Detailed survey papers such as[1] have summarized these algorithms systematically. Mathematical and machine learning techniques have been applied to handwritten character recognition systems for more than three decades[2]. Typically, characters are available as images and image processing techniques (e.g., automated feature extraction, segmentation[3, 4, 5]) can be directly applied to character recognition systems. Character recognition paradigms typically consist of three phases: **segmentation** (words are split into constituent symbols), **feature extraction** (processing of individual characters for prototype generation) and **classification**. Segmentation and feature extraction are not always independent and methods that do not follow all three steps have been developed as well[6].

The current study addresses the following feature extraction and classification problem: Given individual characters that have been segmented accurately, the task is to build prototypes for classifying unseen instances (test data). Character recognition is a very well-studied problem and the aim of the current study is not to review the voluminous literature on this problem. Considering the space constraints and scope of the current study, a few pointers demonstrating diversity in the literature have been mentioned next. Some interesting character recognition schemes are: Inflection points and bitmap based approach[7], algorithm based on the nature of deformation of characters due to the characters *around* and the writer's characteristics[8], the nearest neighbor based approaches[9, 10], probabilistic methods[11], a *distance transform* curve[12] to measure similarity between images, and asymmetric Mahalanobis metric based methods[13]. Advanced and complex models include combined classifiers[14, 15], Hidden Markov Models(HMM)[16, 17] in which various *states* correspond to characters that make up the words and Support Vector Machines(SVMs)[18, 19, 2].

2 Organization of the Paper

From this point on, this paper is organized as follows: Section 3 defines the symbols used in the current study. Section 4 describes the concept of context-dependent divergence and similarity followed by associated cluster structure in section 5. Section 6 describes the supervised learning approach to character recognition using different strategies including extensive discussion of results and comparisons. Finally, section 7 summarizes the findings of the study.

3 Terminology

In the current study, $X = X_{i,e}, i = 1, \ldots, m; e = 1, \ldots, n$ denotes m data samples each with n attributes. Individual data samples are also referred to as n dimensional vectors and the attributes as *variables*. $X_{i,j}$ can take on any real value consistent with its semantics. The notation $X_{i,*}$ denotes the ith data sample, $X_{*,e}$ denotes the col-

umn corresponding to the eth variable. $x = [x_1, \ldots, x_n]^T$ refers to a column vector not necessarily belonging to X. $\sigma^2(x)$ denotes the statistical variance of data vector x. The implicit assumption in this study is that any observed data can ultimately be represented in this quantitative form, that is, as a collection of vectors. C_m^l denotes a cluster identified by label l consisting of m data samples (cluster members), i.e., $|C_m^l| = m$. C^l denotes a cluster of unspecified size identified as l and $i \in C^l$ refers to the ith member in C^l. Similarly, C_m stands for an unlabeled cluster of size m such that $i \in C_m$ refers to ith element in it.

4 Measure of Dissimilarity

Implicit in the definition of patterns as a collection of *like* data points is the concept of distance/dissimilarity. Usually, such a measure is based on the features of the two data points between which it is computed. However, especially for problems such as character recognition, context-dependent dissimilarity measures are highly desirable. The motivation, theory and some applications of such measures based on the concept of *variance about a point* were introduced and applied to various pattern recognition problems in[20, 21, 22]. For the purpose of making this paper self-contained, the definitions and some properties of these concepts are reviewed in this section.

Definition 4.1 *The* variance about a point i *along* e*th dimension is defined as:*

$$\sigma_X^2(i,e) = \sum_{k=1}^{m} (X_{i,e} - X_{k,e})^2 \tag{1}$$

From equation(1), it can be seen that $\sigma_X^2(i,e) = 0$ if and only if $X_{i,e} = X_{j,e}\ \forall j \neq i$, i.e., if and only if all the values of eth attribute are identical. When this variance is different from zero, equation(1) is used to define the weight of the ith element along the component e, denoted by $w_{i,e}$ in equation(2).

$$w_{i,e} = (\sigma_X^2(i,e))^{-1} \text{if } \sigma_X^2(i,e) \neq 0 \tag{2}$$

The concept of variance about a point captures information intrinsic to the character's image representation. It can be proved that it is closely related to the statistical variance along eth component through equation(3).

$$\sum_{k=1}^{m} \frac{1}{w_{k,e}} = 2m^2\sigma^2[X_{*,e}] \tag{3}$$

Thus, *individual weights distribute the variance in the column based on the variance about each point.* The discrepancy between ith and jth point with respect to column $X_{*,e}$, that takes into account the context, i.e., *all* the other points in the column, is defined as $w_{i,e}[X_{i,e} - X_{j,e}]^2$.

Definition 4.2 *The context based* divergence *between data points $X_{i,*}$ and $X_{j,*}$ is defined as in equation(4).*

$$D(i,j) = \frac{1}{n}\sum_{e=1}^{n} w_{i,e}[X_{i,e} - X_{j,e}]^2 \tag{4}$$

The $m \times m$ matrix $D = D(i,j)_{i,j}$ denotes the collection of pairwise divergence between $X_{i,*}$ and $X_{j,*}$.

The following properties of divergence can be easily verified:

1. $0 \leq D(i,j) \leq 1$;
2. Asymmetry: In general, $D(i,j) \neq D(j,i)$.
3. Sum normalized: $\sum_{j=1}^{m} D(i,j) = 1$. Thus, the divergence of each point is distributed over all the other points in context.
4. $D(i,j)$ does not qualify as a *distance* because, in general, it does not satisfy the triangle inequality $D(i,k) \leq D(i,j) + D(j,k)$, hence the name divergence.
5. The divergence is invariant to linear scaling and constant addition. That is, for scalars a_e and b_e, if $S'_{*,e} = a_e \times S_{*,e} + b_e$ and $D' = D'(i,j)$ denotes the corresponding divergence matrix, then $D' = D$. (Proof omitted here)

Equation(4) is used to define the similarity between ith and jth data points as defined in equation(5).

$$S(i,j) = f_\alpha(D(i,j));\ \alpha > 0 \tag{5}$$

where f_α denotes a non-increasing function of $D(i,j)$ with with $f_\alpha(0) = 1$, $f_\alpha(1) = 0$ with the parameter α determining the function f_α. For example, in[20] and in this paper,

$$S(i,j) = 1 - D(i,j)^\alpha \tag{6}$$

with $\alpha = 1$ is used. It should be noted that the asymmetric nature of divergence makes similarity between any two quantities directional. More precisely, $S(i,j) \neq S(j,i)$. Therefore, in the current study, the similarity between i and j refers to the directional similarity between i and j denoted by $S(i,j)$ which will be different from $S(j,i)$ in general. The similarity in equation(6) can be used in clustering algorithms, implementing both unsupervised and supervised strategies. The former, in conjunction with a cluster validation algorithm was used in[23, 20]. The latter, that is, a supervised approach, is developed in the current study based on the cluster structure by defining various recognition strategies.

5 Clusters and Cluster Structure

In the current study, a pattern is considered as *a well-defined structure of interest in observed data*. The structure could be spatial, qualitative, or of any other type which could be used in information processing. The structure in the data is captured in *cluster*, that consists of a set of data samples sharing common properties. The data samples in a cluster should be as similar as possible (or from the same known category), and clusters themselves should be as dissimilar as possible from each other. A cluster is summarized by two parameters: 1) Weights of the cluster members; and 2) Cluster Integrity.

5.1 Cluster Member Weights

Weights of cluster members are based on the idea that not all the members in the cluster are equally significant from the point of view of measuring properties of a cluster. The members which are more similar to other members are assigned higher weights than others.

Definition 5.1 *The* **weight**, ρ_k *of the data point* k *in* C_m *is defined as the normalized similarity of this data-point to all the remaining data-points in* C_m. *More precisely,*

$$\rho_k = \begin{cases} 1 & \text{if } m = 1 \\ \frac{\sum_{l=1, l\neq k}^{m} S(l,k)}{\sum_{j=1}^{m}\sum_{l=1, l\neq j}^{m} S(l,j)} & \text{otherwise.} \end{cases} \quad (7)$$

Equation(7) implies that $\sum_{k=1}^{m} \rho_k = 1$ which captures the idea of completeness of the cluster based on its individual members. In the course of cluster formation and use, other measures of similarity are needed. These include the directional *data-to-cluster*, *cluster-to-data*, and *cluster-to-cluster* similarities as defined in equation(8).

$$\begin{aligned} S(v, C_m) &= \sum_{j=1}^{m} \rho_j S(v, j) \\ S(C_m, v) &= \sum_{j=1}^{m} \rho_j S(j, v) \\ CS_{12} &= \sum_{i\in C^1} \rho_i \sum_{j\in C^2} \rho_j S(i, j) \\ CS_{21} &= \sum_{j\in C^2} \rho_j \sum_{i\in C^1} \rho_i S(j, i) \end{aligned} \quad (8)$$

where v denotes a data point of dimension n.

5.2 Cluster Integrity

One of the most important properties of cluster is its tightness which reflects the discrepancies between its members. Popular measures for measuring tightness include the sum of square error criterion and scatter matrices[24]. It is also referred to as coherence, within cluster distance, or the degree of variability. To measure the tightness of the cluster, the current study introduces the notion of *cluster integrity*.

Figure 1: A Digit 3 Training Instance

Definition 5.2 Cluster integrity *is defined as the normalized average of the similarity between every pair of cluster members. More precisely,*

$$I(C_m) = \begin{cases} \frac{\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}(S(i,j)+S(j,i))}{m(m-1)} & \text{if } m \geq 1 \\ 1 & \text{if } m = 1. \end{cases} \quad (9)$$

6 Supervised Character Recognition

The training data is assumed to contain k non-overlapping clusters denoted by $C^1_{m_1}, C^2_{m_2}, \ldots, C^k_{m_k}$ such that C^i consists of m_i instances of character α_i ($i = 1, \ldots, k$) and $\sum_{i=1}^{k} m_i = m$ is the total number of training instances. The assumption that the clusters are non-overlapping is valid because the training data consists of categorized data in which a given training instance belongs to only one character. To avoid any bias in training for a particular digit, the number of training samples for each digit are kept constant ($m_1 = m_2 = \cdots = m_k$). For each cluster C^i, let $I(C^i)$ denote its integrity computed using equation(9) and ρ_{ij} denote the weight of the jth cluster member. The problem of character recognition consists in correctly identifying an unseen instance (denoted by t) of characters $\alpha_i, i = 1, \ldots, k$ using the clusters $C^i_{m_i}$ available during training phase. The principle idea is to measure the closeness of each test sample to each of the clusters using context dependent similarity described in earlier sections, and assigning t to the nearest cluster. This is analogous to the well established prediction techniques such as maximum likelihood based probabilistic models[25, 26] and minimum distance classifiers using, for example, Mahalanobis distance[27].

6.1 Data Set

The data set used for this work is a subset of NIST data set and consists of character images represented as

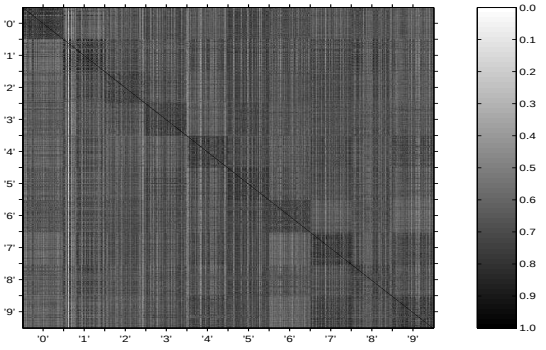


Figure 2: Similarity Matrix: Graphical Representation

a 28x28 matrix of binary digits (for example, Figure 1). The complete data set has 60000 training samples and 10000 test samples with different number of training and testing examples for each digit. The performance of the algorithm is analyzed using a learning curve (i.e., prediction accuracy for various sizes of training data) by randomly generated samples of training instances of difference sizes. In the current study, the number of samples for each digit for training are varied from 25 to 125 in steps of 25. For each sample size several random training sets are generated. Each of these training data is tested on a randomly generated test data which has 1000 samples taken from the entire test data.

6.2 Prediction Matrix

An error in prediction occurs when an instance t of character α_i is predicted as $\alpha_j (j \neq i)$. In the domain of character recognition, the frequency of an instance of α_i being predicted as α_j and vice-versa provides meaningful information in terms of the interrelationships between image representations of respective characters. In the current study, this information is captured in *prediction matrix* (denoted by p_N). More precisely, the (i,j)th entry of p_N, denoted by $p_N(i,j)$ is the percentage of instances of α_i predicted as α_j, when the training data had N instances of each character. Thus, the diagonal elements of p_N will represent the percentage prediction accuracy for each digit and for 100% accurate prediction, p_N will be a diagonal matrix with all diagonal entries equal to 100.

6.3 Training Phase

The training phase for the supervised character recognition algorithm consists of the following steps:

1. Each of the $p \times p$ character instance images is represented as a vector of p^2 dimensions obtained by concatenating p rows (here p=28);
2. The $m \times m$ similarity matrix for entire data set is computed using equation(6);
3. Cluster parameters, i.e., cluster integrity and member weights, are computed for $C^i; i = 1, \ldots, k$.

An image representation of a 500×500 similarity matrix for training size of 500 digits (i.e., 50 for each digit) is shown in Figure 2 where darker regions in the image correspond to higher similarity (see spectrum on the right). The dark straight line forming the principal diagonal stands for $S(i,i) = 1$ in the similarity matrix. The fifty instances of each digit have been grouped together and arranged in an increasing order of digits from left to right and top to bottom direction. The asymmetric nature of similarity is very well depicted in Figure 2. Relatively darker squares along the diagonal represent the similarities corresponding to the instances of the same digit.

6.4 Testing Phase and Results

Testing phase consists of using clusters obtained in the training phase to predict categories of unseen instances. In the current study, several different prediction strategies are employed as described next:

6.4.1 Prediction based on Cluster Integrity

Whenever a cluster is augmented by a new member, its cluster integrity will increase or decrease to reflect the similarity of this new member with the existing cluster members. Thus, this change in the cluster integrity can be used for cluster prediction using equation(10).

$$Category(t) = \underset{i}{argmax} (I(C^i_{m_i} \cup t) - I(C^i_{m_i})) \qquad (10)$$

In other words, the cluster whose integrity changes the most in positive direction after inclusion of t is the predicted category of t. Table 1 shows the prediction matrices for the integrity change method for various selections of training instances per digit. The quality of the clusters formed during the training phase have a direct effect on the prediction accuracy. Table 2 shows relative integrities of the clusters corresponding to different digits for various training sample sizes. The entry 100 implies that the integrity of this cluster was the maximum while 0 implies that the integrity was minimum. The cluster for 7 seems to have very high integrity values when per digit sample size is below 50, but as we increase it to 125 the cluster quality deteriorates. This implies that the variability in the samples for 7 increases as the sample size increases. The clusters corresponding to 0 and 6 are consistently better than others irrespective of the sample size leading to relatively higher prediction accuracies for

Table 1: Percentage Accuracy: Integrity Change Method

S = 50									
95.0	0.99	0.31	0.21	0.47	0.42	0.68	0.16	1.6	0.26
0.046	67.0	4.0	0.60	4.0	0.55	1.7	1.7	18.0	1.9
3.1	2.5	76.0	0.76	0.38	0.90	0.81	1.0	14.0	1.0
0.90	0.67	4.8	79.0	0.24	3.2	0.048	1.0	9.5	0.90
0.16	2.0	0.88	0	87.0	0.052	1.6	0.052	2.7	5.7
0.98	1.3	0.69	6.1	0.86	71.0	1.2	0.63	14.0	3.0
4.3	1.6	0.25	0.20	5.6	4.3	80.0	0	3.3	0
0.047	4.3	4.6	3.7	3.1	0.42	0	68.0	7.7	8.0
1.5	3.5	1.7	0.50	1.8	2.0	0.35	1.2	82.0	5.6
0.36	0.77	0.46	0.31	7.8	0.26	0	2.4	10.0	77
S =75									
95.0	0.66	0.29	0.15	1.4	0.37	0.66	0.15	1.4	0.15
0	73.0	3.2	0.31	6.1	0.43	0.93	1.9	13.0	1.4
2.6	2.6	75.0	0.38	0.53	0.83	0.90	0.83	15.0	0.68
0.94	1.1	4.8	77.0	0.27	2.6	0.13	0.88	11.0	1.3
0.21	2.9	0.36	0	86.0	0.071	1.7	0.14	2.4	6.2
1.4	1.6	0.24	6.7	1.4	67.0	1.6	0.56	16.0	4.1
4.4	1.6	0	0.23	7.0	2.8	81.0	0	2.8	0.23
0.068	4.8	6.3	2.6	3.2	0.48	0	68.0	6.6	8.0
0.80	3.9	1.1	0.15	2.8	1.7	0.15	0.95	83.0	5.2
0.35	1.2	0.50	0.64	7.6	0.28	0	2.8	12.0	75
S =100									
95.0	0.38	0.20	0.050	0.68	0.38	0.45	0.23	2.1	0.15
0.022	71.0	3.8	0.13	5.4	0.87	1.5	1.9	15.0	1.3
2.4	1.5	77.0	0.31	0.59	0.76	0.52	1.1	14.0	1.2
0.56	0.25	5.2	79.0	0.15	2.5	0.025	1.1	10.0	1.2
0.17	2.9	0.32	0	88.0	0.27	1.8	0.025	2.1	5.0
1.4	1.0	0.33	6.4	2.1	71.0	1.0	0.75	13.0	3.2
4.2	1.3	0.16	0.16	7.8	2.7	81.0	0	2.1	0.026
0.025	4.1	5.5	3.9	3.8	0.25	0	68.0	6.5	8.1
1.0	2.8	1.5	0.38	2.3	1.6	0.18	1.1	84.0	5.3
0.19	0.57	0.64	0.57	8.6	0.26	0	1.9	11.0	76
S = 125									
96.0	0	0.13	0	0.40	0.67	0.40	0.27	2.3	0
0	69.0	4.2	0	5.9	1.5	1.0	1.7	15.0	1.2
2.9	1.8	77.0	0.26	0.52	0.91	1.0	0.65	14.0	1.2
0.65	0.13	4.7	78.0	0.13	2.7	0.13	0.78	12.0	0.78
0	2.4	0.53	0	86.0	0.53	1.6	0	2.9	5.9
1.0	1.5	0.15	6.5	2.2	71.0	0.87	0.44	13.0	3.8
4.4	1.6	0.13	0.13	9.7	2.6	80.0	0	1.4	0
0	3.4	4.8	3.5	3.9	1.1	0	70.0	5.4	7.9
0.55	3.3	1.8	0.28	1.5	1.9	0.41	0.69	84.0	5.4
0.12	0.99	0.50	0.37	8.1	0.37	0	2.6	13.0	74

S=No. of training examples per digit
(i, j) entry = %age of digit $(i-1)$ predicted as $(j-1)$

Table 2: Average Relative Integrities of Clusters

S = 25		S = 50		S = 75		S = 100		S = 125	
I	D	I	D	I	D	I	D	I	D
0	1	0	8	0	1	0	1	0	8
7	8	6	1	6	8	0	8	1	1
26	4	19	4	22	4	22	4	25	4
37	9	45	9	52	5	44	3	31	3
41	5	48	5	56	2	45	2	45	7
45	2	50	2	57	9	50	5	45	2
63	3	65	3	61	3	56	9	51	5
79	0	71	0	76	7	63	7	56	9
84	6	85	7	88	0	76	0	76	0
100	7	100	6	100	6	100	6	100	6

S=No. training examples per digit
I = Integrity; D= Digit

all the sample sizes. The clusters for 1 are of the lowest integrity which is directly reflected in poor prediction accuracy. The prediction matrices show the relative number of misprediction for each pair of digits. (It is interesting to note that the prediction matrix is highly asymmetric in that in general $p_N(i,j) \neq p_N(j,i)$).

Equation(8) defined in section 5 measures inter-cluster similarities. The matrix shown in Table 3, shows relative asymmetric similarities between ten clusters (one per digit) for various training sample sizes averaged over a large number of randomly chosen training samples. A zero entry in the matrices means that the similarity was minimum within than matrix. Similarly, the maximum similarity within the matrix is 100 and all the other inter-cluster similarities have been scaled between 0 and 100 for illustration purposes. The following facts can be observed immediately:

1. In *all* matrices, the directional similarity $S(1,0)$ between digits 0 and 1 was minimum and $S(0,1)$ was relatively low;

2. The entries $S(9,7)$ and $S(7,9)$ have very high values which shows high similarity between clusters 7 and 9. This also explains lower prediction accuracy of these digits and corresponding large percentage of mispredictions (7 being predicted as 9 and vice-versa);

3. The context dependent directional similarity approach was successful in distinguishing between digits "3" and "8" as manifested by medium values for $S(3,8)$ and $S(8,3)$;

4. A very high similarity $S(6,5)$ is not reciprocated by smaller values for $S(5,6)$.

Moreover, a change in the number of samples does not affect the *relative* pair-wise similarities. This property is significant in those supervised learning problems where

Table 3: Normalized Inter-cluster Similarities

S = 50									
54	18	48	64	39	95	69	46	58	35
2	54	67	53	52	83	47	68	76	36
24	59	54	65	33	87	53	78	61	33
31	37	56	54	31	98	42	68	64	50
19	48	37	44	54	86	36	76	59	78
35	39	52	71	46	54	50	65	65	44
58	52	66	63	45	99	54	27	61	8
8	45	64	63	58	87	0	54	63	81
25	58	52	64	46	92	39	68	54	58
15	32	38	63	78	84	0	100	72	54
S = 75									
55	20	48	66	40	95	71	46	55	37
0	55	66	53	52	82	48	69	75	37
23	61	55	65	33	86	53	79	59	35
30	39	55	55	31	98	43	69	63	51
18	51	36	44	55	85	37	77	58	79
34	42	50	72	46	55	52	65	63	47
57	55	65	64	44	100	55	28	59	10
6	50	64	64	59	86	1	55	62	82
22	62	51	65	47	91	40	69	55	61
14	35	37	64	78	85	0	99	70	55
S = 100									
55	20	49	67	38	95	71	46	56	37
0	55	65	56	51	83	48	70	74	37
25	62	55	68	31	87	54	78	59	34
32	39	55	55	29	98	45	68	63	50
19	52	36	46	55	85	38	77	58	79
35	43	50	74	45	55	54	66	63	45
57	55	65	66	43	100	55	28	59	10
8	50	63	65	58	86	3	55	61	80
25	62	51	67	45	90	41	68	55	59
16	36	37	65	76	84	2	98	70	55
S = 125									
53	19	50	65	38	92	69	46	53	38
0	53	65	53	50	80	46	69	73	38
26	61	53	65	30	84	53	76	57	34
30	38	53	53	30	96	42	69	61	50
19	50	38	46	53	84	38	76	57	76
34	42	50	73	42	53	53	65	61	46
57	53	65	65	42	100	53	26	57	11
7	50	61	65	57	84	3	53	61	80
23	61	50	65	46	88	42	69	53	57
15	34	38	65	76	84	3	96	69	53

S = No. of training examples per digit
(i,j)th entry = similarity between C^{i-1} and C^{j-1}
Diagonal entries are significant.

number of training samples is limited. The direct correspondence between the entries of this matrix and those in the prediction matrices provides a lot of insight into the underlying relationships between the clusters, which is not the case with maximum likelihood based probabilistic approaches that are not as transparent to users interested in investigating the inter-cluster relationships and draw inferences based on them.

6.4.2 Prediction Based on Cluster Member Weights

Equation(8) defined in section 5 provides a direct way to measure vector to cluster and cluster to vector similarity using weights of the cluster members. This leads to the following two approaches for prediction:

Global Weights Based Approach:

1. Compute the divergence and similarity matrix for the entire training set;
2. Compute the weights of the cluster members $\rho_{ij}^{global}, j = 1, \ldots, m_i$ for each cluster $C_{m_i}^i; i = 1, \ldots, m_i$;
3. Compute new similarity matrices $S'_1, \ldots, S'_k$ such that S'_z is the similarity matrix for the members of cluster $C_{m_z}^z$ and test sample t;
4. The predicted category for t is computed using equation(11)

$$Category(t) = \underset{i}{argmax}\, min(S_z(t, C_{m_i}^i), S_z(C_{m_i}^i, t))$$
$$\text{where } S(t, C_{m_i}^i) = \sum_{j=1}^{m_i} \rho_{ij}^{global} S'_z(t, j);$$
$$S(C_{m_i}^i, t) = \sum_{j=1}^{m_i} \rho_{ij}^{global} S'_z(j, t) \quad (11)$$

The weights used in equation(11) are computed based on the *whole* training set, hence the name.

Local Weights Based Approach: The local weight based approach differs from the global weight based approach only in the computation of weights as described next:

1. Divide the training set into k clusters $C_{m_i}^i, i = 1, \ldots, k$;
2. Compute the new similarity matrices $S'_1, \ldots, S'_k$ such that S'_z is the similarity matrix for the members of cluster $C_{m_z}^z$ and test sample t;
3. Using these similarity matrices compute the *local* weights of the cluster members $\rho_{kj}^{local}, j = 1, \ldots, m_i$ for each cluster;

Table 4: Prediction Matrices for Global & Local Weight Methods

Global Weights Method

93.0	0.21	0.41	0.21	0.83	0.83	1.9	0.21	2.1	0.14
0.36	**22.0**	15.0	1.1	3.7	1.4	1.9	10.0	43.0	2.2
3.2	0.66	**77.0**	1.1	0.33	1.8	1.7	2.2	10.0	1.3
0.65	0.19	5.0	**80.0**	0.19	4.5	0.19	1.7	6.5	1.5
0.21	1.3	0.41	0	**79.0**	0.75	2.5	0.34	3.0	13.0
1.5	0.37	0.81	8.0	1.7	**56.0**	9.1	1.5	16.0	5.5
4.9	0.69	0.41	0.34	6.2	2.2	**83.0**	0	2.0	0.21
0	1.3	8.4	4.2	2.9	0.51	0	**64.0**	6.6	13.0
2.2	1.1	3.2	1.3	2.0	3.9	0.92	1.3	**77.0**	7.4
0.41	0.41	0.48	0.75	7.4	0.82	0	7.4	10.0	**72.0**

Local Weights Method

92.0	0.62	0.34	0.21	0.96	0.83	2.0	0.21	2.3	0.21
0.30	**68.0**	4.3	1.0	3.8	0.90	1.0	2.3	16.0	2.2
3.0	1.2	**76.0**	0.66	0.46	1.6	1.3	1.4	13.0	1.3
0.71	0.45	4.0	**80.0**	0.19	2.8	0.13	1.3	9.5	1.3
0.21	1.8	0.41	0	**85.0**	0.68	2.1	0.21	2.7	7.0
1.5	1.8	0.59	6.9	1.8	**64.0**	1.8	0.88	16.0	4.6
4.4	1.6	0.48	0.27	6.7	2.6	**81.0**	0	2.5	0.21
0	4.7	7.1	4.0	2.9	0.32	0	**62.0**	8.7	10.0
2.2	3.6	1.3	1.2	2.0	2.5	0.59	0.79	**79.0**	7.0
0.27	1.6	0.41	0.61	7.7	0.55	0	4.0	11.0	**73.0**

4. The predicted category for t is computed using equation(12)

$$Category(t) = \underset{i}{argmax}\ min(S_z(t, C^i_{m_i}), S_z(C^i_{m_i}, t))$$
$$\text{where}\quad S(t, C^i_{m_i}) = \sum_{j=1}^{m_i} \rho^{local}_{ij} S'_z(t, j);$$
$$S(C^i_{m_i}, t) = \sum_{j=1}^{m_i} \rho^{local}_{ij} S'_z(j, t) \tag{12}$$

In global weight based approach, the weights are calculated independent of the test data while in the local weight based approach the weights are changed by addition of test sample. Table 4 summarizes the results for local and global weight based approaches (Only p_{25} shown for illustration). These results are in agreement with the results obtained using the integrity based method in that the best results are obtained for the digit 0 and the worst for 1. But the accuracy of these methods is lower than that of the integrity change method. The following advantages of the prediction matrix are evident: 1)It is a concise summarization of entire results; and 2) Certain characteristics of the prediction such as 9 being predicted as 7 and vice-versa can be spotted out quickly.

Table 5: Summary of Results

S ↓	Digits →	0	1	2	3	4	5	6	7	8	9
25	IC	93	44	78	81	79	55	83	64	73	72
	GW	93	22	77	80	79	56	83	64	77	72
	LW	92	68	76	80	85	64	81	62	79	73
50	IC	95	43	78	79	76	69	83	70	79	77
	GW	95	24	78	79	76	68	83	68	80	77
	LW	95	67	76	79	87	71	80	68	82	77
75	IC	95	43	78	75	77	65	83	64	81	76
	GW	95	31	78	75	77	65	82	62	83	76
	LW	95	73	75	77	86	67	81	68	83	75
100	IC	95	43	79	75	80	69	83	58	79	78
	GW	95	26	79	74	80	70	83	58	82	77
	LW	95	71	77	79	88	71	81	68	84	76
125	IC	**96**	49	78	76	80	73	82	47	77	78
	GW	**96**	24	78	74	80	73	82	46	83	78
	LW	**96**	69	77	78	86	71	80	70	84	74

IC=Integrity Change; GW=Global Weight
LW=Local Weight; S=No. of training examples per digit

6.5 Discussion of the Results

The diagonal elements of $p_{25}, p_{50}, p_{75}, p_{100}, p_{125}$ are shown in Table 5 that summarizes results for all three prediction strategies. In all the cases, the prediction for digit "0"zero was the best compared to other digits. This can be attributed the peculiar characteristic of the cluster for digit 0 (C^0), that contains training data samples consisting of a large number of zeros at and around the center of the binary matrix image representation. The high cluster integrity for C^0 in Figure 2 and Table 2 are consistent with the results and lead to the conclusion that context based divergence matrix correctly models cluster for individual clusters. Consistent worst prediction for digit "1" was investigated by manually observing actual image representations of digit "1" for several samples. Slant effect appeared to be dominant in this case because digit 1 appeared in several different slant angles, which leads to poor cluster formation in the training data. Based on their similarities with respect to the shapes, the digits 7 and 9 are the hardest to distinguish. The number of times an instance of 9 is predicted as 7 and vice-versa is substantially high. Very low values for entries (7,0), (4,3), (9,6) and the like demonstrates that the proposed view of cluster structure based on asymmetry indeed succeeds in quantifying the underlying relationships between different clusters correctly.

6.6 Comparison with Mahalanobis Distance based Classifier

Mahalanobis distance [27], derived from the covariance between attributes of multidimensional vectors, has been used in several minimum distance classifiers [28, 29]. The distance of a column vector v from a cluster C^i_m is given by equation(13).

$$d(v, C^i_m) = (v - \mu)'(\sum)^{-1}(v - \mu) \tag{13}$$

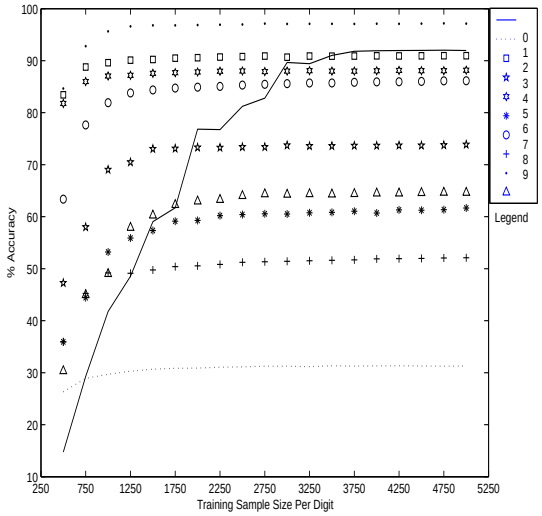


Figure 3: Learning Curve for Mahalanobis Distance

where μ is the column vector of means of the attributes values and $\sum^{-1}$ is the $n \times n$ covariance matrix. A minimum Mahalanobis distance classifier assigns a test sample t to the cluster that minimizes $d(v, C_m^i)$ in equation(13).

The results for character recognition of the data set using Mahalanobis distance are summarized by the learning curve in Figure 3. One of the preconditions for using the Mahalanobis distance is $m > n$ to guarantee invertibility of the the covariance matrix and validity of matrix product in equation(13). (In fact, a popular choice is $m > 2n$.) Keeping this in mind, the training sizes for each digit used in the formation of clusters in the training phase were kept at least 500. The results obtained using the context dependent divergence and associated ideas in this paper have not used as many training instances owing to the time intensive nature of the algorithm, which show a bias for the Mahalanobis distance and a natural expectation that Mahalanobis distance will be more accurate.

The learning curve for Mahalanobis distance classifier for digit 0 improves considerably when more training samples are available and then saturates at maxima of 92%, which is a little less than the results for context dependent similarity based results. The maximum accuracy for 1 is just 31% which is considerably less than the divergence based approach. Overall, the Mahalanobis approach is better predictor for digits 2,4 and 8. In all the other cases divergence based approach consistently outperforms minimum distance classifier based on Mahalanobis distance.

6.7 Complexity and Ongoing work

The main complexity of the framework lies in computation of the divergence matrix which is of the order $O(m^2E)$ for m samples of E dimensions. Although it

Table 6: Prediction Accuracy for Mahalanobis Metric

D → S ↓	0	1	2	3	4	5	6	7	8	9
750	29	29	89	58	86	44	78	45	93	45
1000	42	30	90	69	87	53	82	49	96	49
1250	49	30	90	70	87	56	84	49	97	58
1500	59	31	90	73	88	57	84	50	97	60
1750	62	31	90	73	88	59	85	50	97	62
2000	77	31	91	73	88	59	85	51	97	63
2250	77	31	91	73	88	60	85	51	97	63
2500	81	31	91	73	88	60	85	51	97	64
2750	83	31	91	73	88	61	85	51	97	64
3000	90	31	91	74	88	61	86	51	97	64
3250	89	31	91	74	88	61	86	52	97	64
3500	91	31	91	74	88	61	86	52	97	64
3750	92	31	91	74	88	61	86	52	97	64
4000	92	31	91	74	88	61	86	52	97	64
4250	92	31	91	74	88	61	86	52	97	65
4500	92	31	91	74	88	61	86	52	97	65
4750	92	31	91	74	88	61	86	52	97	65
5000	92	31	91	74	88	62	86	52	97	65

D = Digits; S = No. of training examples per digit

makes the framework time intensive, computation of context requires at least one comparison of each sample with every other, which has complexity $O(m^2)$. It is assumed that there is only one cluster per digit, which prohibits representation of the same digit in more than one clusters. But this can be handled by identifying different clusters of the same digit as different characters and following the proposed approach unchanged. In future, the cases when more than one representations exist for a character can also be dealt with. The ongoing work consists in extension of the framework for image processing and shape modeling in conjunction with other machine learning approaches and significant results in these areas have already been obtained.

7 Conclusion

This study presented a new way of context consideration based on an asymmetric measure of dissimilarity and applied it to develop a supervised character recognition algorithm. The approach conceptually differs from previous works based on asymmetric metrics by keeping track of the available contextual information through context based similarities and other cluster parameters. The prediction accuracy outperforms Mahalanobis distance based classifier in several instances. A very small training size (125 samples of each from total of 60000 samples) gives accuracies well above 80% (and as high as 96%) which underscores the success of the approach. The associated cluster structure and transparent prediction

schemes offer a lot of insights into actual relationships between data, which is not the case with several popular recognition schemes.

Acknowledgments

Waibhav Tembe's work was supported by a Graduate Fellowship from the Ohio Board of Regents. Anca Ralescu's work was partially supported by the Grant N000140310706 of the Department of Navy. The authors thank Dr. Shigeo Abe of Kobe University for sharing the data set used in this study.

References

[1] Casey R. and Lecolinet E. A survey of methods and strategies in character segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):690–706, July 1996.

[2] Abe S. *Support Vector Machines for Pattern Classification*. Kobe University, Kobe, Japan, 2002.

[3] Duda R. and Hart P. *Pattern Classification and Scene Analysis*. John Wiley, 1973.

[4] Gader P. and Khabou M. Automated feature generation for handwritten characters. In *Proceedings of Third International Workshop on Frontiers of Handwritten Recognition*, Buffalo, NY, 1993.

[5] Gader P., Gillies A., and Chaing J. *Handwritten Character Recognition*, pages 223–261. 1994.

[6] Rocha J. and Pavlidis T. Character recognition without segmentation. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 17(9):903–909, 1995.

[7] Martins B. Recognition of isolated hand-printed characters. In *IEEE conference on Systems, Man and Cybernetics*, volume 3, pages 2238–2243, 1996.

[8] Koshinaka T., Nishiwaki D., and Yamada Keiji. A stochastic model for handwritten word recognition using context dependency between character patterns. *Proceedings of the Sixth International Conference on Document Analysis and Recognition*, pages 154–158, 2001.

[9] Smith S., Bourgoin M., Sims K., and Voorhees H. Handwritten character classification using nearest neighbor in large databases. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):915–919, September 1994.

[10] Kovacs Z., Guerrieri R., and Baccarani G. A novel metric for nearest-neighbor classification of hand-written digits. *Proc. of International Conference on Pattern Recognition, IEEE Computer Press*, 2:96–100, 1992.

[11] Kim D., Kim E., and Bang S. A variation measure for handwritten character image data using entropy difference. *Pattern Recognition*, 30(1):19–29, 1997.

[12] Kovacs-V Zs., Guerrieri R, and Baccarani G. A novel metric for nearest-neighbor classification of hand-written digits. *Proc. of 11th International Conference on Pattern Recognition*, 2:96–100, 1992.

[13] Kato N., Omachi S., Aso H., and Nemoto Y. A handwritten character recognition system using directional element and asymmetric mahalanobis distance. *IEEE trans. PAMI*, 21(3):258–262, 1999.

[14] Kittler J., Hatef M., Duin R., and Matas J. On combining classifiers. *IEEE Transactions on Pattern Extraction and Machine Intelligence*, 20:226–239, March 1998.

[15] L. Xu, A. Krzyzak, and C. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 22(3):418–435, 1992.

[16] Kornai A., Mihiuddin K., and Connel S. An hmm-based legal amount filed ocr system for checks. *IEEE Intl. Conf. on Systems, Man and Cybernetics*, 1995.

[17] A. Kundu, He Y., and Bahl P. Recognition of handwritten words: 1st and 2nd order hidden markov model approach. *Pattern Recognition*, 22(3):283–297, 1989.

[18] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[19] Vapnik V. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

[20] Tembe W. and Ralescu A. Context dependent clustering of numerical data with application to a biological dataset. In *Proceedings of International Conference on Artificial Intelligence and Soft Computing*, May 2001.

[21] Tembe W. and Ralescu A. Context based correlation for supervised learning. In *Proceedings. of 14th Midwest Artificial Intelligence and Cognitive Science (MAICS 2002)*, April 2003.

[22] Tembe W. and Ralescu A. Context based modeling of planar shapes - application to handwritten character recognition. In *Proceedings of 10th International Conference on Information Processing and Management of Uncertainty*, 2004.

[23] Tembe W. Pattern extraction using a context dependent measure of divergence and its validation. Master's thesis, University of Cincinnati, Cincinnati, USA, 2001.

[24] Duda R., Hart P., and Stork D. *Pattern Classification and Scene Analysis*. John Wiley-Interscience, 2000.

[25] Everitt B. and Dunn G. *Applied Multivariate Data Analysis*. Oxford University Press, 1992.

[26] Manning C. and Schutze H. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[27] Mahalanobi P. On the generalized distance in statistics. In *Proceedings of National Institute of Science, India*, number 12, 1936.

[28] Jain A. and Dubes R. Algorithms for clustering data. Prentice Hall, New Jersey, 1988.

[29] Everitt B. *Cluster Analysis*. Halsted Press, 3 edition, 1993.

GEVOSH: Using Grammatical Evolution to Generate Hashing Functions

Patrick Berarducci Demetrius Jordan David Martin Jennifer Seitzer
Computer Science Department, University of Dayton
{ berardpb | jordandt | martindw | seitzer }@udayton.edu

ABSTRACT

In this paper, we present system GEVOSH, *Grammatically Evolved Hashing*. GEVOSH evolves hashing functions using grammatical evolution techniques. Hashing functions are used to expedite search in a wide number of domains. In our work, GEVOSH created hashing functions that, on average, perform better than many standard (human-generated) hash functions extracted from the literature. In this paper, we present the architecture of system GEVOSH, its main components and algorithms, and resultant generated hash functions along with comparisons to standard, human-generated functions.

1 INTRODUCTION

Grammatical evolution is a type of evolutionary programming where the output of such a system (i.e., the output being a computer program) is not limited to the LISP programming language. Our system, *Grammatically Evolved Hashing* (GEVOSH) uses the method of Grammatical Evolution to generate hashing functions. Grammatical Evolution is a method of machine generating computer programs of any arbitrary computer language, so long as that language has an associated BNF (Backus-Naur Form) grammar [Ryan, 1998].

Hashing is a method of mapping a wide range of various types of keys into a smaller space for quick retrieval of information. The GEVOSH system produces hash functions by using a diminutive C++ grammar as its input, creating hash functions in this specified language, and evolving the hash functions by applying the genetic operators of mutation and cross over. This section describes Grammatical Evolution, BNF grammar, and hashing functions. Section 1.1 will give a brief explanation of Grammatical Evolution. Section 1.2 will give a brief explanation of BNF Grammar. Section 1.3 will give a brief explanation of hashing.

1.1 GRAMMATICAL EVOLUTION

The precursors of Grammatical Evolution are Genetic Algorithms (GA) and Genetic Programming (GP). Genetic Algorithms is an area of artificial intelligence that can be described as "a search algorithm based on the mechanics of natural selection and natural genetics" [Stanford]. Genetic Programming is "the method of creating computer programs using genetic algorithms" [Hyper Dictionary]. Grammatical Evolution (GE) can be defined as a grammar based genetic algorithm, the purpose of which is to generate executable programs.

GE is clearly similar to GP. GE, however, has the distinction that its input is a grammar in Backus Naur Form (BNF) which allows GE's to generate programs in any language, of arbitrary complexity, including loops, multiple line functions, etc. The use of BNF also allows GE systems to more closely model real world DNA [Ryan, 1998] [Brabazon, 2002] [O'Neill, 2002]. The following table (***figure*** 1) helps to explain the close relationship between GE and biology. The GE part of the table is in terms of our particular system, though our system is very similar to many other GE systems [Nicolau, 2002].

	Biology	Grammatical Evolution
Genetic Material:	DNA	Integer Arrays
Specification of building blocks of protein:	Codon (group of 3 nucleotides)	Integers.
Transcription:	DNA to RNA Codons within RNA translated to determine the sequence of amino acids contained within protein molecule	Process of changing integer arrays to BNF. Mapping of integer arrays to BNF in order to determine the actual program
Results:	Expression of genetic material as proteins in conjunction with environmental factors is the phenotype	Phenotype is a running program.

Figure 1: The Relationship between Grammatical Evolution and Biology

1.2 BNF GRAMMAR

Backus Naur Form or BNF is a context free meta-language. In particular, it is "a formal metasyntax used to express context-free grammars" [Unicode]. Thus, it is the notation used to specify programming languages.

1.3 HASHING

The following definitions are necessary to our discussion on hashing.

Definitions: [Flajolet, 1998]

Collision – the mapping of two or more keys to the same index. The number of *collisions* associated with an index is determined by the number of distinct keys that are mapped to that index.

Seek – an attempt to locate an item when a collision has occurred. The number of *seeks* associated with an item is determined by the number of attempts looking at alternative positions after a collision has occurred.

Linear Probing – the attempt to store a key sequentially after a collision has occurred. For example, if a key is hashed to the 6^{th} position, a linear probe will look to the 7^{th} position to see if the value can be placed there.

Hashing is a method of mapping a large range of keys into a smaller range of indices for compact storage and quick retrieval of information. Thus, the desirable hash function is one that generates few (if any) collisions, and provides the ability to find information without exhaustively searching a table of records.

2 THE GEVOSH SYSTEM

The GEVOSH system is designed to generate static hashing functions. The GE system depicts each member as representing a unique hash function. The fitness of a member is determined by the percentage of keys hashed without a collision. The higher the fitness grade, the better the hash function. The fitness grade is used to rank population members, and ultimately determine which members of the population will survive through the next generation and which will die off.

2.1 COMPONENTS OF GEVOSH

System GEVOSH has 4 main components as follows:

1. A module to load the BNF grammar
2. A module to decode a member and write the hash function
3. A module to gather statistics about the function
4. A module to evolve the members of the population

The relationship of these modules can be seen in *Figure 2* below:

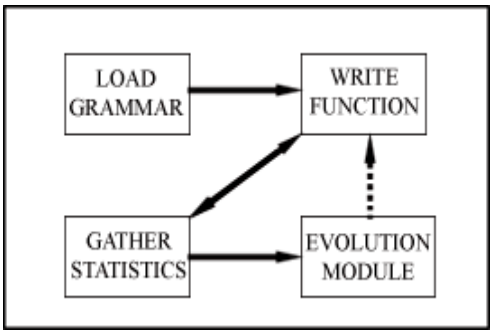


Figure 2: GEVOSH System Diagram

At system startup, the BNF Grammar is loaded.

2.1.1 LOAD GRAMMAR MODULE

The Load Grammar Module opens a file specified by the user and reads in the BNF. As the file is read, the Module stores the grammar in to a grammar data structure. This structure is similar to adjacency lists that are commonly used in graph structures.

The System uses a subset of the C++ grammar. This subset can be seen in *Figure 3*. After the Grammar is loaded into the system it then moves on to the Write Function Module.

	Ident1	Ident2	Ident3	Ident4	Ident5
	<op>	<var>	<num >	<statement-line>	<expr>
R0	+	value	<num > <num >	<statement-line> <statement-line>	<num>
R1	-		0	<var> = <expr> ;	<var>
R2	*		1	<var> <op>= <num>;	<expr> <op> <expr>
R3	/		2	<var> <op> = <expr>;	(<expr>) <op> <expr>
R4	%		3	<var> = (<expr>) <op> (<expr>) ;	
R5	>>		4	<var> = (<expr>) <op> <num> ;	
R6	<<		5		
R7			6		
R8			7		
R9			8		
R10			9		

Figure 3: Grammar used in the GEVOSH system

2.1.2 WRITE FUNCTION MODULE

The Write Function Module decodes a member of the population to decide which grammar rules to follow, and ultimately, what to write out for the function. The first activity of the write function is to decide which template to use for the function.

There are five templates from which to choose. The templates are rather simplistic; they can be seen in *Figure 5*. The use of a template is a commonly used technique in GE [Ryan, 1998]. The main motivation for using a template is to simplify the grammar and reduce the size of the genetically evolving integer array. This smaller array thus increases the chance of compilation of the program that it (the integer array) derives.

After the template is selected, the system will read in identifiers to be parsed (or resolved). The system resolves the identifier by following the BNF rule number that is given by taking the integer at position i modulo the number of rules for the given identifier. An example of this resolution can be seen below in *Figure 4.*

Memeber:	**19**	**26**	**13**	**39**	**48**	**6**

	Value	Number of Rules	Rule to Follow
<Statement-Line>	19	6	1
<var> = <expr> ;		1	0
value = <expr> ;	26	4	2
value = <expr> <op> <expr> ;	13	4	1
value = <var> <op> <expr> ;		1	0
value = value <op> <expr> ;	39	7	4
value = value % <expr> ;	48	4	0
value = value % <num> ;	8	11	8
value = value % 7 ;			

Figure 4: Resolution of <statement-line>

As can be seen in *Figure 4*, if an identifier has only one associated rule, there is no choice, thus it does not use an integer--it just outputs the rule. This resolution process is done for all of the identifiers in the template.

Once the Write Function Module is complete, the system then moves on to the Gather Statistics Module.

2.1.3 GATHER STATISTICS MODULE

The Gather Statistics Module ascertains the merit of the newly created hash function by using and measuring the hash function in a prewritten secondary program. The secondary program is able to embed the hash function by using an included external file (to which the hash function was written). The secondary program takes three variables as parameters; the number of values to be hashed (*n*), the number at which to seed the random number generator with (*i*), and the file to output the number of seeks and collisions.

The secondary program first seeds the random number generator with *i*. After the random number generator has been seeded it hashes *n* values. For each value hashed, it is stored into an array, in order from the first value randomly generated to the nth value randomly generated. This program keeps track of the number of collisions and seeks, and outputs them to the specified file. The collision resolution method used in this program is linear probing. The algorithm for this program can be seen in section 2.1.3.1.

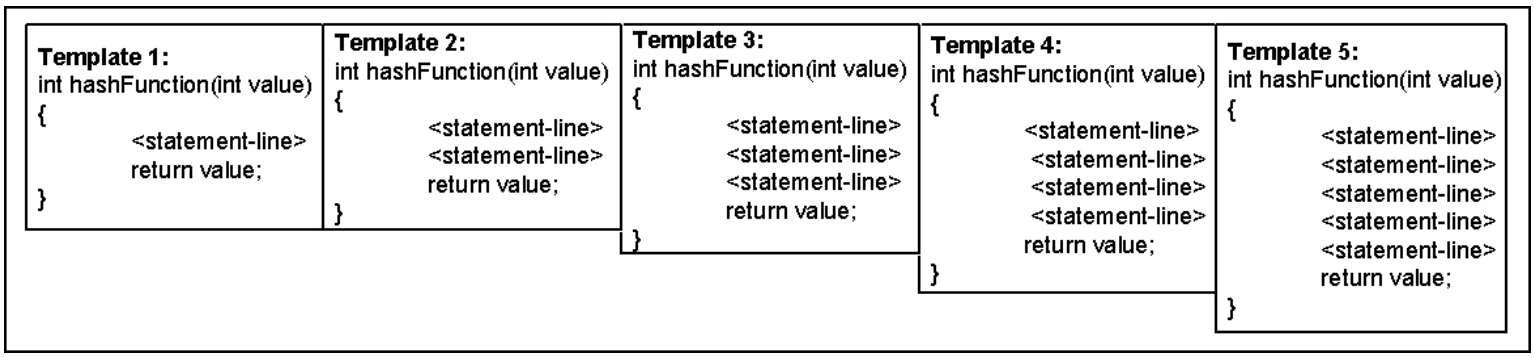


Figure 5: Templates used in the GEVOSH system

The Gather Statistics Module compiles and runs the secondary program containing a newly generated hash function. If the program does not compile or does not run that means that the function that was written in the Write Function Module is not a valid function and will be penalized with a fitness value of 0. If the secondary program does successfully compile and run, the statistics for that hash function are stored. The system will then either go back to the Write Function Module for the next member or, if it is at the last member the system it will then move on to the Evolution Module which can be found in section 2.1.4.

2.1.3.1 SECONDARY PROGRAM

The algorithm for the secondary program is as followed:

Begin secondary program- Algorithm()
  Seed random number generator with *i*
  Set *j* = 0
  Set *l* = 7*n*/6
  Make *array* of size *l*
  For 0 <= *j* < *n*
    Set *value* = random number
    Set *key* = hash_function(*value*)
    Set *key* = *key* modulo *l*
    If something at *array* position *key*
      Set *collisions* = *collisions*+1
    While something at *array* position *key*
      Set *key* = (*key* + 1) modulo *l*
      Set *seeks* = *seeks* + 1
    End while
  End For
  Output *collisions* and *seeks*
End secondary program

2.1.4 EVOLUTION MODULE

In this module the system assigns fitness values to the members of the population (i.e., the current generation of hash functions). Additionally, it is in this module that augmentations to the population occur (i.e., new hash functions are generated). The GEVOSH system augments all of the members of the population except for the member with the highest fitness value. The reason the system does not augment the member with the highest fitness value is because that is one of the rules for convergence in an evolutionary system [Fogel, 2000].

For each hash function, the fitness score is based on the percentage of input numbers hashed that did not result in a collision. As the system evolves the hashing functions, the higher the percentage of numbers hashed not resulting in a collision, the better. After all members are evaluated, they are then collectively used to create new hash functions by invoking the mutation and/or crossover operators to spawn new members in the population.

The GEVOSH system uses finite number of members for each generation, thus new members replace current members of the population. The only member that is guaranteed to survive from one generation to the next, is the member with the highest fitness score.

2.1.4.1 CROSSOVER AND MUTATION

One method of crossover used is the random selection of a splitting point, a number between 1 and 3/4ths of the total length of the integer arrays. Once this splitting point is selected, the system randomly selects two parents from the population. The parents are then split at the selected splitting point, and recombined to make a new member of the population. One method of mutation, on the other hand, is the random selection of one or more array positions to be changed by "flipping" a bit in the integer. That is, if the integer had a binary 1 to be flipped, it would be mutated to contain a binary 0, and vice versa.

2.2 MAIN ALGORITHM OF GEVOSH

Begin GEVOSH-Algorithm()
  Load Grammar
  Randomly Generate N members
  Do Until fitness equals 100 or forced to quit
    For each member of the population
      Write function
      Compile secondary program
      Run secondary program
      Input statistics
    End For
    Compute fitness value
    Evolve population
  End Do
End GEVOSH

3 RESULTS

We are pleased to announce that our system generated two hashing functions that show extremely promising potential. That is, on average, these two functions in some ways are *superior* to six commonly used hashing functions that we extracted from the literature [Wang, 2002]. Each of these hash functions produces collisions that are close in value with each other. We are testing to see how the number of seeks and collisions compares to these hash functions.

The first hash function we use is a simple method that uses the values it will hash and the total number of values it will hash. The function takes the value modulo total to generate a key, where value is the value to bee hashed, and total is the total amount of space in the hash table.

The second hash function we use is Robert Jenkins' 32 bit Mix Function. The function uses a sequence of addition, exclusive-or, and bit shifts to produce a key [Wang, 2002].

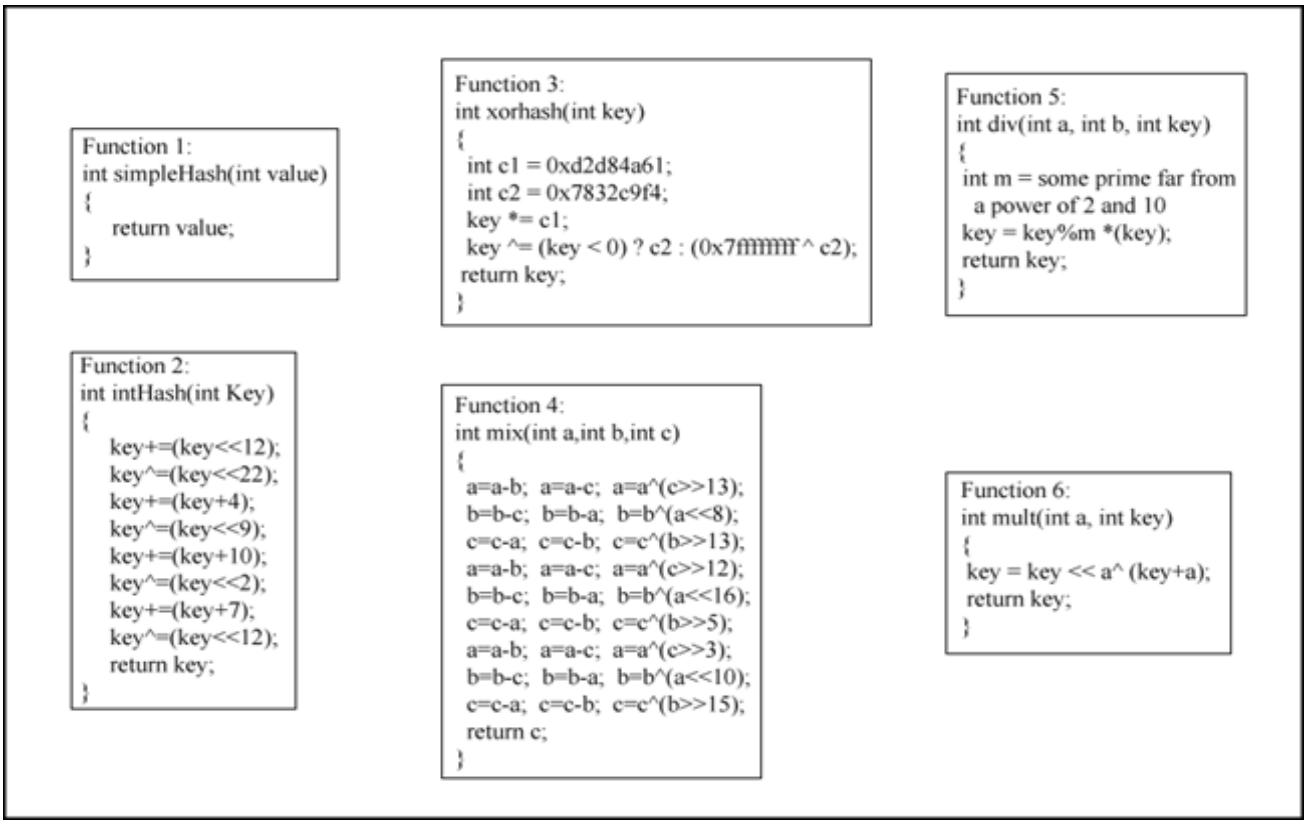
The third hash function uses multiplication and a substitution box to generate keys. The input for the function is the high bit position of multiplication. The function uses the exclusive-or operator on the substitution box value with the value of the multiplication result [Wang, 2002].

The fourth function is Robert Jenkins' 96 bit Mix Function. The function uses subtraction, exclusive-or, and bit shift operations. It takes a key as input, along with two random bits the program initializes before it calls the function. The function has nine rows of operations and each row acts on a variable. This mixes information from each of the variables. The function applies nine bit-shift operations and shifts each key to the right sixty-one bits in total, and thirty-four bits to the left [Wang, 2002].

The fifth hash function uses division with a prime number to generate keys. It uses a prime number that is not close to a power of two or a power of ten. The function uses integer division to divide the key by the prime number. It then multiplies the number by the prime number and subtracts the result from the key [Wang, 2002].

The sixth hash function uses left-shift and exclusive-or operations. The program initializes a random integer and passes it to the function. The function uses the exclusive-or operation. The program initializes a random integer and passes it to the function. The function uses the exclusive-or operation with the random integer and the random integer added to the key. It then takes this result shifts the key to the left this amount of times [Wang, 2002].

The algorithms for these hashing functions can be found in figure 6. Note that the functions return values that with be taken modulus the size of the hash table to complete the mapping of the key.



```
Function 1:
int simpleHash(int value)
{
    return value;
}
```

```
Function 2:
int intHash(int Key)
{
    key+=(key<<12);
    key^=(key<<22);
    key+=(key+4);
    key^=(key<<9);
    key+=(key+10);
    key^=(key<<2);
    key+=(key+7);
    key^=(key<<12);
    return key;
}
```

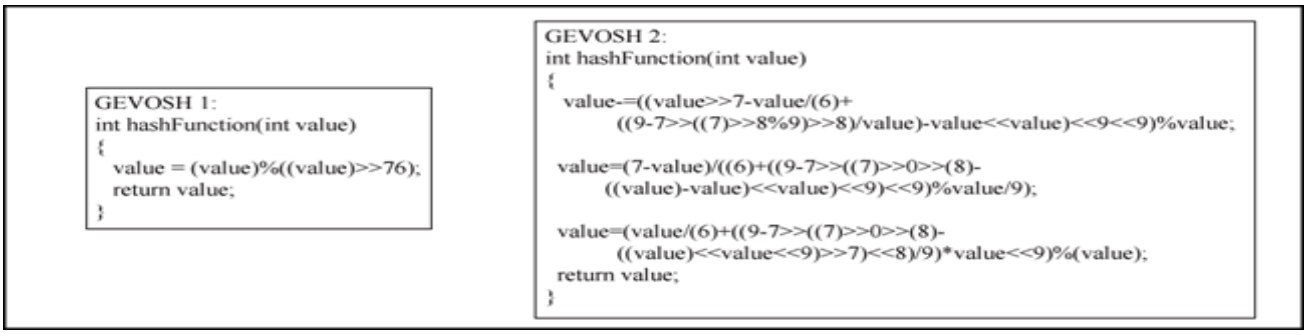
```
Function 3:
int xorhash(int key)
{
 int c1 = 0xd2d84a61;
 int c2 = 0x7832c9f4;
 key *= c1;
 key ^= (key < 0) ? c2 : (0x7fffffff ^ c2);
 return key;
}
```

```
Function 4:
int mix(int a,int b,int c)
{
 a=a-b;  a=a-c;  a=a^(c>>13);
 b=b-c;  b=b-a;  b=b^(a<<8);
 c=c-a;  c=c-b;  c=c^(b>>13);
 a=a-b;  a=a-c;  a=a^(c>>12);
 b=b-c;  b=b-a;  b=b^(a<<16);
 c=c-a;  c=c-b;  c=c^(b>>5);
 a=a-b;  a=a-c;  a=a^(c>>3);
 b=b-c;  b=b-a;  b=b^(a<<10);
 c=c-a;  c=c-b;  c=c^(b>>15);
 return c;
}
```

```
Function 5:
int div(int a, int b, int key)
{
 int m = some prime far from
   a power of 2 and 10
 key = key%m *(key);
 return key;
}
```

```
Function 6:
int mult(int a, int key)
{
 key = key << a^ (key+a);
 return key;
}
```

Figure 6: Compared Hash Functions



```
GEVOSH 1:
int hashFunction(int value)
{
  value = (value)%((value)>>76);
  return value;
}
```

```
GEVOSH 2:
int hashFunction(int value)
{
  value-=((value>>7-value/(6)+
        ((9-7>>((7)>>8%9)>>8)/value)-value<<value)<<9<<9)%value;

  value=(7-value)/((6)+((9-7>>((7)>>0>>(8)-
        ((value)-value)<<value)<<9)<<9)%value/9);

  value=(value/(6)+((9-7>>((7)>>0>>(8)-
        ((value)<<value<<9)>>7)<<8)/9)*value<<9)%(value);
  return value;
}
```

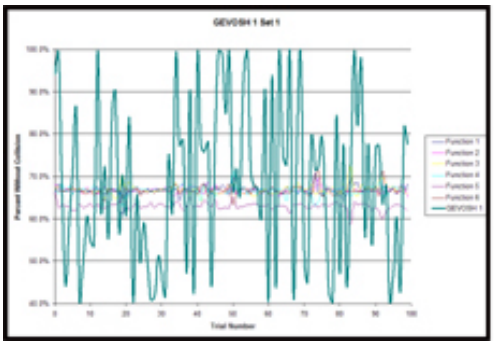
Figure 7: GEVOSH System Created Hash Function

The two functions that were created by the GEVOSH system that we intend to show results for will be labeled GEVOSH 1 and GEVOSH 2, this functions can be seen below in the *Figure 7.*

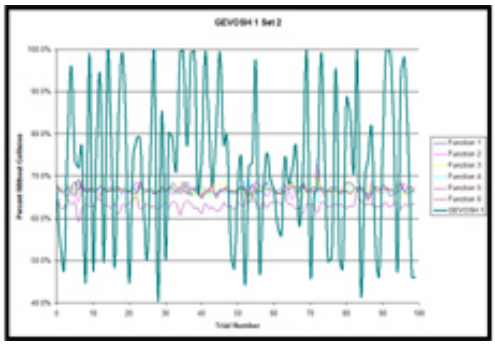
As of now we have not simplified GEVOSH 2, but GEVOSH 1 is a very interesting function. Though value is shifted over more that the number of bits that are contained in an integer all of the test run so far we have been able to map and retrieve keys without a problem. The three sets of results that are to be shown are from three different architectures. Data Set 1 ran on an Intel® Pentium 3 1.6 MHz processor with Linux® Redhat version 7.3 as the operating system, Data Set 2 ran on an Intel® Pentium4 2.4 MHz processor with Windows® XP as the operating system, and Data Set 3 ran on a Unix architecture Sun® Solaris. As can be seen from the charts below on average GEVOSH 1 out performs the six tester functions. Each data set consists of 100 randomly generated sets of data,

with varying sizes of data which is also selected at random.

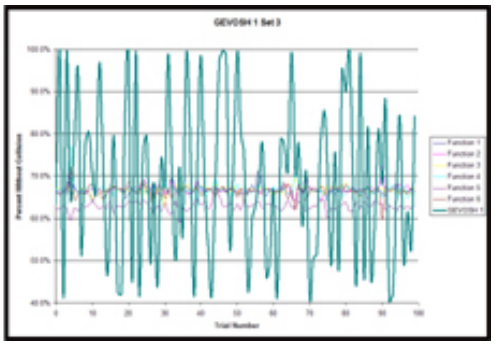
Even though GEVOSH 1 out performs all other hash functions on average, it can be seen from below on the graphs that the function is not stable. That is, the generated function seems to be dependent on the data. In the six other functions are not dependent on the data. The standard deviation of GEVOSH 1 is 0.186 where the standard deviation of the six testing functions are all approximately 0.01. Though GEVOSH 2's average percent without collisions is much lower than all of the other functions, its standard deviation is approximately .02, which means that GEVOSH 2 is a lot less dependent on the Data. These results are promising, since the GEVOSH system can produce both functions with high averages and low standard deviations. All of charts and graphs with the results listed in this section can be found below.



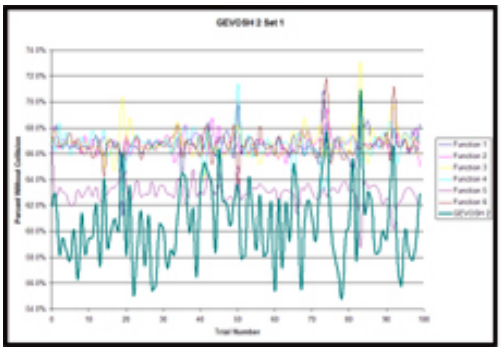
Graph 1: GEVOSH 1 Data Set 1



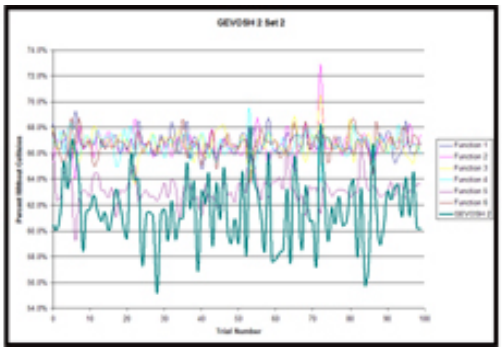
Graph 2: GEVOSH 1 Data Set 2



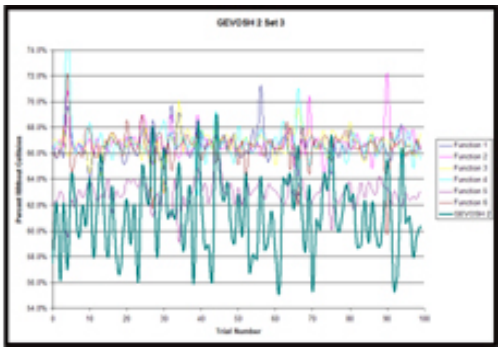
Graph 3: GEVOSH 1 Data Set 3



Graph 4: GEVOSH 2 Data Set 1



Graph 5: GEVOSH 2 Data Set 2



Graph 6: GEVOSH 2 Data Set 3

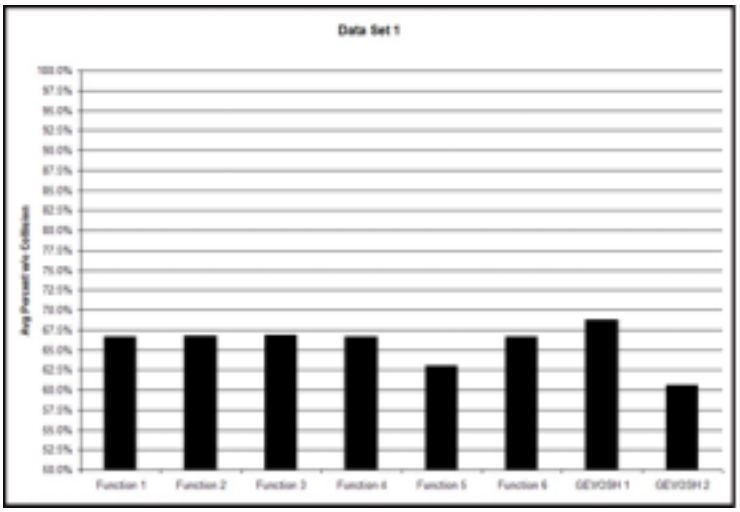


Chart 1: Linux® Redhat 7.3

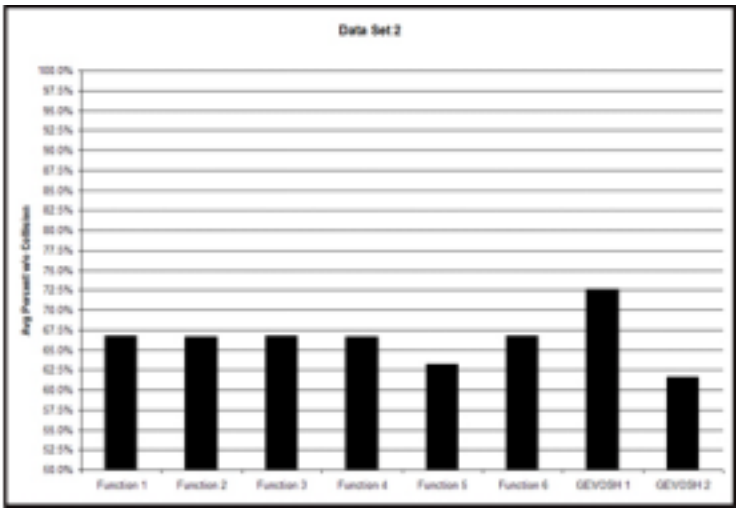


Chart 2: Windows® XP

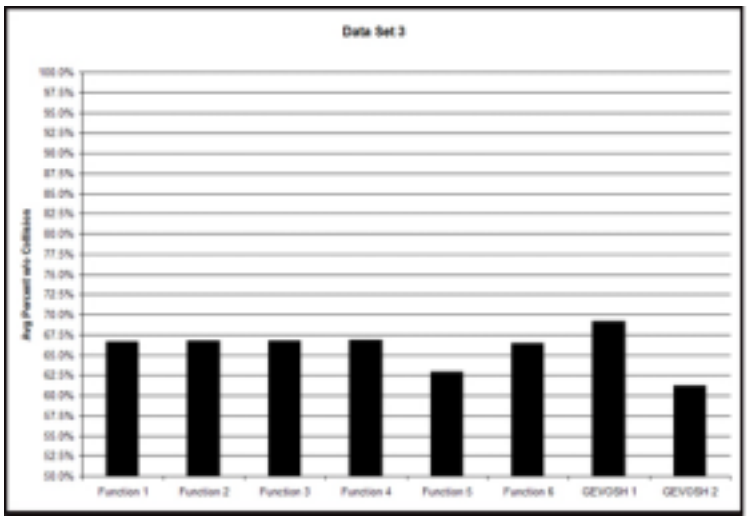


Chart 3: Sun® Solaris

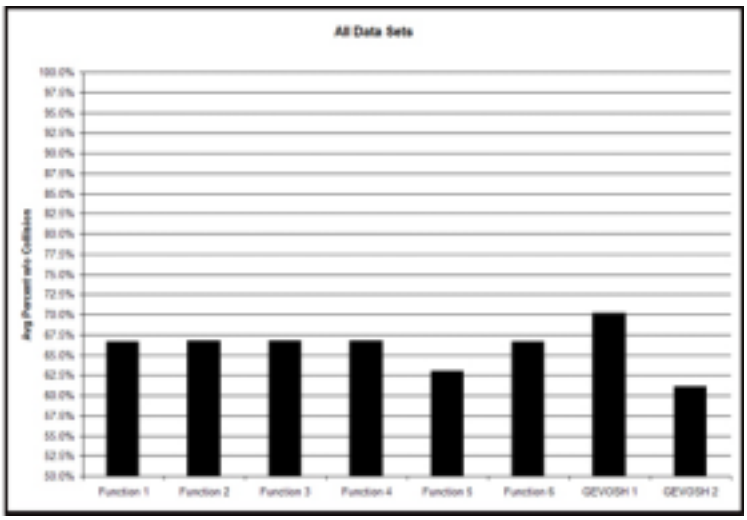


Chart 4: Average Over All Data Sets

4 CONCLUSIONS & FUTURE WORK

We believe that the GEVOSH system is showing promising results in that we have found functions that had a high average of keys hashed without a collision and a function that has a relatively small standard deviation. We believe that as we continue, and with a few modifications to the system, we will be able to find a function that contains both of these favorable characteristics.

In future work, we plan on attempting to generate universal hash functions with our grammatical evolutionary system. We are also planning on looking at generating hash functions that can help with cryptography. At this point, we believe that using Grammatical Evolution to evolve hashing functions will produce unique hashing functions that will be hard to reverse engineer. We will also work on mixing languages to see how that will affect our system. We do not believe that it will affect performance in any way but might be interesting to see the results it might come up with.

We believe that the GEVOSH system is exhibiting promising results and, as in all evolutionary computation systems, are continuing to run GEVOSH through days and days of execution to evolve even stronger hash functions.

REFERENCES

[Ryan, 1998] Ryan C., Collins J.J., O'Neill M. (1998). Grammatical Evolution: Evolving Programs for an Arbitrary Language; EuroGP 1998

[Brabazon, 2002] Brabazon A., Matthews R., O'Neill M., Ryan C. Grammatical Evolution and Corporate Failure Prediction; GECCO 2002

[Corman, 1990] T. Cormen, C. Leiserson, R. Rivest. *Introduction to Algorithms;* The MIT Press, 1990.

[Nicolau, 2002] Nicolau M., Ryan C. LINKGAUGE: Takling hard deceptive problems with a new linkage learning genetic algorithm; GECCO 2002

[O'Neill, 2002] O'Neill M., Ryan C. Investigations into Memory in Grammatical Evolution; GECCO 2002

[Fogel, 2000] Fogel D. Evolutionary Computation *Toward a New Philosophy of Machine Intelligenc;* IEEE Inc.

[Flajolet, 1998] Flajolet P. On the Analysis of Linear Probing Hashing; France Algorithms Seminar 1998

[Wang, 2002] Wang T. Integer Hash Functions; http://www.concentric.net/~Ttwang/tech/inthash.htm

[Stanford]
www.stanford.edu/~buc/SPHINcsX/bkhm15.htm

[Hyper Dictionary]
http://www.hyperdictionary.com/computing/genetic+programming

[Unicode]
http://www.unicode.org/glossary/

Experiments with Machine Learning for Logic Arguments Identification

Vasile Rus

Intelligent Systems Laboratory
Department of Computer and Information Sciences
Indiana University - South Bend
South Bend, IN 46634
{vasile@cs.iusb.edu}

Abstract

This paper reports experiments on applying machine learning to detect functional arguments of verbs such as logical subject. In particular, it is shown that using decision trees for functional arguments detection is beneficial. The paper also argues that linguistically-motivated features gathered from a large corpus can capture functional information.

Syntactic functional information is vital for advanced text understanding technologies such as information extraction, machine translation, question answering and others. Supervised methods are one of the most used and most successful approaches to develop text understanding systems in which in the first phase a group of experts manually annotate a large collection of text with a targeted type of information (in the form of tags/metadata) producing a *corpus* and then, in the second phase, learning methods are developed that use the corpus as a source of supervision to assess their performance. The general method presented here uses Treebank II (Marcus, Santorini, & Marcinkiewicz 1993), a corpus annotated with syntactic information, for learning logic functional arguments.

Despite that Treebank includes functional tags (e.g. LGS for LoGical Subject) in its annotation tags, modern syntactic parsing technologies generated from it offer only surface syntactic information in the form of a bracketed representation in which main constituents and major structural phrases in a sentence are identified. From the bracketed representation produced by syntactic parsers one can easily identify the surface syntactic subject, nevertheless the logical subject requires further processing as illustrated in Table 1: the bracketed form for the two examples is the same (it was generated with Collins' statistical parser (Collins 1996)) and thus a surface level pattern that tags as subject the first NP in a (S (NP VP)) [1] phrasal structure would wrongly tag *something* as the subject of *tell* [2].

[1] NP stands for noun phrase, VP for verb phrase and S for sentence.

[2] *Something told John* is the inverted form of *John told something* a frequent sentential form in Treebank.

To overcome the inability of modern parsing technology to detect the underlying logical syntactic structure , novel methods are necessary that offer accurate, robust and scalable solutions to the problem of finding syntactic functional information.

In this work a model is introduced which is then used to induce automated tools able to detect functional information (logical) in English sentences. The tools are obtained using the C4.5 package for decision tree induction.

In addition, this paper argues that linguistically-motivated features gathered from a large corpus can capture functional information.

Related Work

Usually, when syntactic information is used to study a certain linguistic problem, people either use the bracketed form and are happy with surface level syntactic information or have their own pattern-based methods which lack generality and scalability. In (Stetina, Kurohashi, & Nagao 1998) a general method for word sense disambiguation is presented which yields high performance by taking advantage of full sentential context including surface syntactic information as provided by modern parsers (little extra processing is done on their part). (Lapata 1999) proposed a technique which acquires alternating verbs from large balanced corpora by using partial-parsing methods. As part of the process syntactic patterns to guess, i.e. heuristics, the double object frame, for instance, are applied on top of a parser's output: if the syntactic pattern contains at least two proper names adjacent to each other (e.g. *killed John Kennedy*) then reject. Similarly, in (Stevenson & Merlo 1999), a set of heuristics are used to find counts of transitive frames for verbs: a number, a pronoun, a determiner, an adjective, or a noun were considered to be indication of a potential object of the verb.

We already mentioned that Treebank II includes functional tags in its annotation guidelines (about 20 of them that can be appended to constituent labels, e.g. NP-LGS to denote a noun phrase which is also a logical subject). Those tags were manually inserted in Treebank and play an important role in the development and evaluation of automated tools to assign

Sentence	Bracketed Form	Grammatical Function
Something told John.	(S (NP (NN something)) (VP (VBD told)) (NP (NNP John)))	l-sbj=John d-obj=something
John told something	(S (NP (NNP John)) (VP (VBD told)) (NP (NN something)))	l-sbj=John d-obj=something

Table 1: Examples of similarly bracketed sentences by state of the art parsers but with different underlying logical structure

functional labels to different constituents in a sentence.

The present work is similar to approaches presented in (Pradhan *et al.* 2003), (Gildea & Hockenmaier 2003), (Chen & Rambow 2003) to address the problem of shallow semantic parsing - the process of annotating texts with semantic roles specified either using predicate specific labels (Baker, Fillmore, & Lowe 1998) or predicate independent labels (Kingsbury, Palmer, & Marcus 2002). They address the problem of shallow semantic parsing as a classification problem using a diversified pool of formalisms to induce a classifier (Support Vector Machines, Decision Trees) and sets of features (the sets used by different approaches have many features in common). Our work is similar to those approaches in three ways: (1) we address the task of detecting logic roles (as opposed to semantic roles) as a classification problem (2) we use a set of features similar, at some extent, to those used by the mentioned studies (3) the induced classifier plays an important role in a natural language based knowledge representation: the Logic Form (Rus 2002) and Propbank (Kingsbury, Palmer, & Marcus 2002), in our case and for the mentioned approaches, respectively.

The PropBank project adds a layer of semantic annotation to the Penn English TreeBank. It provides a consistent argument labeling of predicates, particularly verbs, participial modifiers and nominalizations. In the sentence, *John broke the window* the *breaking* event is described by the following argument structure **break(John, window)**. In PropBank, the arguments of the verb are labeled sequentially from ARG0 to ARG5, where ARG0 is usually the subject of a transitive verb; ARG1, its direct object, etc. Adjunct arguments are also marked for temporals and locatives.

In (Moldovan & Rus 2001) a natural language based knowledge representation, namely the Logic Form (LF), is presented which requires logic functional arguments for its predicates. In (Rus 2002) the logic functional arguments were detected using a set of structural patterns that were well suited for small English sentences (definitions of concepts in electronic dictionaries) but for open text that approach lacks scalability.

LF is first order, syntactically simple, logic and was used in many language processing applications. (Davidson 1967) proposed the predicate treatment of verbs and then (Hobbs 1983) applied this concept to automated text processing, particularly interpretation of texts. (Rus 2002) employed LF to Question Answering to search answers that were not explicitly stated in supporting documents.

In LF a predicate is generated for every noun, verb, adjective or adverb . The name of the predicate is a concatenation of the word's base form and its part-of-speech.

An example on how this is done is illustrated for the following sentence:

The Earth provides the food we eat every day.

Its corresponding logic form (LF) is:

Earth:n_(x1) & provide:v_(e1, x1, x2) & food:n_(x2) & we:n_(x3) & eat:v_(e2, x3, x2; x4) & every:a_(x4) & day:n_(x4)

In the example given above, the verb *provides* is mapped onto the predicate *provide:v*, where *provide* is the base form for *provides* and *v* stands for verb (the part of speech of *provide*). Arguments for predicates are of two types: *e* - for events specified by verbs, respectively *x* - for entities. The LF of the entire sentence is the conjunction of individual predicates.

The argument's position is also important as it encodes syntactic information: the second argument for a verb is syntactic subject, the third is direct object and the fourth is indirect object. For instance, the second argument of the predicate *provide:v_(e1, x1, x2)* is *x1* (Earth), the subject of the providing event. Arguments after ; (semicolon) are adjuncts (arguments which are not mandatory to convey the message of the sentence - such as time or place) and were introduced in (Rus 2003).

The Role Assignment Problem

In the following sentence:

The judges hear or try a court case.

there are two verbs *hear* and *try* in a coordination (indicated by preposition *or*) and one direct object *a court case*. The issue is whether the direct object is shared by the two verbs or not. In other words, out of two possible interpretations which one is the correct one:

I1: *The judges (hear or try) a court case.*
LF1: judge:n_ (x1) & hear:v_ (e1, x1, x2) & or (e3, e1, e2) & try:v_ (e2, x1, x2) & nn (x2, x3, x4) & court:n_ (x3) & case:n_ (x4)
I2: *The judges hear or (try a court case).*
LF2: judge:n_ (x1) & hear:v_ (e1, x1) & or (e3, e1, e2) & try:v_ (e2, x1, x2) & nn (x2, x3, x4) & court:n_ (x3) & case:n_ (x4)

In I2 the direct object is *attached* only to the second verb (the verb *hear* being in an intransitive subcategorization frame). For this particular instance the correct interpretation is I1, i.e. the compound noun *court case* is shared. An example in which the direct object should not be shared is:

They steal or commit a violent act.

It is very unlikely to state *steal a violent act* and thus the correct interpretation is:

I: *They steal or (commit a violent act).*
LF: they (x1) & steal:v_ (e1, x1, x2) & or(e2, e1, e3) & commit:v_ (e3, x1, x2) & violent:a_ (x2)

Here is an example of two verbs that do not share the subject, a situation which might be interpreted otherwise at first sight.

An aircraft that has a fixed wing and is powered by propellers or jets.

The reason why the noun *aircraft* is not subject for both verbs, *has* and *power*, is the changing of voice in the coordination, leading to the case where the second verb has its own subject introduced by the preposition *by*.

This paper illustrates how a machine learning method can solve the RA problem. LFI is a timely topic in language processing as proven by the LFI competition as part of SENSEVAL-3 (www.senseval.org) with 26 participants from both academia and industry from all over the world.

Approach

Our approach is to address the RA issue as a classification problem: given a verb in a sentence and a candidate phrasal head the task is to find the most appropriate syntactic role the head plays for that verb. The set of possible roles contains: subject, direct object, indirect object, prepositional object or norole (a value which indicates that the candidate head does not play any role for the given verb). To preview our results, we demonstrate that combining a set of indicators automatically extracted from large text corpora provides good performance.

The key to any automatic classification task is to determine a set of useful features for discriminating the items to be classified. Observing the patterns of syntactic roles for verbs we derive useful features for our classification task.

Head The candidate head could indicate whether it is an appropriate filler for a specific syntactic role given the verb. From the previous examples, we know that *act* cannot stand as an object for *steal*. The verb *write* takes as subject a person or a referent to a person such as a personal pronoun.

Lexical Category of Head A noun is most likely to be direct object of, say verb *give* while a pronoun is most likely to be the subject or indirect object of the verb *write*. Lexical category (part of speech) can be extracted from an annotated corpus or obtained using an automated part of speech tagger. Here, since we use Treebank II the parts of speech are already annotated in the corpus and we only need to extract them.

Voice As we saw previously, the voice of a verb can play an important role on deciding what head word is the subject. In case the verb is in passive voice, most likely the subject is the prepositional object of the following *by*, if any such preposition is present in the sentence. We have a dozen patterns to detect the voice of the verb.

Type of Clause Here, we refer to the type of sentence (S, SINV, SBAR, SBARQ [3]) in which the verb is. Here is an example: *I do not believe it, said Peter DaPuzzo, head of retail equity trading stock prices during program-dominated trading.* where *said John* is an inverted sentence in which the subject follows the verb as opposed to the most common case *John said*. The type is read from the parse tree by following links from verb to parent up the parse tree until a S (S, SINV, SBAR, SBARQ) node is found.

Position of Head If the head is after or before the verb and at what distance in terms of words (punctuation is ignored when the position is determined).

Those features could be automatically extracted from large corpus, either manually annotated or automatically generated.

The basic steps of the feature extraction method are outlined below:

procedure *ExtractFeatures(Sentence)*
- Generate a full syntactic parse for the Sentence
- Stem the words in the sentence

for *verb in Sentence* **do**
- Extract the set of features for each node in the tree relative to the verb
- Classify each node as *norole* or as *logical subject*

end for

Experimental Setup and Results

There are three major issues that we need to address before performing any kind of experiments: what verbs to focus on, where should we gather training data from and what machine learning algorithm to use. In the next few paragraphs we provide answers for each of those issues.

Previous work on verb meaning research, such as (Korhonen, Gorrell, & McCarthy 2000) and (Ted & Carroll 1997), reported experiments on a set of 14 target verbs that exhibit multiple argument patterns: *ask, begin, believe cause, expect, find, give, help, like, move, produce, provide, seem, swing*. We adopted those 14 verbs since we believed it would be a good starting point to have a small set, on one hand, with many argument ambiguities, on the other hand, thus balancing challenges with manageability of the experiments.

Next, we looked for a corpus. Treebank (Marcus, Santorini, & Marcinkiewicz 1993) is a good candidate because it contains role annotations (in a limited form, though). We started by developing patterns

[3] S - regular sentence, SINV - inverted sentence, SBAR - relative clause, SBARQ - relative interogative clause

for tgrep, a tree retrieval pattern-based tool, to identify sentences containing target verbs from Wall Street Journal (WSJ) part of Treebank II (the version with part-of-speech tags) and used the online form to retrieve the data (http://www.ldc.upenn.edu/ldc/online/treebank/). The training set is further processed: a stemmer is applied to obtain the stem (base form) of individual words and then the target verb is identified and the features extracted. One or more training examples (positive and negative) are generated from a sentence (see next section for examples).

As learning paradigm we opted for decision trees. Decision Trees are a most popular method to approach classification problems. The attractiveness of decision trees is due to the fact that, in contrast to neural networks, decision trees represent rules. Rules can readily be expressed so that humans can understand them.

The C4.5, an algorithm for decision tree generation and an extension of ID3, is used to generate a classifier for the RA problem. The algorithm ID3 (Quinlan) uses top-down induction of decision trees.

Given a set of classified examples a decision tree is induced, biased by the information gain measure, which heuristically leads to small trees. The examples are given in attribute-value representation. The set of possible classes is finite.

Only tests, that split the set of instances of the underlying example languages depending on the value of a single attribute are supported. Depending on whether the attributes are nominal or numerical, the tests either have a successor for each possible attribute value, or split according to a comparison of an attribute value to a constant, or depending on if an attribute value belongs to a certain interval or not.

The algorithm starts with the complete set of examples, a set of possible tests and the root node as the actual node. As long as the examples propagated to a node do not all belong to the same class and there are tests left, a test with highest information gain is chosen, the corresponding set of successors is created for the actual node, each example is propagated to the successor given by the chosen test, ID3 is called recursively for all successors.

Some of the additional features of C4.5 over ID3 are: incorporation of numerical (continuous) attributes, nominal (discrete) values of a single attribute may be grouped together, to support more complex tests, post-pruning after induction of trees, e.g. based on test sets. In order to increase accuracy C4.5 can deal with incomplete information (missing attribute values).

One problem with decision trees is that they tend to overfit the training data, i.e. if tested on training cases from which it was constructed it may give a poor estimate of its accuracy on new cases.

The true predictive accuracy of the classifier can be estimated by sampling or by using a separate test file; either way the classifier is evaluated on cases that were not used to build it. However, this estimate can be unreliable unless the numbers of cases used to build and evaluate the classifier are both large.

One way to get more reliable estimate of predictive accuracy is by *k-fold cross validation*. The cases are divided into *k* blocks of roughly the same size and class distribution. For each block in turn, a classifier is constructed from the cases in the remaining blocks and tested on the cases in the hold-out block. The error rate of a classifier produced from all the cases is estimated as the ratio of the total number of errors on the hold-out cases to the total number of cases. Here, we predict the accuracy of our induced classifiers using *k=10* or 10-fold cross validation.

Next, we present two major experiments: (1) using the set of features presented before and (2) adding the verb as an extra feature. Each experiment has two sub-experiments: (i) traces in parse tree from Treebank II are not solved and thus the training and test data sets do not include examples for those traces (ii) traces are solved and the corresponding examples added to the training and test data sets.

Traces are extra tags or artificial nodes appended or introduced in intermediate nodes of a parse tree to mark the presence of a syntactic component which is physically missing: for example the trace *-1* in Figure 1 which denotes the relative clause starting at *SBAR*, indicates that the node *ICH-1* is a pointer or trace for SBAR, explicitly encoding that SBAR plays the role of direct object for the verb *know*. *ICH-1* is artificially introduced in the syntactic parse tree since no corresponding lexical (physical) item exists in the original sentence to justify its presence. It is rather implied.

Experiment 1

In this experiment we focus on identifying a specific syntactic role for a specific verb: the role values can be either **subject** or **none**.

In Figure 1 a parse tree is provided for the sentence: *Chris knew yesterday that Terry would catch the ball*. The numbers below individual words is the index of that word in the sentence, starting with 0.

We pick a verb, say *know*, and then from annotated corpora, training data is generated. For each word in the sentence a positive example is obtained if the word is the subject of the verb. Examples from different verbs are not mixed. Table 2 contains results for a first trial in which traces are not resolved. We pay special attention to punctuation (time stamps such as 10:40 are changed to 1040) and numbers (350,000 is changed to 350000) to comply with the input requirements for the C4.5. learner. In (Mihalcea 2002) commas, dots and columns are assigned a special notation (CO and DO respectively) but that was necessary because commas, dots and columns are features in their learning model (they used Timbl (Daelemans *et al.*), a memory-based learner, which accepts C4.5-like input). For some sentences there is no logical subject identified because it is unspecified as is the case in the following sentence: *The cars are quoted at 351000 and 350000, asked .*

Examples of training data for the sentence in Figure 1 are

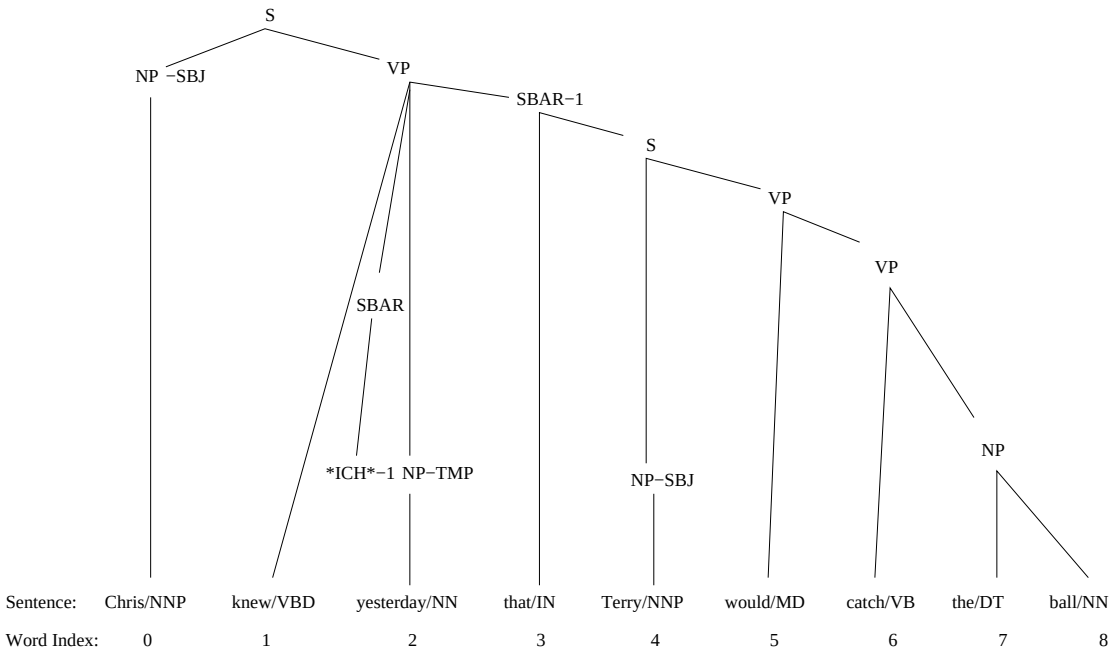


Figure 1: An example of parsed sentence in Treebank II

given below as one training instance per line:

know, Chris, NN, active, S, 1, subject
know, that, IN, active, S, 3, none
know, yesterday, NN, active, S, 4, none
know, that, IN, active, S, 5, none
know, Terry, NN, active, S, 6, none
know, catch, VB, active, S, 8, none
know, ball, NN, active, S, 10, none
catch, Chris, NN, active, S, -7, none
catch, know, VB, active, S, -6, none
catch, that, IN, active, S, -5, none
catch, yesterday, NN, active, S, -4, none
catch, that, IN, active, S, -3, none
catch, Terry, NN, active, S, -2, subject
catch, ball, NN, active, S, 2, none

As we notice entries for *that* are also generated for where they are resolved. For this particular tag ICH (Interpret Constituent Here) we were supposed to generate only the entry for the ICH node but we did not pay attention to that in our experiments reported here. It will certainly be an interesting item for future work.

In Table 2 the line having *all* in the verb column reports results when training examples of all target verbs where considered together in a single experiment, say *all-verb*. The training time becomes larger with larger training sets while the error and estimated error become similar (7.5 - 7.7%). However this illustrates a more general approach in which for all target verbs we generate a single decision tree (to determine if a certain head plays the role of subject for a verb found at distance *pos*) as opposed to having a single decision tree for each verb. The last line in the table shows results when the head feature is ignored. There is a small increase in the error rate (1%) but a simpler, less-lexicalized model is obtained.

Table 3 contains error rates for a trial in which traces in the corpus are resolved (the second subexperiment of the first major experiment).

The feature extraction method includes the new step of *Solve traces*:

procedure *ExtractFeatures(Sentence)*
- Generate a full syntactic parse for the Sentence
- Stem the words in the sentence
- *Solve traces*
- Identify the target *verb* or all *verbs*

for *verb in Sentence* **do**
- Extract the set of features for each node in the tree relative to the verb
- Classify each node as *norole* or as *logical subject*

end for

Double link traces, as the links between indexes 3-2-1 in the following example, are ignored: (SBARQ (WHNP-1 Who) (SQ was (NP-SBJ-2 *T*-1) (VP believed (S (NP-SBJ-3 *-2) (VP to (VP have (VP been (VP shot (NP*-3)))))))))). The number of training examples generated for each individual verbs is larger (on average there are 10% more examples) and the errors are mixed for different verbs: it drops for *believe*, it increases for *ask*, it remains the same for *begin* but

Verb	Training Size	Errors(%)	Estimate Errors(%)	Errors before Pruning
ask	6968	6.8	8.1	7.7
begin	6691	7.7	8.7	8.1
believe	8766	9.0	9.9	9.8
cause	5990	8.8	10.0	9.7
expect	27792	7.8	8.2	7.9
find	6286	7.2	8.5	7.9
give	11764	6.9	7.7	8.1
help	13359	7.7	8.4	9.1
like	18295	7.4	8.0	8.6
move	10885	7.5	8.2	8.3
produce	9579	7.4	8.2	8.3
provide	10000	7.0	7.7	8.0
seem	7536	7.3	8.4	7.8
swing	1248	9.1	9.8	7.1
all	145159	7.5	7.7	7.8
all+no-head	145159	8.5	8.8	8.9

Table 2: Errors reported by the induced decision trees with 10-fold cross validation

Verb	Training Size	Training Size Increment	Errors(%)	Estimate Errors(%)	Errors before Pruning
ask	7886	13%	6.9	8.2	7.6
begin	7184	7%	7.7	8.7	8.1
believe	9624	10%	8.5	9.6	9.3
cause	6563	10%	9.2	9.5	9.7
expect	29613	7%	7.8	8.3	8.2
find	6902	10%	7.0	8.4	7.8
give	12766	9%	6.6	7.4	7.8
help	14531	8.7%	7.4	8.3	9.3
like	19811	8.2%	7.4	8.1	8.0
move	11629	7%	7.1	8.0	8.6
produce	10121	6%	6.9	7.8	8.3
provide	10754	8%	6.8	7.6	7.9
seem	8071	7%	7.2	8.4	7.7
swing	1298	4%	8.6	9.2	7.4
all	160056	10%	7.3	7.5	7.6
all+no-head	160056	10%	8.0	8.3	8.4

Table 3: Errors when traces are solved.

overall there is a *0.2%* decrement (see line *all*). The last line in the table shows results when the lexical information about the word that plays the role is ignored. As we notice there is only a marginal increase in the error rate. It seems like in the absence of deeper semantic information about the word itself (for example the semantic class for objects, usually used to specify selectional restrictions) there is not much impact on the original model by the lexical feature.

Experiment 2

In this second major experiment training examples from all chosen verbs are put together and each example has the verb as a feature. We would like to explore the impact on performance when the verb becomes a feature. As shown in Table 4, which includes results for the first subexperiment, there is no change in the error rates when the verb is considered for the all-verb experiment. We plan to further study the impact of the verb itself on our model by using different sets of verbs. There is a small increase in the error rate when the head of the candidate word is eliminated (see line *all+verb+no-head*). The same trends are present when the training set is expanded with examples containing solved traces (second subexperiment) as illustrated in Table 5.

Class-based Generalization

An improvement in performance was obtained by generalizing examples with named entities as the head feature. For instance, the *IMA* and the *Department of Trade and Industry* were changed to group, *Angelo* and *Keneth Roman* to person and *Arizona* to location. A NE component is used to recognize person names (PER), organizations/groups (ORG) and places/locations (LOC). We applied this techniques on the training data for the verb *ask* and obtained an improvement in performance of about 2.19%. The person category could be also mapped into a personal pronoun. This would help especially when new names (unseen data) is encountered. For common nouns a similar solution could be implemented using a general English taxonomy such as WordNet similar to what Li and Abe (Li & Abe 1998) do for generalizing case frame. Each common noun would be replaced by carefully chosen, more general concepts from the hierarchy. An important issue for such a solution is the set of classes/general concepts to be used.

The class-based generalization method leads to a smaller training set and, consequently, to a smaller training time and smaller decision tree at the expense of more preprocessing when generating the training and test sets.

Comparing Performance with Other Systems

Although there are no systems of which we are aware of that address the task of logic roles identification we mentioned systems that address a similar task of shallow semantic parsing. Table 6 offers a comparative view at the precision of our model (the first model with traces resolved for all-verb experiment) and the ones in (Gildea & Hockenmaier 2003)(G&P in the table) (Pradhan *et al.* 2003)(SVM in the table) for which we picked the best overall precision on arguments 0-5 for the *gold* experiments which use gold standard parses.

System	Precision
Logic	92.5
SVM	89
G&P	85

Table 6: Comparing logic role identification with shallow semantic parsing

Our results are considerably better due probably to the relatively simpler task of identifying logical roles as compared to the shallow semantic parsing which includes two subtasks argument classification and argument identification. The performance of the best mentioned systems in each individual subtask is in the upper 80s or lower 90s, thus comparable to the performance on the logic role identification task.

Conclusions

The results presented in this paper form a upper bound of the performance of our models since the tagging and parsing of the sentences from which we derive our data are accurate (tag or parse errors are present, though at a very low rate, in Treebank).

On the other hand, the results represent a lower bound since we worked with highly ambiguous verbs. We expect our results to improve as we work on a set of verbs that better resembles the natural distribution of argument ambiguity of English verbs.

We plan to introduce tagging and parsing errors and perform experiments on the performance of the method in the presence of such noise. Using noisy state-of-the-art parsers for that purpose will not help as parsers do not provide information regarding syntactic functional arguments at logic level.

Moreover, since the features of the underlying proposed model were automatically extracted from a large, manually annotated corpus, we showed that the construction of such corpora is extremely beneficial for the progress of automated human language technologies.

References

Baker, C. F.; Fillmore, C. J.; and Lowe, J. B. 1998. The Berkeley FrameNet project. In Boitet, C., and Whitelock, P., eds., *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics*, 86–90. San Francisco, California: Morgan Kaufmann Publishers.

Chen, J., and Rambow, O. 2003. Use of deep linguistic features for the recognition and labeling of semantic arguments. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Verb	Training Size	Errors(%)	Estimate Errors(%)	Errors before Pruning
all+verb	145159	7.5	7.7	7.8
all+verb+no-head	145159	8.3	8.6	8.7

Table 4: Error rates with the verb as a feature and no traces

Verb	Training Size	Training Size Increase	Errors(%)	Estimate Errors(%)	Errors before Pruning
all+verb	160056	10%	7.3	7.5	7.7
all+verb+no-head	160056	10%	7.8	8.2	8.5

Table 5: Error rates with the verb as a feature and no traces

Collins, M. J. 1996. A new statistical parser based on bigram lexical dependencies. In Joshi, A., and Palmer, M., eds., *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, 184–191. San Francisco: Morgan Kaufmann Publishers.

Daelemans, W.; Zavrel, J.; van der Sloot, K.; and van den Bosch, A. "timbl: Tilburg memory-based learner - version 5.0 reference guide". ILK Technical Report 03-10.

Davidson, D. 1967. The logical form of action sentences. In Rescher, N., ed., *The Logic of Decision and Action*. University of Pittsburgh Press. 81–95.

Gildea, D., and Hockenmaier, J. 2003. Identifying semantic roles using combinatory categorial grammar. In *Proceedings of 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Hobbs, J. 1983. Ontological promiscuity. In *Proceedings 23rd Annual Meeting of the Association for Computational Linguistics*, 57–63.

Kingsbury, P.; Palmer, M.; and Marcus, M. 2002. Adding semantic annotation to the penn treebank. In *Proceedings of the Human Language Technology Conference*.

Korhonen, A.; Gorrell, G.; and McCarthy, D. 2000. Statistical filtering and subcategorization frame acquisition.

Lapata, M. 1999. Acquiring lexical generalizations from corpora: A case study for diathesis alternations. In *Proceedings of the 37th Meeting of the North American Chapter of the Association*, 397–404.

Li, H., and Abe, N. 1998. Generalizing case frames using a thesaurus and the MDL principle. *Computational Linguistics* 24(2):217–244.

Marcus, M.; Santorini, B.; and Marcinkiewicz. 1993. Building a large annotated coprus of english: the penn treebank. *Computational Linguistic* 19(2):313–330.

Mihalcea, R. 2002. Diacritics restoration: Learning from letters versus learning from words. In *CICLing*, 339–348.

Moldovan, D. I., and Rus, V. 2001. Logic Form transformation of wordNet and its Applicability to question answering. In *Proceedings of ACL 2001*. Toulouse, France: Association for Computational Linguistics.

Pradhan, S.; Hacioglu, K.; Ward, W.; Martin, J.; and Jurafsky, D. 2003. Semantic role parsing: Adding semantic structure to unstructured text. In *Proceedings of the International Conference on Data Mining (ICDM-2003)*.

Rus, V. 2002. *Logic Form for WordNet Glosses and Applications*. Phd thesis, Southern Methodist University.

Rus, V. 2003. A Semantically Enhanced Logic Form: Introducing Adjuncts. Technical report, Indiana University. Unpublished Manuscript.

Stetina, J.; Kurohashi, S.; and Nagao, M. 1998. General word sense disambiguation method based on A full sentential context. In *Use of WordNet in Natural Language Processing Systems: Proceedings of the Workshop*. Somerset, New Jersey: Association for Computational Linguistics. 1–8.

Stevenson, S., and Merlo, P. 1999. Automatic verb classification using distributions of grammatical features.

Ted, B., and Carroll, J. 1997. Automatic extraction of subcategorization from corpora.

Default Reasoning Using Monotonic Logic: Nutter's modest proposal revisited, revised and implemented

Karen Ehrlich
Mathematics and Computer Science Department
SUNY College at Fredonia
Fredonia, NY 14063
ehrlich@cs.fredonia.edu

Abstract

It is sometimes necessary to reason non-monotonically, to withdraw previously held beliefs. Default rules and defeasible reasoning have been frequently used to handle such situations. Some years ago, J. Terry Nutter proposed a form of defeasible reasoning involving additional truth-values that would avoid non-monotonicity in many situations. We adopt and implement much of Nutter's underlying concept, but use a structure of case frames in our knowledge representation to avoid the need for more than the standard truth values. Our implementation captures most of the advantages of the original proposal and in some cases allows more flexibility.

Introduction

As part of an ongoing project in constructing an artificially intelligent cognitive agent known as Cassie, that (among other skills) reads and comprehends narrative text [Shapiro 1989], we are developing a system that builds definitions for unknown words from their linguistic context combined with background knowledge [Ehrlich 1995, Ehrlich & Rapaport 1997, Rapaport & Ehrlich 2000, Rapaport & Kibby 2002].

While reading, one may encounter information that appears to contradict one's previous understanding. In some cases, it is clear that the original understanding was in error. In those cases, we must withdraw previous belief; we must have recourse to non-monotonic reasoning. However, we can often avoid the need for non-monotonicity by expressing our beliefs in less than absolute terms. Defeasible rules are one way to do this: a rule may be blocked in some cases, but is assumed to apply if not specifically blocked.

In a classic example, we have the rule that, by default, birds fly. On reading that Tweetie is a bird, and knowing no contradictory information, the rule would, by default, lead to the conclusion that Tweetie flies. If Tweetie is a healthy young canary, the rule *should* apply, and all is well. If one knows that penguins are flightless birds, and reads of Opus, a penguin, one would not conclude that Opus flies. One would infer that Opus is a bird, but the defeasible rule would not be applied, because we have specific information blocking its application in this case. We know that the rule does not apply to Opus, and all is again well. So long as one has all relevant information before attempting to deduce whether a particular bird flies, there should be no trouble. However, if one must make a decision in advance of potentially relevant data, the fact that a defeasible rule was used will have no effect on the conclusion drawn, or the need to revise it.

For example, if we read of Elmer the emu, and all we know of emus is that they are birds, we will conclude that Elmer flies. If we later learn that emus are no more flight-capable than penguins, we'll need to retract the conclusion, just as if it were based on an indefeasible rule that all birds fly.

Rather than use defeasible rules that rely on knowing in advance whether they are applicable in certain cases, we employ a method, based largely on theoretical work by Nutter [1983a,b], in which some rules have consequents marked as presumably true. Frequently, this avoids the need for non-monotonicity. In our example from above, we would have the rule that, if something is a bird, then presumably it flies. Opus, the penguin, and Elmer the emu, being both birds, would fit the antecedent, and we would conclude that presumably each flies. Learning (for example) that emus do not fly does not, then, produce a contradiction, but rather the concatenation that Elmer could be presumed to fly though in fact he does not.

Underlying Software: SNePS

Our system is built on SNePS, a semantic-network-based knowledge representation and reasoning system developed by Stuart C. Shapiro and the SNePS Research Group [1996, 1999], which has facilities both for parsing and generating English and for belief revision [Shapiro 1979; Shapiro 1982; Shapiro & Rapaport 1987; Martins & Shapiro 1988; Shapiro & Rapaport 1992; Cravo & Martins 1993; Shapiro & the SNePS Implementation Group 1999].

SNePS has been and is being used for several research projects in natural language understanding and generation including Rapaport [1986], Almeida [1987], Peters & Shapiro [1987a,b], Peters, Shapiro & Rapaport [1988], Rapaport [1988a,b], Peters & Rapaport [1990], Wyatt [1990], Shapiro & Rapaport [1991], Yuhan [1991], Ali [1993], Ali & Shapiro [1993], Kumar [1993], Shapiro [1993], Chalupsky & Shapiro [1994], Wiebe [1994], Almeida [1995],Rapaport & Shapiro [1995], Yuhan & Shapiro [1995], Chalupsky & Shapiro[1996], Wiebe &

Rapaport [1996], Johnson & Shapiro [1999a,b], Shapiro, Ismail & Santore [2000], Shapiro & Johnson [2000], and Shapiro & Ismail 2003.

The Nature Of SNePS Networks. Each node in a SNePS network is a term in a language of thought that represents a mental object (possibly built of other mental objects). Labeled arcs link the nodes to one another.

> [A]ll information, including propositions, "facts", etc., is represented by nodes [and] propositions about propositions can be represented with no limit.
>
> Arcs merely form the underlying syntactic structure of SNePS. [O]ne cannot add an arc between two existing nodes. That would be tantamount to telling SNePS a proposition that is not represented by a node. [Shapiro & Rapaport 1987: 266-267]

Anything about which one can think is represented as a node. Nodes represent the concepts embodied by the nouns, pronouns, verbs, adjectives, and adverbs of natural language. Base nodes (those that have no arcs emerging from them) typically represent a cognitive agent's concepts of particular objects or entities, or they represent individual words or sensory inputs. Molecular nodes represent structured concepts, including those that might be expressed in natural language phrases and sentences [Maida & Shapiro 1982, Shapiro & Rapaport 1987]. A special variety of molecular node is the pattern node, which has arcs to variable nodes. These pattern nodes and variable nodes are used in the construction of rules, as we shall see below.

Certain propositions may be asserted in SNePS, but it is possible to represent propositions without asserting them. Thus, a cognitive agent whose mental state is represented as a SNePS network can consider a proposition without necessarily believing it. Typically, a proposition embedded within another proposition is not asserted (although it can be; see Wiebe & Rapaport [1986], Rapaport [1986], Rapaport, Shapiro, & Wiebe [1997], Rapaport [1998]).

Rule-Based Inference In SNePS. SNePS allows the creation of universally quantified rules such as: (For all v_1, v_2, v_3) [(Before (v_1, v_2) & Before (v_2, v_3)) => Before (v_1, v_3)]. In SNePS list-representation such a rule could be represented as:

```
(M1! (FORALL V1 V2 V3)
     (&ANT (P1 (BEFORE V1 AFTER V2))
           (P2 (BEFORE V2 AFTER V3)))
     (CQ (P3 (BEFORE V1 AFTER V3))))
```

In this representation, a node is described as a list, the first element of which is its name, and subsequent elements of which describe the arcs leading from it. Each of these descriptions is a list whose first element is the arc-name and whose second element is a list of all the nodes at the

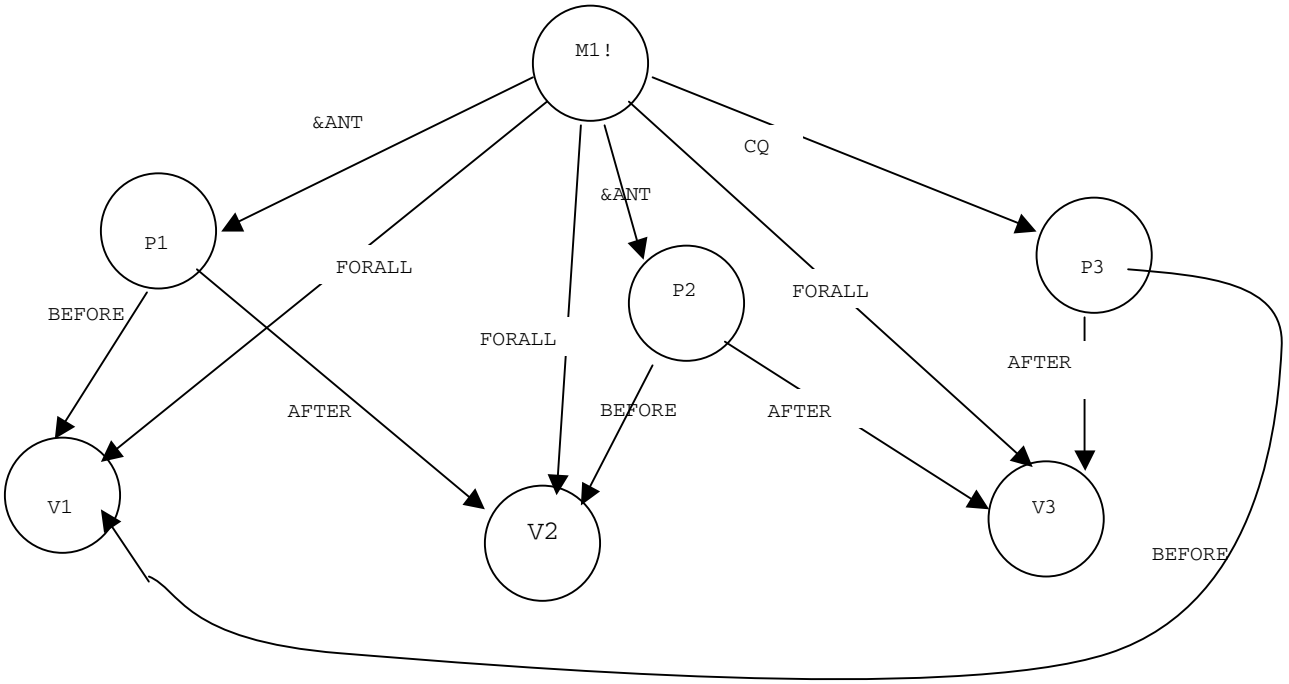


fig.1 Rule (M1) describing the transitivity of "before" and "after"

heads of those arcs. A typical SNePS rule node has one or more of each of the following arcs: FORALL pointing to the variable node(s) of the rule, ANT or &ANT pointing to the antecedent pattern(s), and CQ pointing to the consequent pattern(s) of the rule. If the antecedent is a single proposition, or if there is a choice among possible antecedents, we use the ANT arc. If the rule requires that multiple antecedents be matched, we use the &ANT arc. So, in rule M1!, both patterns P1 and P2, must be satisfied by matching against other propositions (by substituting for the variables V1, V2, and V3) in order to allow the rule to fire and create a new asserted proposition that matches the consequent pattern P3. A graphical representation of M1! is depicted in *figure 1*.

Assertion is indicated by an exclamation mark following the name of the node (e.g., M1!) If we think of the network as Cassie's mind, an asserted proposition is one that believed to be true and an unasserted node represents a concept that has been entertained but not necessarily believed. Coming to believe a previously considered idea, then, involves the assertion of a pre-existing propositional node. Ceasing to hold a previous belief entails "un-asserting" an asserted node. The antecedents of the above rule will be satisfied if there exist a pair of asserted nodes with "before" and "after" arcs, and the "before" arc from one points to the same node as does the "after" arc from the other, for example:

(M14! (BEFORE B23) (AFTER B24))
(M19! (BEFORE B24) (AFTER B28))

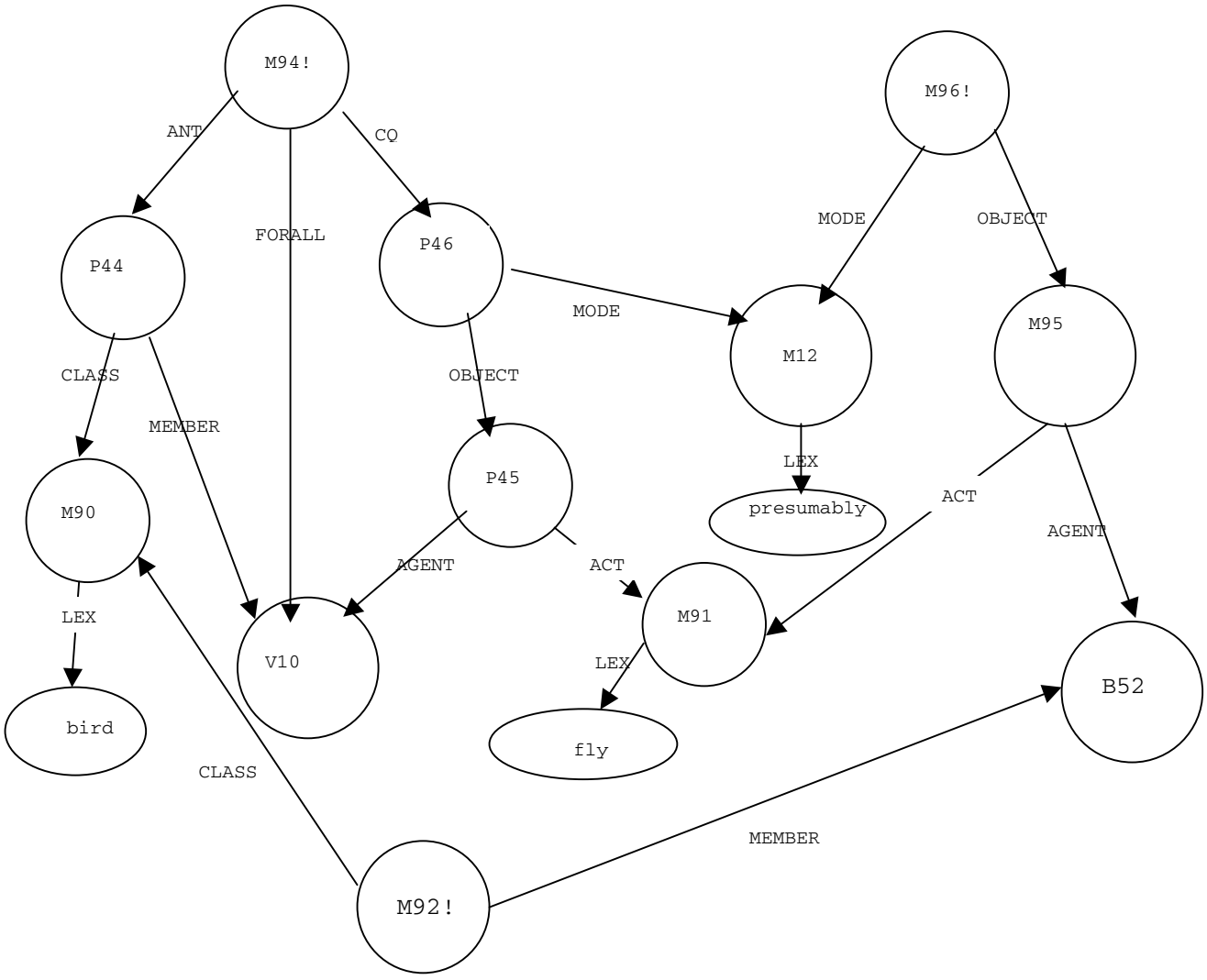


fig. 2 Rule(M94)that if V10 is a bird, then presumably it flies and derived assertion (M96) that a particular bird, B52, flies.

Here, the asserted propositions M14! and M19! match the pattern nodes P1 and P2, so that, when the rule M1! fires, it will build a new asserted proposition (M20!) to the effect that the time represented by B23 is before the time represented by B28.

```
(M20! (BEFORE B23) (AFTER B28))
```

Case Frames for "Modal Logic"

In our implementation, rules with consequents that are presumably true differ from ordinary rules only in that the consequent takes the form of a (MODE OBJECT) case frame. For example, node M94! (see *figure 2*) is the rule that, for all V10, if V10 is a bird then presumably V10 flies.

```
(M94! (FORALL V10)
      (ANT (P44 (CLASS (M90 (LEX bird)))
                (MEMBER V10)))
      (CQ  (P46 (MODE (M12 (LEX presumably)))
                (OBJECT (P45 (ACT (M91 (LEX fly)))
                             (AGENT V10))))))

(M96! (MODE (M12 (LEX presumably)))
      (OBJECT (M95 (ACT (M91 (LEX fly)))
                   (AGENT B52))))
```

B52 is Cassie's concept of a particular bird, and node M96! is the belief (created by the firing of M94!) that presumably B52 flies. Notice that node M95, the proposition that B52 flies, is not asserted. At this point Cassie does not believe either that B52 flies or that it doesn't. New information could cause either M95 or its negation to be asserted, *without* contradicting M96!

Note that, as used here, "presumably" is just a word representing a concept in the Cassie's knowledge base. It is not a logical operator. Nutter originally proposed the use of "presumably" as an operator, and suggested expanding the number of truth values to four: "true", "presumably true", "false", and "presumably false". This is different from the five truth values ("monotonically true", "default true", "unknown", "default false", "monotonically false") employed in the Cyc KB. In Cyc, a "monotonically true" assertion is true in all circumstances, whereas a "default true" assertion is an assertion (often a rule) to which there may be exceptions [Cycorp 2002]. Nutter's theory allows a proposition to be both "presumably true" and "false" without there being a contradiction. The gloss (as in the case of Elmer or Opus flying) would be something like: "there is reason to believe that *P*, but not *P*". Our system uses only the standard SNePS truth values: assertions are believed to be true; assertions of negation mean that the negated proposition is false; unasserted propositions have no real truth value as far as the system is concerned. It might be interesting to one day implement a "presumably" operator, expand of the number of possible truth-values, and compare such an approach with our case-frame approach. Currently, however, we believe that the case-frame approach captures the important elements of Nutter's proposal without the need for additional truth-values.

Using a case-frame approach also allows us a greater degree of flexibility in the construction of rules than would be the case if we used a logical operator and the revised truth-values. Nutter suggested that, if a presumably true assertion matched the antecedent of a rule, the assertion built as a result of the rule's firing would also be presumably true. When this is what we want, we can write our rule this way, with the (MODE presumably OBJECT <pattern>) case-frame appearing in both the antecedent and the consequent: If presumably *P*, then presumably *Q*. But our approach also allows us to have a rule with a "presumable" antecedent and a consequent that is baldly asserted, e.g.: "If, presumably, *x* purrs, then *x* is feline." We would be unlikely to presume *x* purred, *unless* it were feline.

So, our case-frame approach allows us to write the same sort of default rules (If *P*, then presumably *Q*) as would the operator and truth-values approach, and it allows us the flexibility to write rules of the form "If presumably *P* then *Q*". The trade-off we make is that, in some cases we may need two rules, where using the operator and expanded set of truth-values would require only one. Under Nutter's proposed implementation, the antecedent of a rule such as "If *x* purrs, then *x* is feline" could match the belief that "Presumably, Tom purrs" and lead to the conclusion that "Presumably, Tom is feline". Our approach does not allow that match. If we wanted a rule that would fire when it was found that Tom purrs *or* that presumably he purrs, we would need to write that rule with a disjunction of two antecedents: "If *x* purrs, or if presumably *x* purrs, then *x* is feline". If we wanted the unqualified assertion to lead to an unqualified conclusion, and a presumably true assertion to lead to a conclusion marked as presumably true, then we would need two rules.

It seems probable that some such implicit qualification is part of most humans' understanding, though it is not generally expressed. People are apt to state their rules of thumb without qualification unless the exceptions are explicitly drawn to their attention, but they maintain those rules even when well aware that there are exceptions. When exceptions are pointed out, a typical reaction is something on the order of "well, of course, but in general…". There is not usually any sense that one's belief was in error, or in need of modification.

As mentioned at the beginning of this article, the case frame implementation of presumably true propositions is being currently being used in a project on contextual vocabulary acquisition. Much background knowledge is

encapsulated in rules that employ this case frame. When asked for a definition of a word, Cassie selects relevant aspects of her knowledge and experience with that word to report. Just which aspects are chosen depends upon the type and quantity of her exposure to the word, and the contexts in which it occurs, as well as what background knowledge she has. If she has information about the presumable actions, functions, structure, etc., of objects named by a certain noun, she will report those in preference to the observed actions, functions, structure of individual instances of that noun, but lacking such information, she relies on what she knows of individuals. On the other hand, if she has hard and fast rules (really *definitional* rules) such information takes precedence over the "presumably true" behaviors, etc. Similarly in defining verbs, Cassie employs a hierarchy in analyzing the effects of actions. Those effects known to follow from certain actions are reported preferentially, but the known probable effects of actions are usually reported as well, if available. If no such knowledge is to be had, Cassie may hypothesize as to the possible effects of the action described by certain verb, based on observation of changes described in the narrative in which she encounters the word knowledge [Ehrlich 1995, Ehrlich & Rapaport 1997, Rapaport & Ehrlich 2000, Rapaport & Kibby 2002].

Conclusion

While neither traditional defeasible rules nor rules whose consequents are marked as presumably true can entirely relieve us of the need for non-monotonic reasoning, both (and especially the latter) can reduce the frequency of such a need. Either is probably a more accurate representation of the knowledge with which we reason (although they are usually less accurate representations of how we report our knowledge) than classical, indefeasible, non-modal logic is. We have used case-frames in our knowledge representation and reasoning system to implement a variant of Nutter's proposed modal logic, marking certain propositions as "presumably true", and allowing that a proposition may be both "presumably true" and "false" at the same time without contradiction. This approach allows us to reason monotonically wherever possible (reserving the need for non-monotonic reasoning to those situations in which previous beliefs are actually shown to be erroneous), but does not require the implementation of more than the classical truth values.

References

Ali, S. S., (1993). A Structured Representation for Noun Phrases and Anaphora *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society* (Hillsdale, NJ: Lawrence Erlbaum Associates): 197-202.

Ali, S. S., & Shapiro, S. C. (1993). Natural Language Processing using a Propositional Semantic Network with Structured Variables, *Minds and Machines,* 3(4): 421-451

Almeida, M. J., (1987). Reasoning about the Temporal Structure of Narratives, *Technical Report 87-10* (Buffalo, NY: SUNY Buffalo Department of Computer Science).

Almeida, M. J., (1995). Time in Narratives" in Duchan, J., Bruder, G.& Hewitt, L. (eds.*), Deixis in Narrative: A Cognitive Science Perspective* (Hillsdale, NJ: Lawrence Erlbaum Associates).

Chalupsky, H. & Shapiro, S. C. (1994). SL: A subjective, intensional logic of belief *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society,* (Hillsdale, NJ: Lawrence Erlbaum Associates) 165-170.

Chalupsky, H. & Shapiro, S. C. (1996). Reasoning about incomplete agents, *Proceedings of the Fifth International Conference on User Modeling (UM-96),* (User Modeling, Inc.) 169-177.

Cravo, M., & Martins, J. P. (1993). SNePSwD: A Newcomer to the SNePS Family, *Journal of Theoretical and Experimental Artificial Intelligence 5:* 135–148.

Cycorp. (2002). http://www.cyc.com/cycdoc/ref/glossary.html

Ehrlich, K., (1995). Automatic Vocabulary Expansion through Narrative Context, *Technical Report 95-09* (Buffalo: SUNY Buffalo Department of Computer Science).

Ehrlich, K., & Rapaport, W. J. (1997). A Computational Theory of Vocabulary Expansion, *Proceedings of the 19th Annual Conference of the Cognitive Science Society (Stanford University).* (Mahwah, NJ: Lawrence Erlbaum Associates): 205–210.

Johnson, F.L.& Shapiro S. C. (1999a). Says Who?--Incorporating Source Credibility Issues into Belief Revision, *Technical Report 1999-08*, (Buffalo, NY: SUNY Buffalo Department of Computer Science and Engineering).

Johnson, F.L.& Shapiro S. C. (1999b). Finding and Resolving Contradictions in a Battle Scenario, *Technical Report 99-09*, (Buffalo, NY: SUNY Buffalo Department of Computer Science and Engineering).

Kumar, D., (1993). A unified model of acting and inference. in Jay F. Nunamaker, Jr., Jay F. and Sprague, Jr., Ralph H. (eds.) *Proceedings of the Twenty-Sixth Hawaii International Conference on System Sciences Volume III*, (Los Alamitos, CA: IEEE Computer Society Press) 483-492.

Maida, A. S., & Shapiro, S. C., (1982). Intensional concepts in propositional semantic networks *Cognitive Science*, 6(4):291–330. Reprinted in Brachman, R. J., & Levesque, H.J. (eds.) *Readings in Knowledge Representation*,(Los Altos, CA: Morgan Kaufmann, 1985)

Martins, J. P., & Shapiro, S. C., (1988). A Model for Belief Revision, *Artificial Intelligence, 35*: 25–79.

Nutter, J. T., (1983a). Default Reasoning Using Monotonic Logic: A Modest Proposal, *Proceedings of the National Conference on Artificial Intelligence* (*AAAI-83, Washington DC*), (Los Altos, CA: Morgan Kaufmann): 297–300

Nutter, J. T., (1983b). Default Reasoning in A.I. Systems, *Technical Report 204* (Buffalo: SUNY Buffalo Dept. of Computer Science).

Peters, S. L., & Rapaport, W. J., (1990). Superordinate and Basic Level Categories in Discourse: Memory and Context, *Proceedings of the 12th Annual Conference of the Cognitive Science Society* (*Cambridge, MA*) (Hillsdale, NJ: Lawrence Erlbaum Associates): 157-165.

Peters, S. L., & Shapiro, S.C., (1987a). A Representation for Natural Category Systems I, *Proceedings of the 9th Annual Conference of the Cognitive Science Society* (*Seattle*) (Hillsdale, NJ: Lawrence Erlbaum Associates): 379-390.

Peters, S. L, & Shapiro S.C., (1987b). A Representation for Natural Category Systems II, *Proceedings of the 10th International Joint Conference on Artificial Intelligence* (*IJCAI-87*, *Milan*) (Los Altos, CA: Morgan Kaufmann): 140-146.

Peters, S. L., Shapiro, S. C., & Rapaport, W. J., (1988). Flexible Natural Language Processing and Roschian Category Theory, *Proceedings of the 10th Annual Conference of the Cognitive Science Society* (*Montreal*) (Hillsdale, NJ: Lawrence Erlbaum Associates): 125-131.

Rapaport, W. J., (1986). Logical Foundations for Belief Representation, *Cognitive Science, 0*: 371–422.

Rapaport, W. J. (1998a). How Minds Can Be Computational Systems, *Journal of Experimental and Theoretical Artificial Intelligence 10*: 403–419.

Rapaport, W. J., (1988b). Syntactic Semantics: Foundations of Computational Natural-Language Understanding, in Fetzer, James H. (ed.), *Aspects of Artificial Intelligence* (Dordrecht, Holland: Kluwer Academic Publishers).

Rapaport, W. J., & Ehrlich, K. (2000). A Computational Theory of Vocabulary Acquisition, in Iwanska, L. & Shapiro, S. C. (eds.), *Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language* (Menlo Park, CA/Cambridge, MA: AAAI Press/MIT Press).

Rapaport, W. J., & Kibby, M. W. (2002,). Contextual Vocabulary Acquisition: A Computational Theory and Educational Curriculum, *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2002; Orlando, FL)*

Rapaport, W. J., & Shapiro, S. C., (1995). "Cognition and Fiction" in Duchan, J. F., Bruder, G. A., & Hewitt, L. (eds.), *Deixis in Narrative* (Hillsdale, NJ: Lawrence Erlbaum Associates).

Rapaport, W. J., Shapiro, S. C., & Wiebe, J. M., (1997). Quasi-Indexicals and Knowledge Reports, *Cognitive Science 21*, 1:63-107.

Shapiro, S. C., (1979). The SNePS Semantic Network Processing System," in Findler, N. V. (ed.), *Associative Networks* (New York: Academic Press).

Shapiro, S. C., (1982). Generalized Augmented Transition Network Grammars for Generation from Semantic Networks, *American Journal of Computational Linguistics, 8*: 12-25.

Shapiro, S. C., (1989). The CASSIE Projects: An Approach to Natural Language Competence, in Martins, J. P., & Morgado, E. M., (eds.), *Proceedings of the 4th Portuguese Conference on Artificial Intelligence (EPIA-89*), Lecture Notes in Artificial Intelligence 390 (Berlin: Springer-Verlag): 362-380.

Shapiro, S. C., (1993). Knowledge Representation for Natural Language Processing, *Minds and Machines, 3*,(*4*): 377-380.

Shapiro, S. C., Ismail, H. O., & Santore, J. F., (2000). Our Dinner with Cassie, *Working Notes for the AAAI 2000 Spring Symposium on Natural Dialogues with Practical Robotic Devices,* (Menlo Park, CA:AAAI) 57-61.

Shapiro, S. C., Ismail, H. O., (2003). Anchoring in a Grounded Layered Architecture with Integrated Reasoning, *Robotics and Autonomous Systems 43*, 2-3:97-108.

Shapiro S. C. & Johnson, F. L. (2000) Automatic Belief Revision in SNePS. In C. Baral & M. Truszczynski, Eds., *Proceedings of the 8th International Workshop on Non-Monotonic Reasoning NMR2000,* 2000, unpaginated, 5 pages. Also Technical Report 2000-01, (Buffalo, NY: SUNY Buffalo Department of Computer Science and Engineering).

Shapiro, S. C., & Rapaport, W. J., (1987). SNePS Considered as a Fully Intensional Propositional Semantic Network, in Cercone, N., & McCalla, G., (eds.) *The Knowledge Frontier: Essays in the Representation of Knowledge* (New York: Springer-Verlag).

Shapiro, S. C., & Rapaport, W. J., (1991). Models and Minds: Knowledge Representation for Natural-Language Competence, in Cummins, R. & Pollock, J. (eds.) *Philosophy and AI: Essays at the Interface* (Cambridge, MA: MIT Press).

Shapiro, S. C., & Rapaport, W. J., (1992). The SNePS Family, *Computers and Mathematics with Applications, 23*: 243-275.

Shapiro, S. C., & the SNePS Implementation Group (1996). A Dictionary of SNePS Case-Frames, online: www.cse.buffalo.edu/sneps/Bibliography/bibliography.ht

ml#Manuals.

Shapiro, S. C., & the SNePS Implementation Group, (1999). SNePS-2.5 User's Manual, *SNeRG Technical Note* (Buffalo, NY: SUNY Buffalo Department of Computer Science and Engineering) online: www.cse.buffalo.edu/sneps/Bibliography/bibliography.html#Manuals.

Wiebe, J. M. (1994). Tracking Point of View in Narrative *Computational Linguistics, 20*: 233-287.

Wiebe, J. M., & Rapaport, W. J. (1986). Representing *De Re* and *De Dicto* Belief Reports in Discourse and Narrative, *Proceedings of the IEEE, 74*: 1405-1413.

Wiebe, J. M., & Rapaport, W. J. (1986). Representing *De Re* and *De Dicto* Belief Reports in Discourse and Narrative, *Proceedings of the IEEE, 74*: 1405-1413.

Wyatt, R., (1990). Kinds of Opacity and Their Representations in Kumar, D.(ed.) *Current Trends in SNePS-Semantic Network Processing System: Proceedings of the First Annual SNePS Workshop* (Buffalo, NY: Springer-Verlag): 123-144.

Yuhan, A. H., (1991). Dynamic Computation of Spatial Reference Frames in Narrative Understanding, *Technical Report 91-03* (Buffalo, NY: SUNY Buffalo Dept. of Computer Science).

Yuhan, A. H., & Shapiro, S. C., (1995). Computational Representation of Space in Duchan, J. F., Bruder, G. A., & Hewitt, L., (eds.) *Deixis in Narrative: A Cognitive Science Perspective* (Hillsdale, NJ: Lawrence Erlbaum Associates).

Using Selectional Restrictions to Parse and Interpret Student Answers in a Cardiovascular Tutoring System

Chung Hee Lee and Martha W. Evens

Department of Computer Science
Illinois Institute of Technology, Chicago, IL, 60616
Phone: 312-567-5153
Fax: 312-567-5067
leechun@iit.edu and evens@iit.edu

Abstract

Selectional restrictions are used in parsing both complete sentences and fragments typed in by students using an intelligent tutoring system. The selectional information helps in word sense disambiguation, parsing, anaphora resolution and the production of appropriate logic forms. We present two approaches based on the HPSG formalism. The first method employs HPSG's BACKGROUND feature and applies a constraint satisfaction mechanism after the parsing process is complete. The second method uses an ontology or subhierarchy of referential indices and applies the constraints that it provides during the parsing process. We conclude that the second approach is more efficient than the first, and that it is easier to expand to other domains.

Introduction

We are building an alternative parser for an intelligent tutoring system, CIRCSIM-Tutor (Michael et al., 2003), to handle long student answers to open questions. CIRCSIM-Tutor presents students with a short paragraph describing a perturbation to the blood pressure and asks them to predict changes in certain key cardiovascular parameters. It marks any errors in these predictions in red and then starts a remedial dialogue to help the students correct their errors. The students can type in any free text input they choose; most of the time they choose to type in brief fragments. Glass's (2001) information extraction parser does an excellent job of parsing the brief responses to most system questions, but it does not even try to parse the answers to a series of open questions that are generated when students are doing particularly well, in an attempt to enrich the experience for the better students. In this paper we describe the student input to CIRCSIM-Tutor during 66 sessions in November 2002. Then we discuss the new parser and describe how it takes advantage of information about selectional restrictions to disambiguate the input and produce appropriate logic forms.

Selectional restrictions can be used in various ways to encode knowledge about the syntax of the language used, about the semantics of the domain, and about the pragmatics of the system interaction. In this paper we will most often refer to semantic subsortal constraints defined by the ontology of the domain, but all kinds

of constraints can be handled in this way. We use our ontology to represent semantic selectional restrictions as referential indices.

Answers to Open Questions

Students respond to typical questions from the system with brief fragments. Some questions ask for the name of a parameter and the typical student types "co" or some other parameter name. Others ask for the direction of change and the student types "increase" or just "inc" or "+" or "up." The open questions were designed to push students into giving an explanation and they often enter much longer input. In answer to the question: "Why did you predict that IS would not change?" the system received complete sentences from twelve students, including:

"IS is a reflex response"
"there has not been a barorecptor reflex yet"
"Remembering the concept map, IS is reflected by the baroreceptor reflex."

Seven students answered with a *because* clause; among them:

"because it's a direct response and changing resistance wouldn't affect contractility of the heart just yet"
"Because only the baroreceptor firing rate directly affects IS."
"because baroreceptor reflex has not been activated yert"

Although these clauses are not complete sentences they are still much more complex than the fragments that the system ordinarily received from students; the last one reminds us that these fragments include just as much bad grammar and spelling as more typical inputs. Most of the rest of the answers to this question were noun phrases or a negative plus a noun phrase, which the information extraction parser may be able to handle:

"part of baroreceptor reflex"
"neural"
"not a direct relationship"
"no neural response"

As you might expect, the form of the question affects the form of the answer. Most of the answers to the question "What does the baroreceptor reflex do?" are verb phrases – but they are often more complex than anything the system ordinarily sees from students.

"regulates map"
"increase tpr, hr, is, map"
"attempt to keep MAP constant"
"indicate when pressures ncrease/decr"
"tries to change the direction of the direct response, tries to get variables back to normal"
"moderates blood pressure"

This variety of sophisticated inputs seems to require another approach to parsing.

Selectional Restrictions and the Domain Ontology

We can define selectional restrictions as semantic sortal constraints imposed on the participants in the action or as syntactic constraints imposed by the grammatical functions of the language. With these restrictions we can account for the felicity of (1) and (4), and the infelicity of (2) and (3).

(1) Cardiac output (co) increases central venous response (cvp).
(2) *Cardiac output (co) increases oil prices.

(3) *Cardiac output(co) increases direct response (dr).
(4) Cardiac output(co) increases in direct response (dr).

To account for sentence (1) we can make use of a constraint that the transitive sense of the verb "increase" refers to an event in which one cardiovascular variable causes another to change. The unacceptability of sentence (2) derives from a violation of this restriction. "Oil prices" are not known in our system; in a bigger system the ontology would tell us that they are economic variables, not cardiovascular ones. Turning to example (3), we note that "direct response (dr)" refers to a time period or phase of the baroreceptor reflex and not to a cardiovascular variable. To interpret sentence (4), we need to consider another (intransitive) sense of "increase" in which the cardiovascular variable referred to by the subject rises. We accept sentence (4) and interpret it correctly, because it obeys the restrictions on this sense of "increase". Selectional restrictions can generally be used in this way to capture facts about the world or about specific domains. Alshawi (1992) argues that they have many uses in practical applications, in spite of the limitations noted by Allen (1995) in interpreting metaphorical uses of language as in sentence (5) and determining the truth value of negated sentences like (6).

(5) My car drinks gasoline.
(6) Total peripheral resistance cannot increase oil prices.

Give the value of selectional restrictions in practical applications, we decided to explore how they can be utilized in a parser for an intelligent tutoring system based on HPSG or Head-Driven Phrase Structure Grammar (Pollard and Sag, 1994) using an ontology already developed for this domain (Lee et al., 2002) and designed to work with a unification-based parser.

Background Representation

The first approach to combining selectional restrictions with HPSG that we tested makes use of the CONTEXT|BACKGROUND (CX|BG) features that Pollard and Sag (1994) define for "felicity conditions on the utterance context", "presuppositions or conventional implicatures", and also "appropriateness conditions". They invented a new concept that they call a "qfpsoa" for quantifier free parameterized state of affairs to capture semantic roles in CX|BG. For example, the lexical sign of "increase" accommodates a "cardiovascular output (co)" qfpsoa which functions as a type in the cardiovascular ontology and as a subtype of the "cardiovascular variable (c-v)" qfpsoa, as shown below in Figure 1.

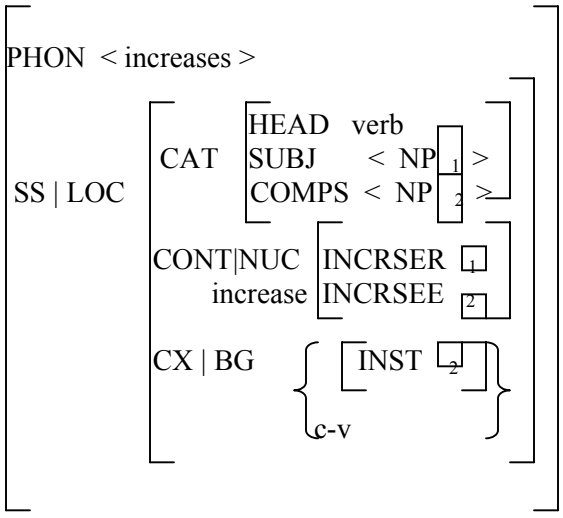


Fig. 1. Lexical Sign for the Transitive Sense of the Verb "increase"

The lexical sign for "cardiac output (co)" is represented as a nominal object that belongs to the class labeled *phase* in the ontology of the cardiovascular domain. The diagram for this design is shown in Figure 2 below.

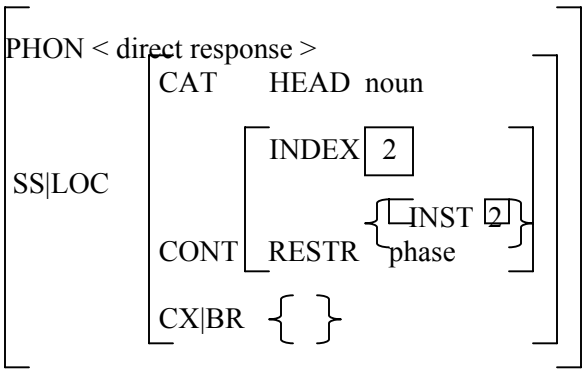


Fig. 2. Lexical Sign for "direct response"

The lexical sign for "cardiac output (co)" is represented as a nominal object that belongs to the class "cardiovascular variable (c-v)" as shown in Fig. 3.

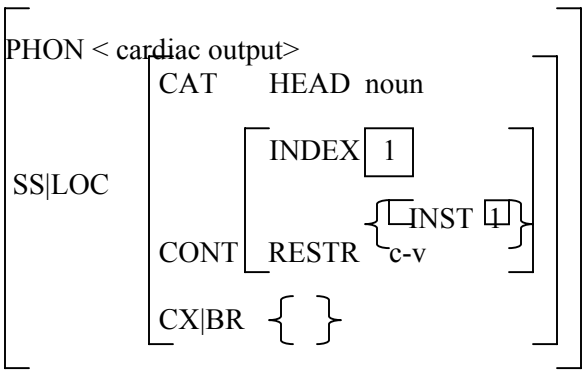


Fig. 3. Lexical Sign for "cardiac output"

To propagate features in a daughter node to its mother node, we can utilize the principle of contextual consistency stated by Pollard and Sag. This principle states that the set of CX| BG values of a mother node is the union of the CX| BG values of its daughter nodes. The subcategorization principle checks the subcategorization requirements of the lexical head (here the complement daughter must be "cardiovascular variable (c-v))". Using this principle the system checks the validity of the complements of the "increase" verb. The head feature principle guarantees that the features of the head are projection of the features of daughters. The system uses those principles to obtain the structure in Fig. 4 from sentence (1).

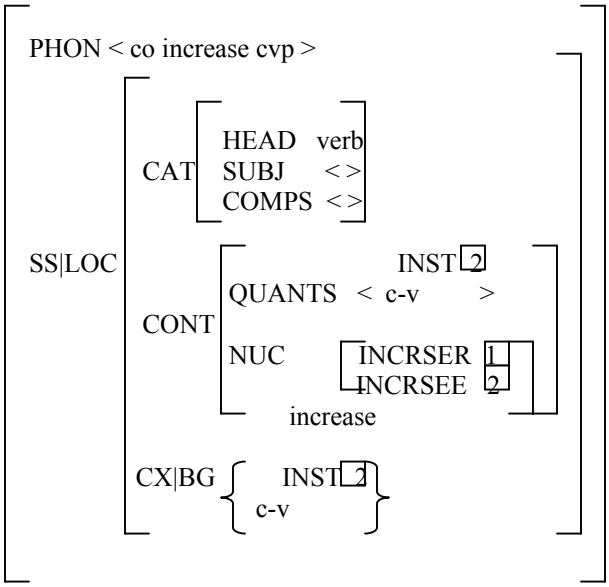


Fig. 4. Result of Parsing Sentence (1)

Because of the incompatibility between the type or class of "direct response (dr)" in CX|BG with its corresponding type in CONT|QUANTS, we cannot proceed any further with sentence (3) once we get the diagram in Fig. 5, since the transitive verb "increase" requires the type or class of "cardiovascular variable (c-v)), while the complement in the sentence is of type "phase". Neither "cardiovascular variable (c-v)" nor

"phase" subsumes the other in the sort hierarchy shown in Figure 6.

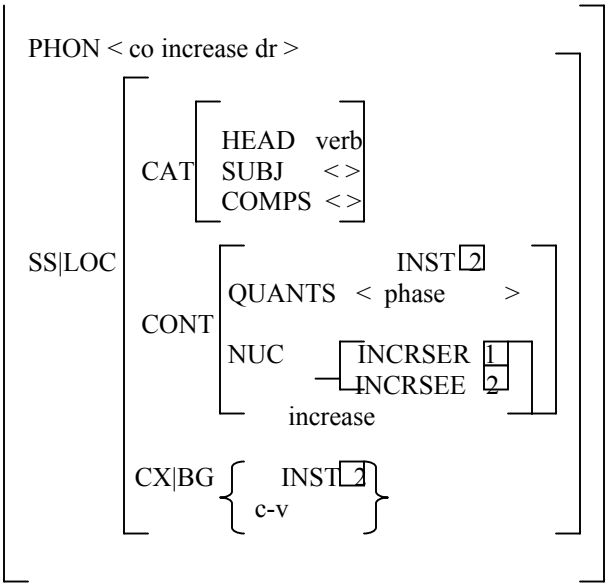


Fig. 5. Result of Parsing Sentence (3)

To obtain correct parses and reject ill-formed sentences, the system has to know where "cardiovascular variable (c-v)" fits in the ontology and also where "phase" fits. In order to account for these requirements, we employ selectional restrictions that represent this knowledge in the form of a hierarchy of domain related entities. We use the ontology that was proposed in (Lee et al., 2002) and analyze the patterns of fragmented answers with the tutor's questions. To interpret the fragmented question coherently with the question, we can adopt use the same hierarchy for semantic restrictions. Current CX|BG values of the verb are replaced to represent anticipated semantic restrictions and CX| BG values of the verb are moved to CONT|INDEX features. The most important advantage is interleaving the processing of the syntactic component with the semantic component; in addition the hierarchy helps us in the selection of anticipated categories. We now discuss an alternative approach that allows the ontology of world entities to be represented using the existing HPSG framework.

Index Mechanism

The current version of HPSG has a hierarchy of feature structure sorts, but it mainly classifies syntactic phenomena. As proposed by Ginzburg and Sag (2000), the existing HPSG hierarchy can be augmented to represent a new subtree of entities from the domain. This can be achieved by partitioning the reference (ref) of HPSG sorts into subsorts that correspond to our cardiovascular domain. In order to represent the semantic hierarchy in Figure 6, we can encode this information with reference (ref) type. In other words, referential indices are partitioned into sorts, and the indices of each sort can be anchored to the entity of the corresponding type. This characteristic of the sort hierarchy comes from the feature-type representation schemes and is used to check constraint satisfaction. With this modification, the lexical signs for "increase", "cardiac output (co)" and "direct response (dr)" change into the forms shown in Figures 7, 8, and 9.

We can specify that the "increase" is a kind of "increase" type and has "c-v" type as its object syntactic role in Figure 7. And from Figure 8 and 9, we can see that "co" is classified as a noun in terms of pos (part of speech) and classified as "c-v" type, and that "direct response (d-r)" is also classified with pos noun and classified with "phase" sort.

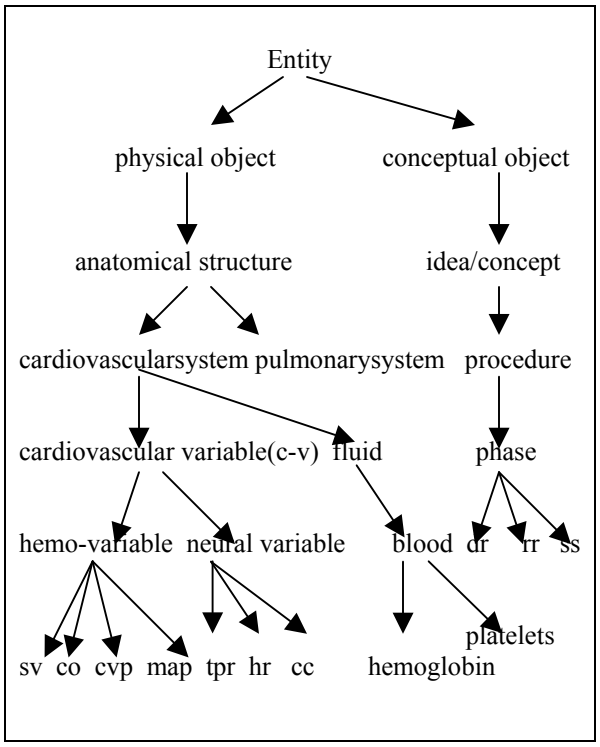


Fig 6. Ontology of the Cardiovascular Domain (partial)

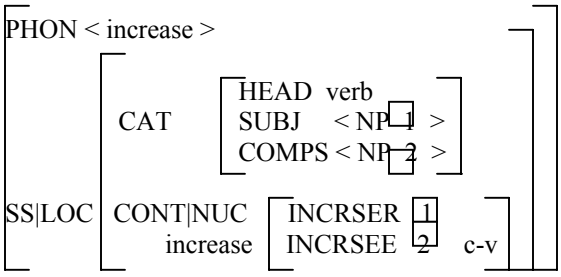


Fig. 7. Lexical Sign for "increase"

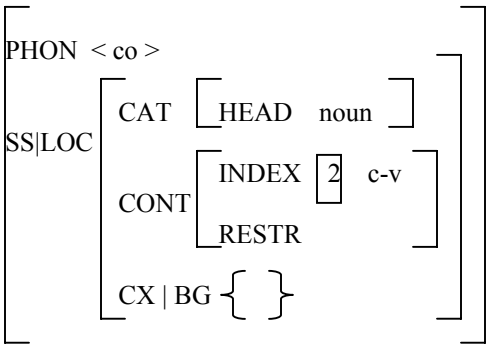


Fig. 8. Lexical Sign for "cardiac output"

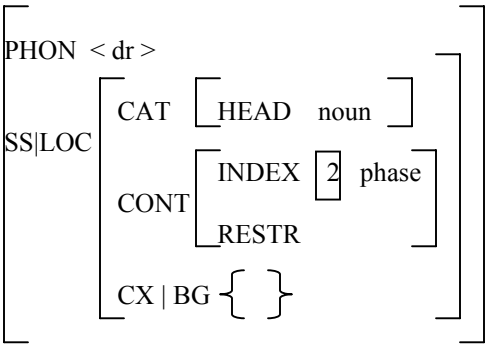


Fig. 9. Lexical Sign for "direct response"

Unification of indices proceeds in the same manner as unification of all other typed feature structures (Carpenter, 1992). The parsing of sentence (3) fails, because it attempts to unify an index of sort "cardiovascular-variable (c-v)" with an index of sort "phase". This attempt leads to the decision that these features are incompatible, and the parser fails to produce any result. In contrast, the parsing of sentence (1) succeeds, since the lexical sign of "central venous pressure (cvp)" comes with an index of sort "cardiovascular-variable (c-v)" that is compatible with the sort of the verb "increase," so the two indices can be unified, and then the parser produces the desired result.

The main advantage of this approach is to apply semantic information during the syntactic processing to obtain a result. Moreover, we can specify selectional restrictions in the HPSG hierarchy of feature structures using a qfpsoa, and we can represent this information in the lexicon. When words with similar semantic content have the same semantic restrictions, we can represent them with qfpsoas in the ontology, so we can represent lexical entries in a more concise form. For

example, the verbs "increase," "decrease," and "change" all behave the same way and can be classified in the same category in the cardiovascular domain.

Our system must parse and respond to short answers, made up of fragments of various kinds. Example (7) shows an answer that consists of a verb phrase only. We have included both the tutor question and the student answer, because understanding the answer requires using information from the question.

(7) T: What does the baroreceptor reflex do ?
S: Regulates map.

To process fragments, we have to add a small number of grammar rules in which part of the sentential contents come from the contextual feature QUD (question under discussion). Since the question is a *what* question and *what* is the direct object of the verb, we project the fragment into a new sentence with the subject NP from the question as its subject.

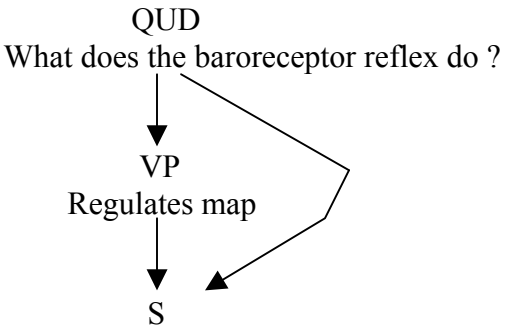


Figure 10. QUD Representation

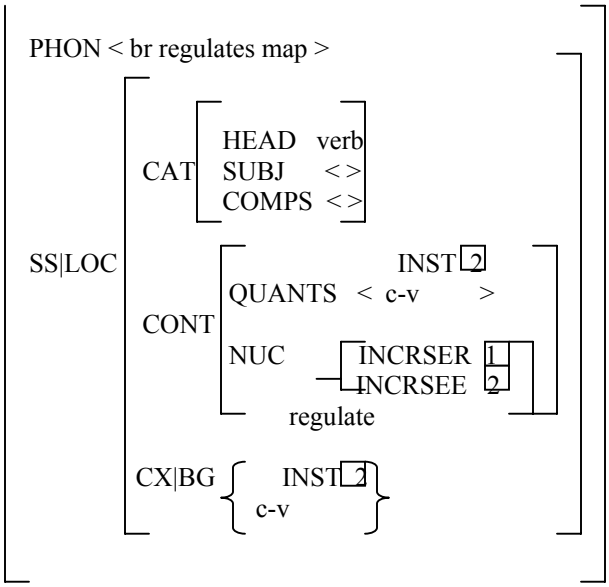


Figure 11.Result of Parsing Example (7)

With QUD we can check the syntactic constraints and construct the intended content of the fragment. The result of parsing this synthesized sentence is shown in Figure 11. Of course, using this approach the parser must check its corresponding semantic type or ontology with its sortal hierarchy during parsing.

Conclusions

We have presented two methods to incorporate selectional restrictions in HPSG. The first approach is to express selectional restrictions in CONTEXT | BACKGROUND features. The second approach represents them using subsorts of referential indices. The first approach requires no modification of the current HPSG feature structures. It makes use of the distinction between "literal" signs that are expressed by CONT and "non-literal" signs that are expressed by CONTEXT. In the second approach requiring the object to denote a

"cardiovascular variable(c-v)" is part of the non-literal meaning. So the second approach is very efficient when an ambiguous word or phrase is encountered during processing since it detects any violation of constraint satisfaction conditions. We found that using the second approach, the parser constructs fewer parse trees, so it is useful when it is adopted in a limited domain. It may not be as useful in a larger domain. But in our domain it makes it possible for us to interpret questions coherently along with their fragmented answers.

Acknowledgments

This work was supported by the Cognitive Science Program, Office of Naval Research, under Grants No. N0014-94-1-0338 and N00014-02-1-0442 to Illinois Institute of Technology, and Grant N00014-00-1-0660 to Stanford University. The contents do not reflect the position of policy of the government and no official endorsement should be inferred.

References

Allen, James. 1995. *Natural Language Understanding*. Benjamin/Cummings, Menlo Park, CA.

Alshawi, Hiyan. 1992. *The Core Language Engine*. MIT Press, Cambridge, MA.

Carpenter, Bob. 1992. The Logic of *Typed Feature Structures*. Cambridge University Press, Cambridge, UK.

Ginzburg, Jonathan and Ivan Sag. 2000. *Interrogative Investigations*. CSLI Publications, Stanford, CA.

Glass, Michael S. (2001). Processing Language Input for an Intelligent Tutoring System. In J.D. Moore, C.L. Redfield, & W.L. Johnson (Eds.), *Proceedings of Artificial Intelligence in Education.* IOS Press, Amsterdam, 210-221.

Lee, Chung Hee, Jai Hyun Seu, and Martha Evens. 2002. Building an Ontology for CIRCSIM- Tutor. *Proceedings of the 13th Midwest Artificial Intelligence and Cognitive Science Conference*. Chicago, IL. 161-168.

Michael, Joel, Allen Rovick, Michael Glass, Yujian Zhou, and Martha Evens. 2003. Learning from a Computer Tutor with Natural Language Capabilities. *Interactive Learning Environments,* 11(3), 233-262.

Pollard, Carl and Ivan Sag. 1994. *Head-Driven Phrase Structure Grammar*. CSLI, University of Chicago Press, Chicago, IL.

Using an Ontology to improve the Relevance of Information Retrieval Results

Louis Green & Injoo Jeong
East-West University
jeoninj@ iit. edu

Yeongkwun Kim*
*Western Illinois University
Y-Kim2@ wiu.edu

Martha W. Evens**
**Illinois Institute of Technology
evens @iit.edu

Abstract

In this paper, we describe the ontology enhanced medical information retrieval (OEMIR) to support a medical student patient simulation system to teach basic science combined with clinical problems in a problem based learning environment. It tries to provide appropriate information and guide users to basic science information using a customizable thesaurus.

1. Introduction

The search and retrieval of useful and relevant information using electronic means, including computerized search engines, has been a challenging task at the least. This is due mostly to the fact that we must specify key words and phrases when performing a search, and there is usually a vast number of documents and materials that contain those words and phrases. This leaves the user with the task of sifting through perhaps thousands of documents in order to find those that are relevant.

Therefore, we are researching methods of providing information that is relevant to someone's given area of interest. The result of this study is our ontology enhanced medical information retrieval (OEMIR) system and this application provides relevant information by performing two major tasks. First, it focuses on a single *domain* (or subject of interest) so that a context will be associated with any keywords or phrases provided. Second, it uses a thesaurus to locate documents that may not contain the exact phrase or keyword, but instead only words that are *related* to what the user is searching for. These two steps will ensure that the user is provided with only the information he/she intended to receive.

The need for a more intelligent information search and retrieval system has been the force behind years or research and development of ontologically enhanced search applications. We have, therefore, studied the works and theories of other researchers and have used this information, along with our own knowledge and expertise, to construct the OEMIR system.

The model upon which this application was built is fairly simple, and it is based upon one fundamental principle: **individual words form one or more concepts, and these concepts have one or more relationships to other concepts.** In using this principle, we were able to construct a structure of data tables and query mechanisms that would ultimately maximize the usefulness and relevance of the information retrieved.

2. Patient Simulation

Diagnosis is a very difficult process for physicians to manage in everyday life. They suspect possible diseases, and build some initial diagnostic hypotheses based on the patient's complaint. Then they try to confirm their diagnosis through interviewing, taking a medical history, a physical examination, laboratory tests, and so on. Illinois Institute of Technology has developed a patient simulation system to support a problem-based learning curriculum [4, 6]. The system produces a simulated patient that serves as basis for both clinical and basic science instruction. The medical student can ask questions about the simulated patient's symptoms and signs, and orders necessary tests. After they get the results from the system, they

can make decision about a diagnosis and possible treatment.

3. Methodology

The basic functionality of the OEMIR system works in much the same manner as your typical search engine in that it searches for documents that contain certain strings of text. However, when presented with the problem of determining how to develop a search engine that was *semantically driven*, it was understood that an intelligent system was necessary. In our research and development of this project, we focused on the Cardiovascular system of our special area of interest.

3.1 Ontological Structure

The underlying theory behind the construction of the ontological database is that, as mentioned earlier, individual words form concepts and concepts can have one or more related concepts. Therefore, the database consists of three tables. The first table is the **DisorderTerms** table, which stores the individual words that are used to construct concepts (terms).

Table 1. DisorderTerms table

TERM	CLASS
Ischemic	A
Extrasystole	N
Pulmonary	A
Hypertensive	A
Orthopnea	N

The second table is the **DisorderConcepts** table, which stores important concepts with respect to the cardiovascular domain texts.

Table 2. DisorderConcepts table

ConceptID	Concept
59	Orthopnea
73	Extrasystole
41	Backward
75	Hypertensive heart disease
62	Pulmonary Hypertension

The third table is the **ConceptRelationships** table, which stores one or more concepts that are related to a given concept in the **DisorderConcepts** table.

Table 3. ConceptRelationships table

ID	Link	Relationship	Descriptor
75	Cause	Extrasystole	
41	Type	Heart failure	
62	Cause	Heart failure	Right-sided
73	Result	Heart Failure	Hypertensive
59	Result	Heart Failure	Left-sided

To illustrate the relationship between these data tables, consider the term “ischemic heart disease”. The words “ischemic”, “heart”, and “disease”—along with their types (noun, adjective, or verb) are stored in the **DisorderTerms** table. The concept itself (“ischemic heart disease”) is stored in the **DisorderConcepts** table. According to the cardiovascular texts, “ischemic heart disease” is a cause of extrasystole, so “extrasystole” has been entered in the **ConceptRelationships** table as being related to “ischemic heart disease”. Therefore, whenever a user wishes to search for information regarding “ischemic heart disease”, he/she will also be given information about “extrasystole” and any other concepts that are related to their primary search term.

In constructing the database and its tables, we utilized various texts from the book, Harrison’s Principles of Internal Medicine [1]. A sampling of these texts was examined and key words were noted. Additionally, these key words were automatically associated with both the subject of interest (cardiovascular system) and the current topic of discussion as determined by the title of the document.

It is worth mentioning that the only reason the individual terms are stored—along with their types—is that it is required in order for term decomposition (finding embedded terms) to be possible.

3.2 Embedded Terms

Due to the complex and sometimes ambiguous nature of the English language, for any search term consisting of multiple words, there exists the possibility that additional terms are "embedded" within it. For example, "forward, backward, or high-input heart failure" imply "forward heart failure", "backward heart failure", and "high-input heart failure." As defined in this program, an embedded term exists when there are multiple adjectives that immediately precede one or more nouns. Each individual adjective can be applied to the noun(s) in almost every case.

Example [*a = adjective and n = noun*]:

> "ischemic (a) or hypertensive (a) heart (n) disease (n)" **implies** "ischemic (a) heart (n) disease (n)" and "hypertensive (a) heart (n) disease (n)"

In each of the above cases, embedded terms were extracted by performing the following steps.

1. **For a given string of words, locate a string of multiple adjectives**

2. **Locate the nouns or verbs that belong to these adjectives**. The meaning of the word "belong" is the nouns or verbs that follow one adjective and come immediately before the next adjective (or end of sentence). For example, consider the string:

   "Forward (a) or Backward (a) heart (n) failure (n) and ischemic (a) or hypertensive (a) heart (n) disease (n)"

   In this string, the nouns "heart" and "failure" follow the adjectives "Forward" and "Backward" and immediately proceed the next adjective, which is "ischemic". Thus, "heart" and "failure" are said to *belong* to "Forward" and "Backward", and the nouns "heart" and "disease" belong to the adjectives "ischemic" and "hypertensive". Therefore, the embedded terms within this string will be:

   - Forward heart failure
   - Backward heart failure
   - Ischemic heart disease
   - Hypertensive heart disease

3. **Apply each individual adjective to the noun(s) that belong to it**.

3.3 Query Execution

When the user supplies one or more search terms, the terms are parsed and any embedded terms are added to the list of search terms. For each of these search terms, an SQL query will be executed in order to find any related terms. The query consists of a primary query (outer) and sub-query (inner). The inner query searches the **DisorderConcepts** table and returns the *ConceptID* of the record whose *Concept* field value is equal to the current search term. The outer query then locates all records in the **ConceptRelationships** table whose *ID* matches the *ConceptID* value found in the inner query (notice that *ConceptID* field in the **DisorderConcepts** is a foreign key in the **ConceptRelationships** table). Figure 1 shows the OEMIR system model. Figure 2 shows a query example.

4. Conclusion

With careful research and development, we were able to design a search engine capable of intelligent information retrieval. This is accomplished by limiting the scope of the search to that of the selected domain—or subject of interest, parsing the search terms to extract embedded terms, and using the original and embedded terms to gather related concepts. The original, embedded, and related concepts are all, therefore, used to locate relevant documents. In order for us to see that this will improve the overall speed and accuracy at which relevant information can be found, we are planning to manage further investigations by applying our approach to patient simulation system developed at Illinois Institute of Technology.

References

Isselbacher, K., Braunwald, E., Wilson, J., Martin, J., Fauci, A., and Kasper, D., 1994. *Harrison's Principles of Internet Medicine*. McGraw-Hill, Ic., New York.

Kim, J., Evens, M., Jeong, I., and Trace, D. 2001. Generating Concept-Feature Relationships for Patient Simulation Using a Textbook Index. *Proceedings of the ISCA 16th International Conference on Computer and Their Applications*, Seattle, WA, pp. 192-195.

Kim, J., Evens, M., and Trace, D. 2002. Building an Authoring Tool for Concept-Feature Relationships by Decomposing Phrases in the Text. *Proceedings of the ISCA 17th International Conference on Computer and Their Applications*, Seattle, WA, pp. 261-264.

Kim, J. 2002. Intelligent Information Retrieval System: Merging Basic and Clinical Science into a Medical Student Learning Program. *Ph.D. dissertation, Department of Computer Science, Illinois Institute of Technology, Chicago,IL.*

Trace, D., Evens, M., Naeymi-Rad, F., and Carmony, L. 1990. Medical Information Management: The MEDAS Approach. *Symposium on Computer Applications in Medical Care, Washington, DC, pp. 635-639.*

Yang, K., Evens, M., Trace, D., and Chern, J. 2000. Patient Simulation for Medical Education. The 16th International on Conference Advanced Science and Technology, pp. 141-148.

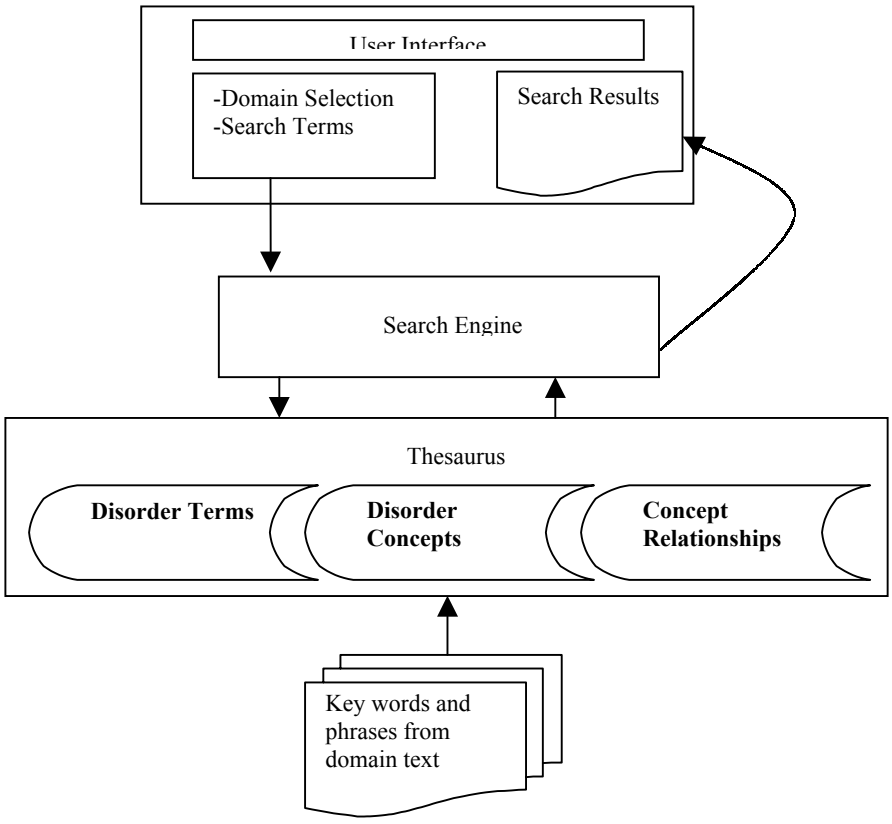


Figure 1. OEMIR System Model

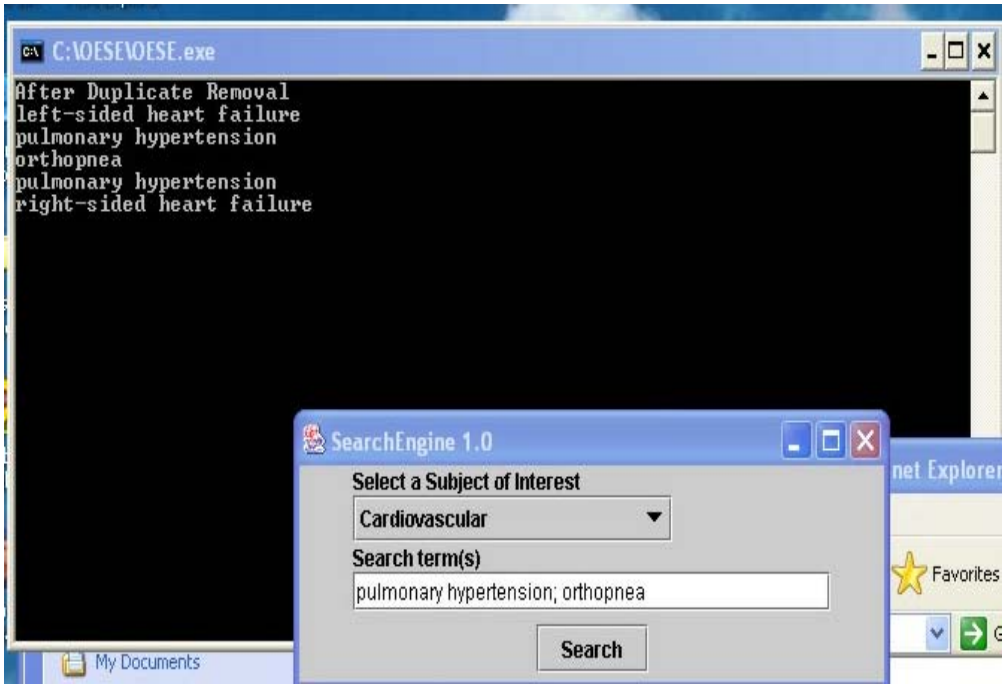


Figure 2. A query example

Creation of a Style Independent Intelligent Autonomous Citation Indexer to Support Academic Research

Eric G. Berkowitz, Mohamed Reda Elkhadiri

Department of Computer Science
Roosevelt University
1400 North Roosevelt Boulevard
Schaumburg, IL 60173

Abstract

This paper describes the current state of RUgle, a system for classifying and indexing papers made available on the World Wide Web, in a domain-independent and universal manner. By building RUgle with the most relaxed restrictions possible on the formatting of the documents it can process, we hope to create a system that can combine the best features of currently available closed library searches that are designed to facilitate academic research with the inclusive nature of general purpose search engines that continually crawl the web and add documents to their indexed database.

Introduction

RUgle is a system composed of three major components, a Web scanner, a document analyzer and cross-referencing system. Using the system it is possible to perform searches of academic and other research papers extracted from the World Wide Web without needing to sort through the seemingly endless number of loosely related or valueless Web documents returned by traditional searches.

Background

Current search engines for the World Wide Web fall into one of two categories. The first is the general purpose search engine examples of which are Google.com, Yahoo.com and AltaVista.com. The engines attempt to index the entire expanse of the World Wide Web using the words and phrases in the document as tokens with which to build an index. Using these search engines, one can search a very large collection of documents from the Web. Unfortunately, this abundance of documents leads to the retrieval of many documents that may be of little worth. The lack of focus on quality also creates collection that include a large number of papers but may not include many scientific papers (Lawrence, Bollacker and Giles 1999). Additionally, the use of the entire collection of words, in the document as tokens for the index can cause a search to retrieve many irrelevant documents, particularly when the words in the search term are quite common (Bradshaw Sheinkman and Hammond 2000). For example, if one is seeking information on the content of a will one might use the word "will" as a search term. Since "will" has other meanings as a verb and a proper noun, the common search engines will return a plethora of irrelevant documents. One might then be tempted to use the search term "will and testament." This phrase might however lead to the exclusion of many relevant documents since the word "testament" does not always accompany the word "will," even in the correct context. Another problem is that if one finds a relevant document in the list returned by the search engine, having this document does not always lead to additional relevant documents. Even if the document in question has hyperlinks to other documents, the reason one might link from this document to others on the Web are many and varied and not always based on any form of relevance. Since the links in a document on the Web are placed there for the specific purpose of facilitating navigation around the WWW they incorporate the document authors' ideas, both conscious and subconscious regarding other documents a reader should view. This brings up three major shortcomings of navigating the Web via links in existing documents.

1. The link's sole purpose is to facilitate the navigation from one document to another and therefore incorporate the ideas of the designer, both conscious and subconscious, regarding other documents a user should view.
2. Designers of documents usually possess a very poor knowledge of what is available outside their own collection of documents or their own site and therefore, given the purposeful nature of links on the WWW, those links that a designer provides to other sites are often few and inadequate.
3. Related documents are often not linked at all since the designer of one document may not be aware of the existence of other relevant documents, or may simply not care to provide links to the other documents, or may even want to impede (or at least not facilitate) navigation to those other documents.

Thus attempting to navigate the WWW from within a single site fits the old saying of not being able to see the forest for the trees; one often cannot move away from one location even if a bird's-eye view would allow one to see that useful information is very near.

Library collections such as CiteSeer attempt to resolve many of the issues listed above. Documents are only included in the collection to be searched if they meet some criteria of relevance. The content of the document is not used as a single collection of equally weighted tokens and

one can currently only search for terms in the authors' names, paper title, or body. Thus one is more likely to receive a list of valuable, relevant documents from a search. Still these mechanisms suffer from shortcomings of their own. They rely on closed collections of documents. Addition of a document to the collection usually requires manual data entry or manual data correction making the process quite slow (Lawrence and Giles 1999(3)). For example, the DBLP engine relies on manual data entry. In order to facilitate the indexing mechanisms used, the documents are usually restricted to a single domain or a collection of domains. For example, CiteSeer uses a list of URLs known to contain papers and actively attempts to download papers from these sites with secondary reliance on active searching. It is, however, restricted to a limited set of domains and still remains quite dependent on manual data entry. Authors are encouraged to log on to the site and to manualy correct the entries for their papers. Such search engines also usually require documents to be submitted or at least the submission of a top level URL to be searched. Manual filtering of the documents or top level URLs is required to guarantee the domain restricted nature of the collections. They do not find documents on their own. Thus, while they may aid in finding only quality, relevant documents when searching for information in an included domain, they exclude a wealth of other quality, relevant documents that have not been added to their collection.

RUgle is the first product of our research into producing a system that combines the best features of both types of searches while minimizing the effect of their shortcomings.

Design

RUgle currently operates using five computer systems running Solaris, Linux, and Windows as shown in Figure 1. The Solaris server, called Epsilon, is the main database server and contains the cross reference database with information extracted from the papers along with tracking information used by the crawler. A Linux computer, named Xeon, serves as a backup database server. Another Linux computer, called RUgle, runs the code for the crawler and document analysis interacting directly with the database server and the WWW. The windows machine, called Winbox runs code required for analysis of documents in Microsoft proprietary formats and operates under control of the document analyzer running on RUgle. The second Solaris server, named Lambda, runs the user interface code described later in this paper. From a logical perspective, RUgle's main engine is a single loop that works as follows:

- A single unprocessed citation is retrieved from the database.
- The citation is sent to the main crawler that attempts to download the referenced document using traditional search engines.
- The downloaded document is analyzed and:
    - the title and authors' names are entered into the database.
    - the bibliography entries are entered into the database and stored for later processing by the system.
- The next bibliography entry is retrieved and the loop repeats.

This logical flow is shown in Figure 2.

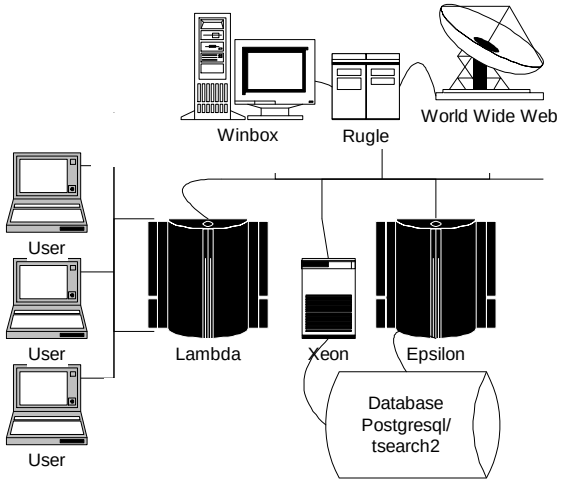


Figure 1

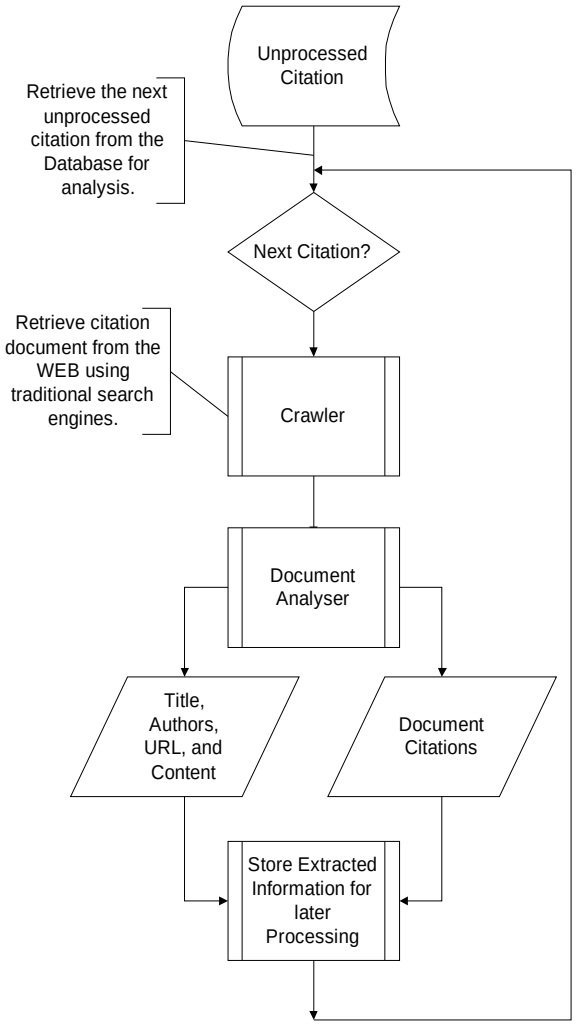


Figure 2

Active Acquisition of Documents on the Web

In order not to be restricted to a small collection of submitted documents, RUgle must actively scour the Web for documents to add to its collection. Thus RUgle employs a crawler or Web spider. RUgle's crawler is however very different from traditional crawlers in the manner in which it moves from document to document on the Web. It does not use hyper-text links to do so. As described above, these links do not always represent any objective form of relationship between documents. Instead, RUgle analyzes the content of the document currently being processed. RUgle extracts from the document the name of the author, the title and the bibliography. RUgle is designed with the assumption that a document of value will contain citations to other works. After a document is analyzed, RUgle iterates over the bibliography and extracts the components of each entry, particularly the authors of the cited work and its title. It then uses information provided by traditional search engines to try to locate the cited work on the Web. Thus, rather than relying on hyperlinks or a set of common tokens, works are considered related because one cites the other. Since a bibliography lists cited works, it is far less susceptible to purposeful manipulation than hyperlinks placed to facilitate Web navigation. Papers that are found are then downloaded from the Web. Their title, author and bibliographies are extracted and then queued for searching in a breadth first manner. The issues encountered in developing a crawler of this kind are involved in actually extracting the critical information from a document. It is for this reason that the traditional library collections require a full set of BibTex entries to be submitted with each paper to be indexed that incorporate information on the paper itself and the complete bibliography.

Formatting Languages

In order to process the documents and find the elements RUgle needs to index them and continue scanning the Web, RUgle needs to be able to analyze the document content. Given the almost ubiquitous nature of Adobe's Portable Document Format (PDF) on the Web we began by processing this type of formatting. Rather than parsing the formatting tags in the PDF, it was decided to use available tools designed to render a PDF document as ASCII text while maintaining as close an approximation as possible to the documents original layout. RUgle's first step in processing a document is to convert it to text. After converting the document to text RUgle begins to analyze its content.

While the PDF format is very common on the Web, we found it to be far more common in the sciences, slightly less common in the social sciences, and even less so in the humanities. If RUgle were not to be skewed against indexing papers in the humanities and toward the hard sciences it needed to be able to process documents as they are posted on the Web by researchers in those fields. Our investigation revealed that it would be crucial for RUgle to process documents in Microsoft Word (doc) format for it to have a more universal indexing ability. Doc format is far more complex than PDF, and, unlike PDF, it is not an open standard. Thus there are few tools to process documents in this format and most of the available ones have shortcomings that preclude them from processing even large subsets of documents in this format, particularly documents with embedded graphics. Even if such a tool were available, it would require constructing a whole new interface to the parsing routines and complicate system enhancements. It was decided therefore to convert doc format documents to PDF as a preprocessing stage and then let the PDF processing engine take over. The most reliable tool for the task proved to be Microsoft Word itself. RUgle searches the Web for documents in both PDF format and doc format. Documents in doc format are automatically passed through Microsoft Word and rerendered as PDF documents for further processing.

Columns

The first hurdle to processing documents originally designed for publication in journals is that many of them tend to be formatted as two columns. Linear processing of such documents yields collections of sentence fragments and renders the document useless. This because, while visually the document is in two columns and a human reader easily identifies columns on a page and treats them as separate pages when reading, the columns have no manifestation in the actual stored document which is stored as a collection of pages with linear text from left to right across lines of the full page. RUgle must determine if a document is indeed formatted in columns, and, if it is, recreate the document in one column with linear text.

RUgle does this by scanning a page and searching for word gaps (spaces) that appear in the same place in the text in all lines of the page. RUgle is built on the assumption that the probability of such a gap being a random occurrence is very low. If RUgle finds such a gap it treats it as a column break and rerenders the document wrapping the text from the left margin to the column break for the left column and the end of the column break to the right margin for the right column. This analysis is particularly difficult on pages that include bibliography entries since such entries often incorporate various amounts of spacing and indentation in the bibliography itself.

Analyzing Document Content

Since RUgle is designed not to require documents to be submitted but to locate them independently and also not to require manual data entry, RUgle needs to be able to find the data it needs to process the document on its own. Were we to restrict RUgle to a limited domain, one would think this task would be somewhat simple since, for example, papers on computer science are usually formatted using the style dictated by IEEE or the ACM. This domain specific information proved however, to be of limited values since

RUgle is intended to be completely generalized and index documents in all fields.

Our first efforts were to build parsers for the most common document formatting styles in use including APA, PAR, Chicago style, AMA, ACM, IEEE, MLA, and AAAI. Several such parsers were written and incorporated into early analysis subsystems for RUgle.

Quite surprisingly, we discovered, that even for may documents claiming to be formatted according to the dictates of a particular style, they do not actually follow all the rules of that style (Lawrence and Giles 1999(2)). While aesthetically, the papers tend to adhere closely enough to the style dictates that they look good when grouped together with other documents that also claim to follow the given style in a single publication, they actually deviate from the fine details to a sufficient degree to significantly complicate computer analysis. Distinctions that are lost to the human eye are quite apparent to any scanning and parsing algorithm. Thus, though we had created a large variety of parsers we were having difficulty analyzing even the types of documents for which they were specifically written. We learned that to function as intended, RUgle needs to not only process documents that adhere to a wide variety of formatting styles, such as APA, Chicago style, AMA, ACM, IEEE, MLA, and AAAI., but also documents that only loosely follow one of these styles, and documents that do not follow any specific style at all. Since writing an entire collection of parsers, even if a doable task, proved to be of limited value, we attempted to determine what could be assumed about the way people format documents.

We discovered that the majority of bibliographies fall into one of two categories: those that place the date at the end of the citation (an end date entry or EDE) and those that place the date in the middle of the citation (a mid-date entry or MDE). In the latter case, the date, possibly along with some collection of punctuation, usually separates the authors' names, that precede it, from the title of the publication that follows it. In the former case it is usually punctuation alone, and sometimes little more that a single space, that separates the authors' names from the title and they usually are in that order in the citation.

Thus we built a style independent, three-tiered system for processing bibliographies. The lowest tier is made of two parsers, one that processes EDEs and one that processes MDEs. Above that is a tier that attempts to unify stylistic elements of each type of entry so that the parser can successfully analyze it. At this level RUgle attempts to determine what form of punctuation the author used, such as interspersing commas or periods between bibliography entries and enclosing the date in parentheses. Once this information is determined, the entry is modified to conform more closely to a single style understood by the parser. At the top tier, RUgle attempts to determine whether the entries should be sent to the MDE unifier and parser or the EDE unifier and parser.

The other issue is to find the title and author of the document being analyzed itself. For this we integrated into RUgle the algorithms originally developed for an automated literature review system we had been developing. The algorithm determines the title by assuming the approximate location of the title in the document and then removing all the text that is not the title thereby leaving only the title. RUgle employs a two tiered system for extracting the title. The first tier attempts to determine where the title and authors' names should appear in the document. The second tier attempts to discard all other information from that section of the document and to determine from the remaining text, the title of the document and the names of its authors.

User Interface

Conventional search engines maintain complete full text indexes (FTIs) of the documents in their database. This allows a user to search for a document using any word or phrase that appears in the document. The document is treated as a single string of tokens with equal weight. The weight associated with any given token in the document is increased based on the number of times it appears in the document. Thus words in parenthetical phrases are given the same weight in indexing as words from a title, section heading or other presumably more important sections of a document that are intended by the author to summarize and typify the content of the document. RUgle maintains a full text index of only the title and authors' names. While it is the intention to maintain an FTI of only the title, the current algorithm confuses the title and author to a sufficient degree that it was decided to index both entries as a single unit. The result is that a search for a term such as "executive compensation" results in a list of documents that:

1. meet RUgle's criteria for quality. They have a stated title, author and a bibliography.
2. contain the words in the search term in the title of the document.

That is, by the criteria used to design RUgle, a quality document that the author believed addresses the terms in the search. The results page of a RUgle search is shown in Figure 3.

When a user select a title from the list on the results page, the user is presented with the following information:

1. The title and author of the paper.
2. A list of all of the citations in the bibliography of the paper.
3. A list of all cited papers that also exist in RUgle's database.
4. An image of the first page of the selected paper.

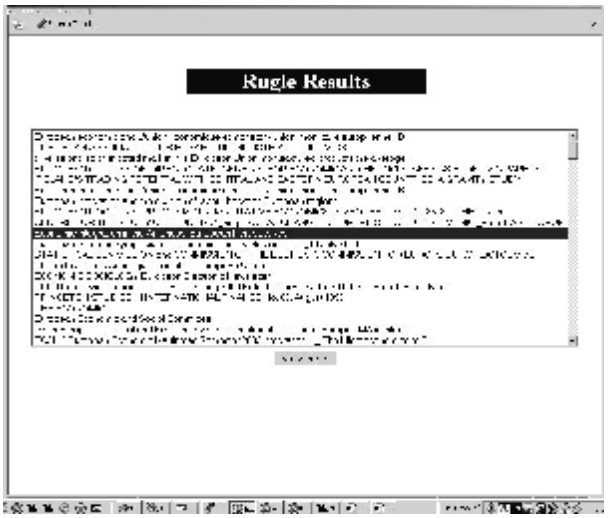


Figure 3

A user can decide to:

- return to the list of titles.
- download the paper.
- select a new paper from list of cited papers that are in RUgle's database and view it. The document display page is shown in Figure 4.

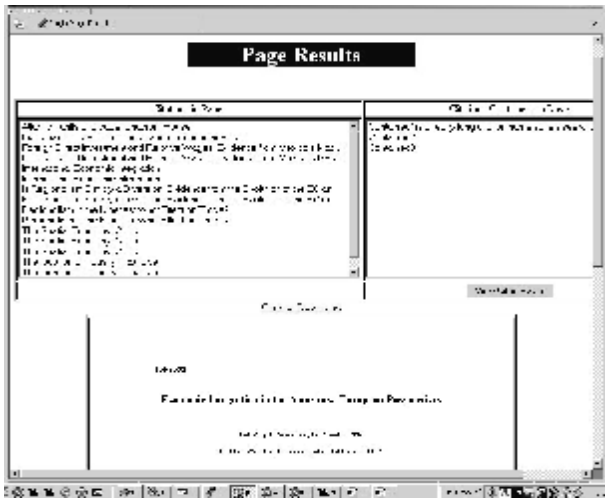


Figure 4

Analysis of Current Performance

RUgles has currently downloaded approximately 700,000 documents from the WWW. To determine how well the current algorithms are performing their tasks, we selected a random sample of 200 papers and reviewed the title and author names that RUgle had determined for them. The results are shown in Table 1.

Criterion	Percentage of Papers
Exact Title	43.27%
Title + Extra Text	11.54%
Title + Author as Title	5.77%
Partial Title	12.5%
Total Titles: 73.08%	
Exact Author	25.96%
Author + Extra Text	22.11%
Partial Author	2.88%
Total Authors: 50.95%	

Table 1

The performance for the bibliography extraction is somewhat lower. From 418174 papers scanned for a bibliography, entries were only extracted from 133882 of them. Much of this can be explained by RUgle's inability to filter out documents whose content makes them appear academic but which in reality are not and do not contain bibliographical data such as professors' resumes, course syllabi, course notes, industry white papers, etc.

Conclusion

RUgle is quickly becoming viable as a system for general academic research. Its mechanisms, however, ignore much of the visual information hidden in a document as described in Berkowitz and Mastenbrook 2002. Utilizing this information would enhance the quality of the analysis and the indexing and it is our intention to incorporate the Purpose Encoding Document Abstraction language into future versions of RUgle. RUgle's generalized scope and its ability to index papers from all fields of research are helping to make it a useful research tool in its own right. The system's current ability to locate and then analyze bibliographies is lacking and research is being done on how to improve this aspect of the system. We are also working to improve the ergonomics of the user interface. One problem we are currently facing is the magnitude of the database which currently stores over seven million bibliography entries even with RUgle's limited ability to extract this information. Queries on the database are beginning to take more time than desired, slowing down both the updates and the end-user interface. We are researching methods to improve the overall design of the database along with methods for distributing and replicating the data to accelerate read-only queries from the user interface.

References

Eric Berkowitz and Brian Mastenbrook. “A Visual Formatting Purpose Representation Language to Enhance Automated Document Classification, Retrieval, and Indexing” In Proceedings of the 16th Annual Midwest Computer Conference, Schaumburg, IL, April 2002.

S. Lawrence, and C.L. Giles, "CiteSeer: An Automatic Citation IndexingSystem," IEEE Communications, Volume 37, Number 1, 1999, pp. 116-122.

S. Lawrence, and C.L. Giles, "Digital Libraries and Autonomous Citation Indexing," IEEE Computer, Volume 32, Number 6, 1999, pp. 67-71.

S. Lawrence, C.L. Giles, and Kurt D. Bollacker, "Autonomous Citation Matching," Proceedings of the Third International Conference on Autonomous Agents, ACM Press, New York, 1999.

S. Lawrence, C.L. Giles, and Kurt D. Bollacker, "Indexing and Retrieval of Scientific Literature," Proceedings of the Eighth International Conference on Information and Knowledge Management, 1999, pp. 139-146.

S. Bradshaw, and K. Hammond, "Automatically Indexing Documents: Content vs. Reference," In Proceedings of the Sixth International Conference on Intelligent User Interfaces, San Francisco, CA, January 14-17, 2002.

S. Bradshaw, A. Scheinkman, and K. Hammond, "Guiding People to Information: Providing an Interface to a Digital Library Using Reference as a Basis for Indexing," In Proceedings of the Fourth International Conference on Intelligent User Interfaces, New Orleans, LA, January 9-12, 2000

Developing Surrogate Simulation Models for a Computationally Intensive Real World Application

Anshu Manik, Abhishek Singh, Shengquan Yan

Department of Civil and Environmental Engineering
University of Illinois, Urbana-Champaign
NCEL, 205 North Mathews Avenue, Urbana, IL
manik@uiuc.edu; asingh8@uiuc.edu; smyan@uiuc.edu

Abstract

In the field of Engineering, computer models have become ubiquitous for prediction of physical or stochastic phenomena. However, such models are constrained by their computational requirements, and it is often necessary to couple them with simpler models to speed up the process of decision-making. This paper investigates various machine-learning based approaches to build surrogate models for a pavement construction payment-risk prediction model. These include neural networks, decision trees, support vector machines, and their hierarchical combinations. The paper aims at highlighting some of the issues in building such models from an engineering perspective.

Introduction

Several states in US follow end result specifications (ERS) in highway construction. Under ERS framework a contractor is paid the bid amount if he achieves the desired quality in the highway constructed. Poorer quality construction would lead to a reduction in the amount paid to the contractor and better quality could fetch bonus payment (Buttlar, 2000).

Quality is estimated by measuring certain quality characteristics like density, air voids, asphalt content etc. But the measurement process has certain uncertainties involved with them because of measurement errors, operator error, instrument bias etc (Buttlar, 2000). This means that if the payment to the contractor is based on these measurements there is a probability that he either gets more pay than he deserved or is paid less than that. Payment risk is defined as the difference between the payment made to the contractor based on the above-mentioned quality characteristic measurements and the ideal pay based on the actual quality of the pavement constructed.

It is desirable to determine the magnitude of risk involved with such construction processes and relate them to the factors that may be affecting them. This knowledge can be used to reduce this risk.

The parameters that have been identified to affect risk significantly are (1) Production variability (2) Measurement variability (3) Targeted mean quality (4) Sample size (5) Bias in measurements of contractor, agency and third party (6) Upper and Lower specification limits for the quality characteristic and (7) Tolerance allowed in comparing the measurements of the contractor, agency and third party.

It is desirable to have a simulation that would take the above-mentioned parameters as input and would predict risk in the form of a risk plot showing magnitude of risk verses the mean quality characteristic and confidence intervals for the risk. Monte Carlo based simulations are generally used for this purpose. But such simulations are often computationally intensive and therefore not very useful for batch processing because of prohibitive time requirements. This paper explores the possibility of using neural networks, decision trees and support vector machines for developing surrogate simulation models.

In this case although the simulations work with generation of normally distributed random numbers the calculations and conditional processing of data make the output non-linear and harder to map.

Regression Analysis

A step wise approach was followed while building models for the risk problem, starting with the simplest and subsequently going to the more complex. The first types of model tested were the linear (in parameters) multivariate multiple regression models. The training data was used to calibrate quadratic and cubic models, while the test data was used for validation.

The general equation used for the regression model was:

$$Y = AX$$

Where Y is the matrix of the dependent variable, X is the matrix of some function (polynomials in our case) of the independent variable, and A the parameter matrix.

Linear regression represents the dependent variables (which in this case were average risk, and upper and lower confidence intervals on risk), as a linear combination of some function of the independent variables. For good regression analysis it is critical that there is a good understanding of the inherent phenomena to be modeled. In this case since conditional stochastic simulations were used to generate data, it was difficult to formalize the

phenomena in terms of mathematical functions of the independent variables. Moreover it was known that there were significant nonlinearities and discontinuities in the phenomena that were difficult to characterize.

The regression model did not yield good results on testing. The errors were high, and the trends in the data were not well captured. More complex regression methods could have been tried, but were held back due to the inability to formally represent the inherent phenomena. While regression is a good way to parameterize known functions, it is not a very good option for modeling of complex phenomena. Interestingly, the regression analysis, though apparently unsuccessful, gave valuable insights into dependencies of the independent and dependent variables. The original 'X' matrix was found to be ill-conditioned due to high correlation between the upper and lower tolerance limits. Removing one of these variables led to more stable results for subsequent models.

Neural Network Surrogate Model

The first machine learning tool used to model this phenomenon was the fully connected, feed forward neural network (Russel & Norvig, 1995). The input layer had eleven nodes, each corresponding to one parameter. The output was the mean risk and higher and lower confidence limits corresponding to one value of chosen quality characteristic. A Monte Carlo based simulation was developed to generate 60,000 data points for training purpose and 25,000 data points for testing. The input values were chosen randomly from within their feasible range. The data sets were normalized to fall between 0.0 and 1.0.

Several network architectures were designed and tested. The MSE were determined with 50,000 random cases in the training set and additional 25,000 cases in the test set. Number of cycles run in training was fixed at 150. Preliminary runs indicated that after 150 cycles generally the change in MSE with cycles was very small. The steps to the optimal neural network architecture have been briefly described below.

1. First a network with one hidden layer was tried. But the errors were higher than acceptable.
2. Two or more hidden layers were expected to capture the nonlinearities of the phenomena better, so several two hidden layer architectures were tested. MSE in all the cases, as shown in figure 1 remained between 0.115 and 0.16.This shows that two hidden layers definitely gives lower error than one hidden layer. Also, comparing all the MSE plots the network with 15 neurons in each of the hidden layers gave the best result.

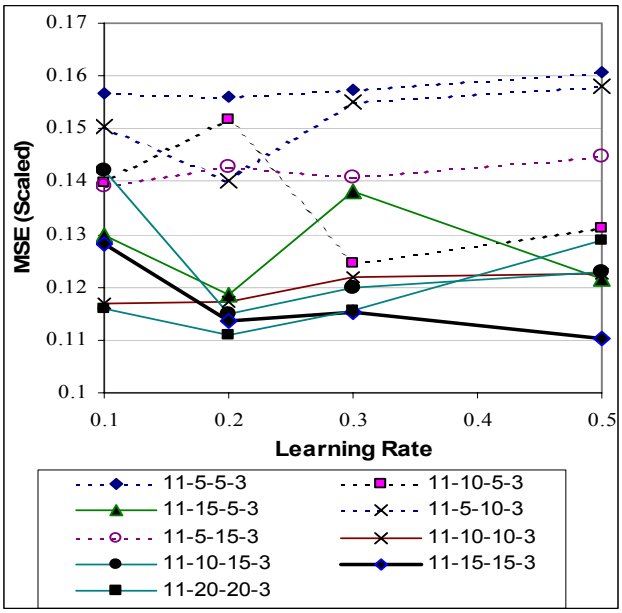


Figure 1: MSE when tested on test data

3. In most of the cases MSE increased as learning rate increased from 0.2 to 0.3 and increase was lesser in some cases beyond a value of 0.3. In some cases, however, MSE actually went down a little for a learning rate of 0.5. The later case thus seemed to be unstable compared to a learning rate of 0.2. Similar attempts were made with 3 hidden layers but the MSE were found to be much higher than those from two hidden layers.
4. The chosen neural network (11-15-15-3) was run for 500 cycles and the errors against the number of cycle are plotted in figure 2. Considering constraints on the processing time it was decided that the network could be trained for 250 cycles without compromising on solution quality.

An important factor in training neural network is to have sufficient number of training cases while avoiding over-fitting. In all the cases plotted so far 50,000 training cases were used. To determine the optimal number of training cases required the chosen network was trained with 60,000 cases and then with 85,000 cases. The test errors in both the cases were almost identical. However, the decrease in error when using 60,000 cases as compared to that with 50,000 cases was significant. This shows that 60,000 cases may be a good number for the training set. Moreover it was also found that training errors were comparable to the errors on test cases not present in the training set. This indicates that the network is getting trained to adequate precision but over-fitting has not occurred.

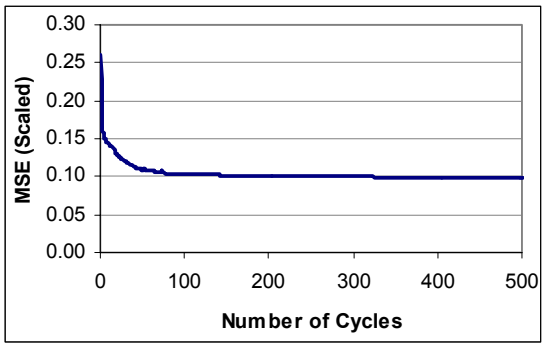


Figure 2: MSE against number of cycles of training for optimal network

In summary the neural architecture that was chosen to map the pay factor risk is as follows:

- Fully connected, feed forward, back propagation
- Learning Rate : 0.2
- Hidden Layers : 2
- Neurons in hidden layers : 15, 15
- Number of Cycles : 250
- Momentum : 0
- Basic Architecture : 11-15-15-3
- Training Cases : 60,000
- Test Cases : 25,000

Results from Neural Network Model

Figure 3 is one way to contrast the predicted values of payment risk with the expected values. In this plot the dark solid line represents the expected value of mean risk for a particular set of input parameter values. The lighter band around it is formed by plotting the corresponding predicted values. The thickness of the band shows the spread in prediction. Smaller spread would mean that the predicted mean risks are closer to the expected mean risks and hence errors are small and vice-versa. In the sample result the spread in mean risk is almost ± 7.5% for the first part of the plot.

Figures 4 and 5 show the results for two standard risk plots. The entire plot is obtained by running the neural network simulation for each data point required on the plot. Figure 4 shows that for these cases the predicted values are very close to the expected values. If the network does predict risk with this much of accuracy then the chosen network would be considered as a good surrogate simulation model. Figure 5 shows another representative case where the predicted values are significantly away from the expected ones especially for higher voids values. This shows that although the average error associated with the network predictions are acceptable; in some particular cases the errors may be unacceptably high (especially around very high or very low input values). This provides motivation for trying some improvements in the surrogate model so that such errors can be reduced.

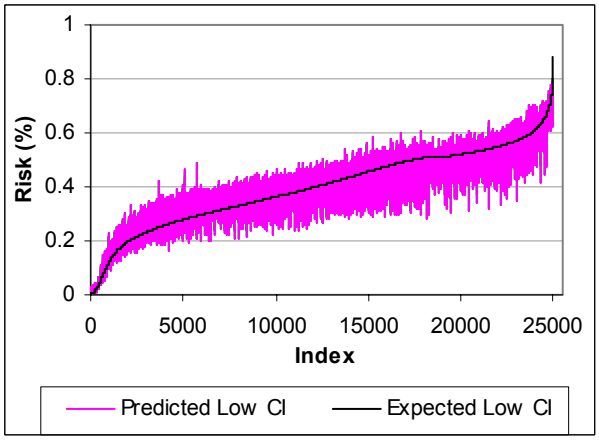


Figure 3: Expected Vs predicted mean risk from the neural network

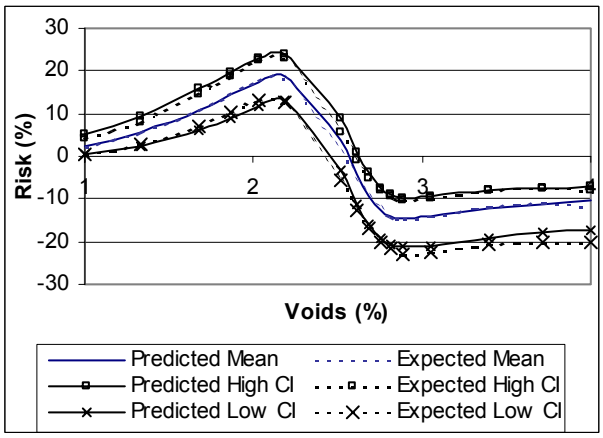


Figure 4: An example of payment risk predicted by the neural network

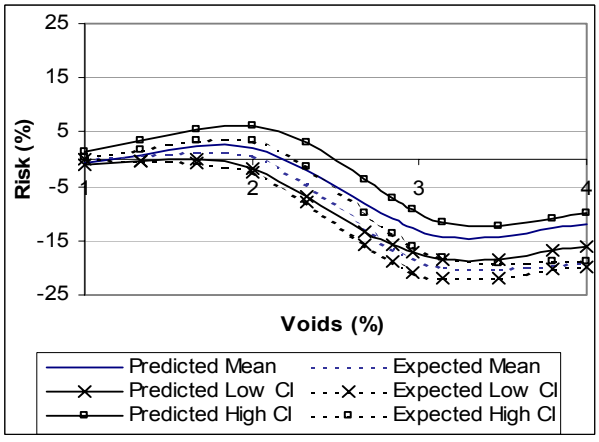


Figure 5: Another example of payment risk predicted by the neural network

Hierarchical Model

It was earlier seen that the simple 2 layer neural network gave high errors around the extreme points. This result was expected since artificial neural networks typically capture the trends in data but do not do too well at predicting extreme events. In this case it was felt that the neural network had problems dealing with very high and very low values simultaneously. One solution was to split data into positive risk and negative risk cases and have separate neural networks for each. It was hypothesized that using this kind of a hierarchical approach would lead to increased accuracy at the extreme points. Hierarchical models typically use a top-down approach in which the complexity of prediction increases as we go further down. The data set was split by using a robust classification tool which would classify the data into expected positive or negative risk. Then prediction tools specific to each domain would be used, leading to more accurate results all over. Figure 6 shows the flowchart for the hierarchical model.

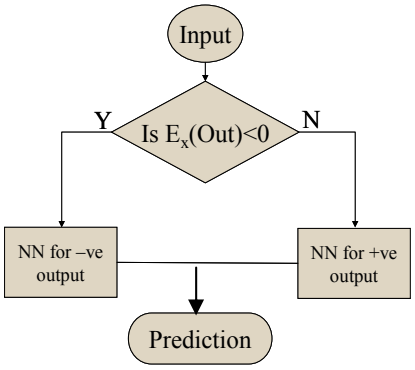


Figure 6: Flowchart for Hierarchical Method

To test this hypothesis the training data and test data were split *a priori* into positive and negative cases. Separate neural networks were then trained and tested on each set and the two results were combined to give the results shown in figure 7.

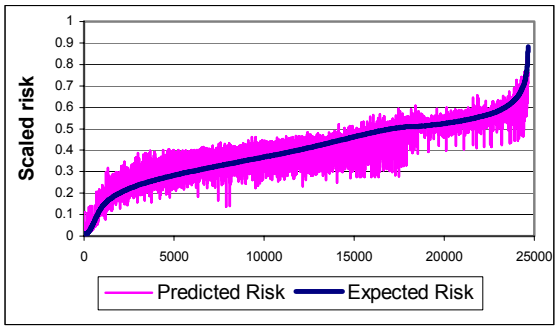


Figure 7: Hierarchical model with perfect splitting

When compared to Figure 3, the errors in most regions are seen to reduce significantly. Quantitatively the root mean square error in the positive cases reduced to 0.04, while for the negative cases this reduced to 0.07, compared to original root mean square error of around 0.1 with the best neural network architecture. The results with perfect classification at the top level were encouraging, thus it was decided to go further with this approach.

For the higher level classification tool both neural networks and decision trees (Quinlan, 1986) were tested on the training and testing data and the one with lower expected error was chosen. The decision tree gave an overall error of around 5% on the test set. However the size of the optimal decision tree was very large (2100 nodes after pruning) leading to the conclusion that the phenomenon was complex even from the classification point of view. The neural network results were very similar to the decision tree, with 5.5% testing error for single layer NN, and 5.1% testing error on double layer NN. Since the decision tree is conceptually simpler and faster to train, the decision tree was used as the first level classification tool.

In the training cycle, first the decision tree was used to split the data into expected positives and negatives, and then separate neural networks were trained on each set. The other approach was to simply split the training data *a priori* into positive and negative samples and then train separate neural networks on the split data. The former approach was seen to give better test results (RMS error = 0.08) than the latter (RMS error = 0.13). Since the decision trees are being used to classify continuous data, there is classification error at the boundary leading to negative instances going into the positive NN and vice-versa. In the *a priori* case since the neural network does not have any of the other data kinds in its training set, it gives much higher prediction errors for these outliers. Using the Decision tree to split the training data leads to some negative instances in the training set for the positive NN and some positive ones in the negative NN training set, thus the neural network are better trained to deal with classification outliers.

The results of the hierarchical model for the test data are shown in figure 8. However, due to the misclassification there are still some cases with negative risks (close to 0.5) being predicted as positive and vice-versa.

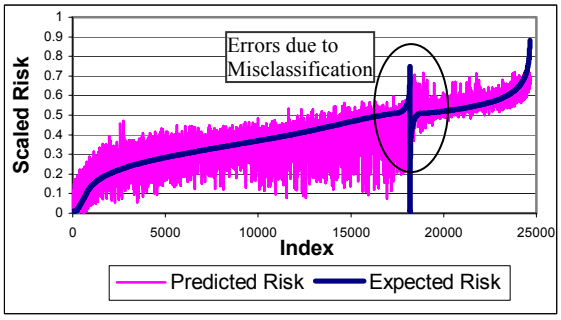


Figure 8: Hierarchical model with test data

Results from Hierarchical Model

The hierarchical approach was seen to give better accuracy all over. However outliers near the split boundary led to unreliability of predictions close to zero risks. Seeing each individual risk graphs (figure 8 and 9) these errors were more pronounced.

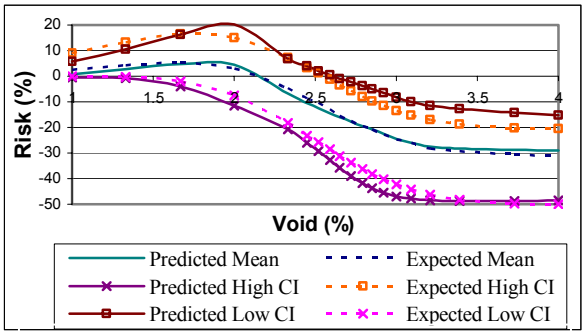


Figure 9: Hierarchical model result

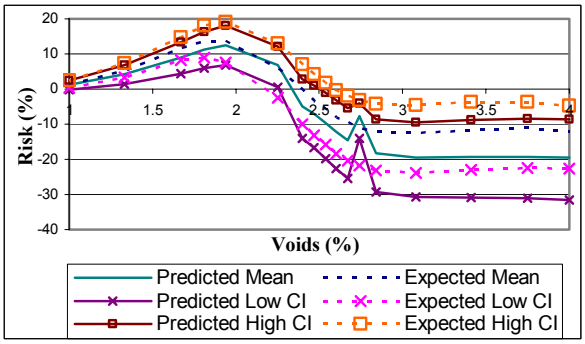


Figure 10: More hierarchical model result

Even though the predicted trends are much closer to the actual values, there are predictions that are obviously not within the acceptable error margin. On the other hand the simple neural network gave slightly worse errors, but the results were much more consistent.

Support Vector Machine Model

The third approach used for the surrogate model for risk predictions was support vector machine (SVM). Unlike neural networks, the current SVM model is limited to only one output. Since there are three outputs in the highway construction problem, we set up three independent SVMs. The three SVMs use the same inputs and independently predict the mean pay factor, lower and upper pay confidence limits.

The basic theory behind SVM regression is of mapping points from feature space to higher dimensional space by a kernel function (Burges, 1998). But instead of finding a separating hyper-plane in the higher dimensional space for SVM pattern recognition, the SVM regression tries to find a small tolerance hyper tube that can best fit the points in the higher dimensional space. An optimization algorithm is then used to minimize a loss function which compromises the smoothness and error of the hyper tube.

The code for the SVM was obtained from Ruping (2000). There are a number of parameters involved in SVM regression. Among them the most important one is the kernel function. Because there are no known guidelines to choose the kernel functions and their corresponding kernel parameters in this problem a number of trial and error runs were performed to find the best parameter combination. Once all the trial and error runs were finished, the best parameters thus obtained were applied to the large data set.

Finding the Best Kernel Function

The first batch of trial and error runs were repeated a number of times to get average trends based on a smaller data set to test the four different kernel functions namely, polynomial, neural, radial and Anova kernels. Linear kernels were not tested because regression analysis had already indicated that the data set was highly non-linear. To begin with, recommended empirical values for the kernel parameters were used. The mean square error of the three outputs and their average are shown in table 1. It is obvious that the Anova kernel has a much better result than the other three kernels, leading to the selection of the Anova kernel as the kernel function.

Table 1: MSE with different kernels

kernel	MSE1	MSE2	MSE3	Average
anova	0.004977	0.003887	0.008453	0.005773
polynomial	0.08469	0.085784	0.111027	0.093834
radial	0.01992	0.023336	0.021663	0.02164
neural	>10000	>10000	>10000	>10000

Choosing the Best Degree for Anova Kernel

Anova kernel has two parameters – the exponential degree and the multiplier gamma. First the gamma value was fixed at 0.5 and the degree was varied to see how the mean square error changed on different degrees. Figure 11 shows the results of this experiment.

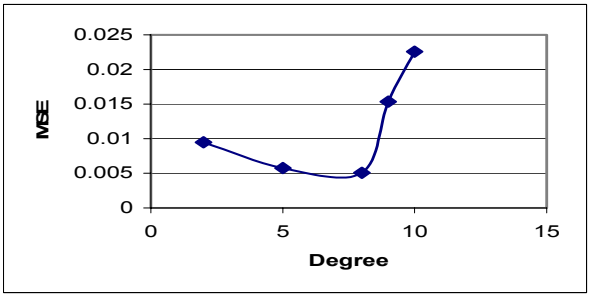


Figure 11: Anova SVM MSE at different degrees

The figure shows that the best mean square error is achieved at degree equal to seven. The error is seen to increase rapidly for degree above eight. The error also

increased slightly for degree less than five. Thus a safe range for the degree was between 5 and 8.

Choosing the Best Gamma for Anova Kernel

Based on the above results the degree was fixed at 5, and gamma was varied to see the relationship between the error and gamma as shown in figure 12.

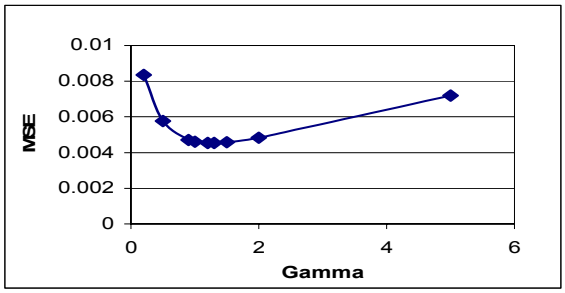


Figure 12: Anova SVM MSE at different gammas

The least mean square error was obtained at gamma equal to 1.2. When gamma was less than 1.0, the result degraded very quickly. The error increased slightly after gamma was higher than 1.5. In this case, the safe range for gamma is [1.0, 1.5].

Thus, from the trial and error tests the parameter values were set at $\gamma = 1.2, d = 6$. These were then used for the testing of the model.

Results from SVM Model

Figure 13 shows the prediction on the lower bound of payment risk on the scaled data. The lighter band shows the predicted values corresponding to the darker solid line, which represents the expected value. The data is sorted by the expected values. Clearly, the error bound is wider than neural network prediction. The average error is 23%, which is worse than neural network error. However, the SVM captures the extreme points better than neural network.

One disadvantage with using SVMs is no good guidelines are available for choosing the kernel functions and parameters. Trial and error runs are unreliable and time consuming. Here the problem was simplified by using a small training and testing data set to identify the better parameter combination by fixing one while varying the other. This method, however, can not guarantee optimal results. Firstly, the small data set may lead to parameters that result in a model with a good fit locally, but sub-optimal predictions all over. Secondly, the best value of the second parameter found by fixing the first parameter can not guarantee it is still the best if the first parameter is then changed to another value.

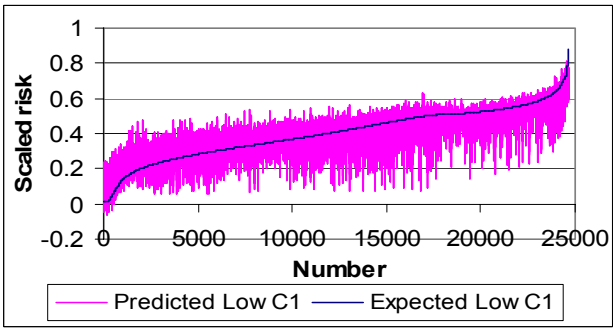


Figure 13: SVM result on the scaled data for lower bound of pay factor

Conclusions

This paper looks at some of the issues involved in developing machine-learning based models for a computationally intensive simulation model.

Performance of the various machine learning tools, was seen to be parameter dependent. For the neural network, one needs to specify the architecture, learning rates, and convergence criterion. For the SVM the kernel function and its parameters were critical. This study demonstrates the need for an established and dependable theory for parameterization of machine learning models.

Neural networks were seen to give acceptable results, capturing the general trend of the phenomena, but being unable to predict extreme events. The hierarchical approach improved the results, to some extent, but suffered due to the presence of outliers, near the classification boundary. Although for this case the SVM results are not as good as the neural networks, a better parameter combination may improve the result. Furthermore, SVM was able to capture the extreme points. This indicates that a combination of SVM and other prediction tools, such as a decision tree or neural network in a hierarchal model, may be worthwhile. This will be investigated in future research.

References

Burges, Christopher J.C., A tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery 2, 121-167, 1998.

Buttlar WG, Hausman, J., ILLISIM Program for End-Result Specification Development. *Journal of the Transportation Research Board*, 1712, 125-138, 2000.

Quinlan, J. R., Induction of decision trees. *Machine Learning*, 1:81-106, 1986 Principe, Jose C. and Euliano, Neil R. and Lefebvre, W. Curt. *Neural and Adaptive Systems: Fundamentals Through Simulations*. pg 101. John Wiley and Sons, New York, 1999.

Ruping, Stefan, mySVM – Manual, 2000

Russell, Stuart, Peter Norvig, Artificial Intelligence-A Modern Approach, Prentice Hall, 1995

Templating Optimal Orthopedic Implant Using a Decision-support System

Jing Xie, Beomjin Kim

Department of Computer Science
Indiana University – Purdue University
Fort Wayne, IN 46805 USA
jing@mchsi.com, kimb@ipfw.edu

Abstract

The field of orthopedics makes use of x-ray images to ascertain fractures and disease in bone. Historically this has involved obtaining an X-ray image of the effected bone and making visual assessment of the defect. In addition, the field has traditionally relied on such images to determine the best orthopedic implant, a process referred to in the orthopedic industry as implant templating. The process relies upon the experience of the surgeon in making best possible guess to determine the size of the implant and the positioning of the implant. Naturally such an endeavor may result in errors and corrections have to be made during surgery. Manual implant templating using traditional X-ray is labor-intensive, time-consuming processes and requires the skill of highly trained physicians or technicians, and measurement variation caused by human factors. However, with advances in computer and imaging technologies**,** digital image and related analysis is becoming commonplace in assisting diagnosis and surgery in the orthopedic industry. In this study, by taking full advantage of digital X-ray and computer technology, we have developed a semi-automated procedure to template a hip implant, by making use of an automatic edge detection method. The procedure defines the boundaries of femur stem from digital X-ray image and therefore the direction of the stem axis. The result of this semi-automated procedure shows a close agreement to results generated from traditional manual method. The developed method not only improves the measurement accuracy, but also addresses known problems in the manual method including measurement variations cased by human intervention. Beyond the accuracy that such a technique offers to clinical practice, it is expected that this semi-automated method will save time and labor for orthopedic surgeons and eventually will support their decision-making procedure.

Introduction

Digitized image with integrated computer system has become an important tool in medical field assisting diagnosis, surgeries, and therapy (Kakeda et al. 2004; Dahab et al. 2004; Wolf et al. 2003; Shrout et al. 2003; Seeberger et al. 2004). Orthopedic industry, the branch of medicine that deals with the skeleton system, its repair, disease diagnosis and implantation, is no exception. Traditionally, X-ray films have played an important role in executing a successful orthopedic surgery. Main applications of X-ray films include the preoperative determination or estimation of implant size, placement location and postoperative outcomes analysis. Over the last few years, the importance of preoperative implant templating has significantly increased because planning for surgery is now a mandatory and prudent part of the procedure for many joint replacements. In addition, the current trend for minimal invasive surgery makes sizing at the operational table more difficult and places increased reliance on accurate pre-operative implant templating. Traditional methodology of implant templating requires the surgeons to manually lay printed acetates templates of a series of prosthesis of various sizes over X-ray film. Once the optimal size and position has been established, the acetate is fixed in position with tape and used for visual reference during the operation. This traditional approach demands time and labor by highly trained orthopedic surgeons or technicians, and can easily lead to greater levels of intra- and inter-observer variability. Any errors generated from templating may increase the complexity of the operation and may potentially comprise the success of the operation.

This paper introduces a semi-automated templating method that can accurately determine the size of hip implant. The process involves matching the size and shape of an implant to the anatomical size. In particular, by locating the bone-tissue boundaries of femur stem on a digitized X-ray image with supporting edge detection algorithm, the method is able to automatically define the direction of the stem axis and consequently the implant size. This semi-automated method addresses problems in the conventional manual approach by reducing the time taken to perform the required measurements to template and eliminating the inter- and intra-observer variations. The resulting technique is fast, easy to perform, and is designed to assist orthopedic surgeons in their decision-making processes. The study compared the results of the semi-automated method to the manual method to demonstrate the accuracy of the decision-support system.

Method

The following procedures were used to select an optimal hip implant on digitized X-ray image:

Image Scaling

In conventional templating, the operating surgeon has to deal with two different magnification scales. The first is on

the x-ray film and the second on the various acetate images representing implants at some magnification. Based on the imaging environment and the anatomical location, the X-ray film is generated either in compressed or magnified mode. The conversion of the X-ray image on the film to the actual size of the anatomy is indispensable preprocessing for digital templating. The scaling factor (magnification /compression) on the X-ray film is obtained either by placing a scale marker on the side of the anatomical joint when taking X-ray image or using the scale on the X-ray plate. The magnification scale on the series of acetate images are usually defined by manufacture. Usually these two magnification scales are not identical. In the traditional method, surgeons do this adjustment manually, which is both time-consuming and labor-intensive. In the developed digitized templating method, by executing this procedure in automated fashion, the system addresses these limitations.

After resolving the scaling conversion, the image is digitized and displayed on the output device. With the aid of custom software, the user interactively measures several factors on the computer terminal to find an optimal template. The digital system requires an additional scale translation to convert on screen pixel-based measurement to actual size of templates. The above two conversion tasks were expressed in the following formula;

$$M_{mm} = \alpha * M_{pixel} / I_{resolution}$$

$$\alpha = C / S$$

where α is a conversion factor, C is the coefficient to transform pixel magnitude to millimeter units, and S is the scaling factor to transform any compressed/magnified X-ray image to original dimension. $I_{resolution}$ is the image resolution of X-ray, M_{pixel} is the measurements in pixel on image, and M_{mm} is the "true" measurements in millimeter. This magnitude conversion is applied to both the horizontal and vertical dimensions.

Computing the Location of Stem Axis

The thighbone or femur is the main component of a hip. Determining the exact location of the femur is essential in templating for the correct implant. This location determines not only the size of the femur implant but also the positions of other implantable hardware such as the acetabular cup. The illustration in Figure 1 shows the centerline of a femur stem, the neck and the acetabular cup. Without going in detail, it is apparent that if the centerline (Line AB, Figure 1) of the stem axis is misplaced, the femur implant would be wrongly judged and the associated acetabular shell, centered at point C in Figure 1 would be misplaced relative to the anatomy.

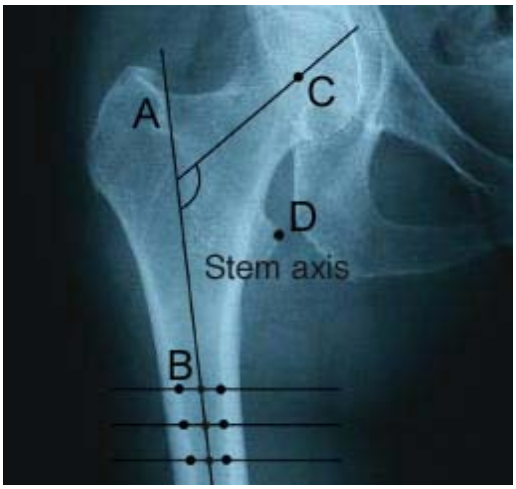


Figure 1. Illustration of hip femur stem axis on X-ray.

The location of the femur stem is defined by the stem axis, which is the geometrical center of the stem. In the traditional method the user will need to identify multiple reference points on the X-ray film to define the stem axis. This approach increases the complexity and inevitably to significant levels of inter- and intra-observer variability for the final measurements. The present semi-automated system has developed a technique that computes the stem axis in automated fashion.

Due to the nature of acetate and therefore digitized X-ray images, hard material, such as bone, appears as higher intensity color, and softer material, such as tissue, appears as lower intensity color. An abrupt change in intensity on the X-ray image indicates the boundary between bone and tissue. The stem axis can be considered as the centerline of two femur stem edges that will be passing through the locations, which are at the equidistance from the two stem edges. Therefore, one of the main tasks to tackle was to automatically detect the edges of the stem.

The nature of X-ray image including the characteristic low resolution and the ubiquitous noise made it hard to identify the stem edges. An example of edges on a digitized film is presented in Figure 2, where the image intensity is plotted against pixel location. The edge should be detected at locations where the image intensity shows ramps corresponding to the edges of the femur. However, as shown in Figure 2, extra minor peaks appear on the "Before" profile due to the inherent noise of the acetate and thus digitized image. To eliminate or reduce the influence of noise in identifying the "true" edges, a smoothing operation was applied to the original digitized image before applying edge detection algorithm. Gaussian smooth filter was used to smooth the image and eliminate noise so that only main features were retained (Sander et al.2000; Gonzalez 1992).

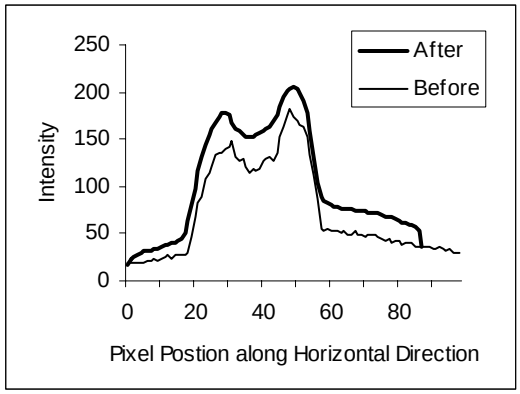


Figure 2. Change in pixel intensity on an arbitrary horizontal line before and after applying Gaussian smoothing filter.

Generally, for a noise-affected image even prior to edge detection, smoothing by Gaussian filter is the first mathematical operation. As the algorithm requires edge detection in one dimensional (1-D), 1-D Gaussian filter of Equation 1 was used:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-x^2/2\sigma^2} \quad \text{Eq. 1}$$

where σ is the standard deviation of the distribution and x is the pixel position in 1-D. σ affects the size of the Gaussian filter. Increasing Gaussian filter size results in smoother and yet more blury image.

The effect of Gaussian smoothing has a similar fashion as the mean filter function (Boyle and Thomas 1988; Davies 1990; Vernon 1991; Gonzalez 1992; Haralick and Shapiro 1992; Horn 1986). The degree of smoothing is determined by the standard deviation of the Gaussian equation. Gaussian yields a `weighted average' of each pixel's neighborhood, with the average weighted more towards the value of the central pixels, as opposed to the mean filter's uniformly weighted average. Hence, Gaussian provides gentler smoothing and preserves edges better than a similarly sized mean filter.

The "After" profile in Figure 2 shows the results after applying Gaussian filter to the "Before" profile. It is evident that the profile resulting from the Gaussian operation is smoother. A study has reported that while the smoothing operation reduces noise and minor edges, it may cause a distortion of the edge locations (Yitzhaky 2003). The severity of that distortion depends mainly on the size of the smoothing operator, and on the spatial shapes of the edges (Yitzhaky 2003). This artifact was not seen in the results of this application.

In Figure 2, the profile could be represented as a continuous function of intensity represented as $f(x,y)$, which could further be simplified to $f(x)$ in 1-D. The first derivative of $f(x)$ generates a local maximum at an edge. Depending on the orientation of the gradient, the maximum can be in either positive or negative. First derivative was applied to the Gaussian smoothed digitized X-ray image and locations of edges in 1-D were identified as the maximum and minimum in the resulting image. Figure 3 shows the first derivative in 1-D of the curve in Figure 2.

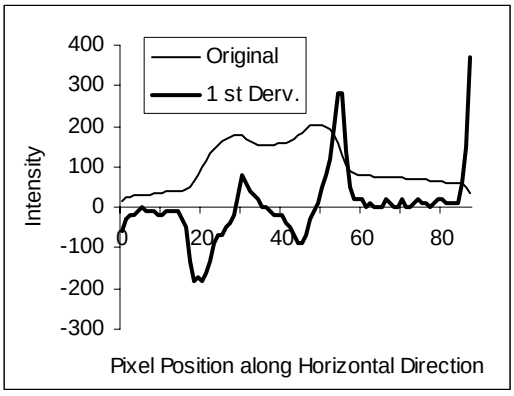


Figure 3. First derivatives of the X-ray.

It was noticed that in Figure 3, there were multiple peaks after applying for first derivative. Some peaks were due to remaining noises on the digitized X-ray, in spite applying the Gaussian smoothing operation. To further eliminate interference associated with this remaining noise in accurately identifying real peaks, the threshold level was set to high absolute value to filter out spurious peaks associated with noise. The selection of a threshold value is an important design decision that depends on a number of factors, such as image brightness, contrast, level of noise, and edge direction (Gonzalez 1992). To recover the edges, the gradient image must be segmented using a global or local (*i.e.*, adaptive) threshold operator. The choice of a threshold value determines the resulting segmentation and, therefore, the perceived quality of the edge detector. It is useful to consider the cumulative histogram of the gradient image in selecting an appropriate threshold value. In a typical approach, the top 10 to 20 percent of the largest gradient values were selected as edge points (Gonzalez 1992; Haralick and Shapiro 1992). Once peaks caused by noises were filtered out, peaks correspond to edges could be easily identified. Hence, pixels corresponding to edges would then be identified.

It should be pointed out that in this application, the edge was detected only in 1-D along X direction (horizontal); therefore the edge detection algorithm could be simplified so that first derivative of the image only applies in 1-D. However, the algorithm of edge detection in two dimensions can be implemented in the same way for possible expansion of the application to any spatial orientation of edges.

After edge points at a particular vertical position (Y) were determined, the rest of edge points along Y direction were identified by scanning through the image vertically using the same edge detection algorithm. At any given Y

value (vertical position), the mid-point between edge points was then calculated. As theoretically stem axis consists of mid-points of each pair of edge points, the liner equation representing stem axis was determined and optimized using least square method based on the calculated midpoints.

Once the stem axis was determined, in conjunction with a single user-defined reference point at lesser trochanter (point D in Figure 1) of the hip joint, the program measures stem width at 80 millimeters below the lesser trochanter along the stem axis. With additional user defined points on the digital X–ray, neck length and angle can be subsequently determined. These measurements were used to compile the size of the required implant.

Templating

Once the required size of the implant was obtained, the system compared the measured size to a series of implants of various sizes. The implant size closest to the measured size was determined the best match. In case there were more than one best matching sizes, a situation that may arise when the algorithm generated size lies between two adjacent templates, the user was informed by the system to choose a size based on the user's experiences and judgment. Once the user chooses the desired size of the implant, the template of the best matching implant size was superimposed on top of the X-ray image. The system then allows the user to manipulate the position of the template interactively to further confirm the match. The manipulation includes rotation and shift in two dimensions of the template on top of the digitized X-ray.

Supporting System

The developed system allows information related to X-rays image including patient identification, acquisition date, and prosthesis to be stored in a database for the ease of retrieval. Illustrations of prosthesis templates created in CAD application were reformatted as sets of data points (.dat file) using a supporting software program. Information on implant including size was also stored in a database.

The application also has the capabilities of radiographic analysis on X-rays of prosthesis. Surgeons can indicate radiolucency and osteolysis, a kind of bone disease, directly on the X-rays and save the information for future use.

Multi-level Viewing

X-rays only give surgeons one level of view. Sometime due to spatial size constraint, surgeons cannot have detailed view at certain areas. In the present system, zoom-in feature provide multiple levels of detailed views with varying magnification factors applied to the X-rays. This feature is designed to enable surgeons to identify certain tiny features and perform more accurate measurements on the X-ray. In addition, by generating fisheye view, zoom-in allows surgeon to view the same detail on both original and magnified images at same time and easily correlate measurements between the magnified and the original images. A series of geometric transformation were utilized to accomplish this correlation.

Experiment

In order to investigate the accuracy and efficiency of the developed system, an experiment was conducted using the assistance of a technician experienced in the traditional approach to determine implant. The results by traditional approach were compared to the results produced by our semi-automated application. The same technician also performed templating using the semi-automated system. The experiment recorded implant size to be used and time taken by both methods. A window-based custom software was developed based on algorithms discussed earlier and executed on a PC that a 2.4 GHz computing speed with 256MB main memory.

Randomly selected X-rays of unidentified patients were used for templating for both techniques. Digital X-rays were obtained by scanning original acetate X-rays (VIDAR SIERRA Plus Digitizer).

Results

For each X-ray sample, best matching implant sizes determined by both techniques were recorded. The difference between the two sizes were calculated and shown in Figure 4. It is evident that the computer system yields very close results to those obtained by the traditional method in all ten studies. The difference, if any, is also within the error of clinically acceptable range (± one size) obtained by traditional templating method (horizontal lines across 1 and –1 in Figure 4).

In addition to accuracy, the study also qualitatively demonstrated that the average time taken for templating using the semi-automated developed system was much less than using the traditional method.

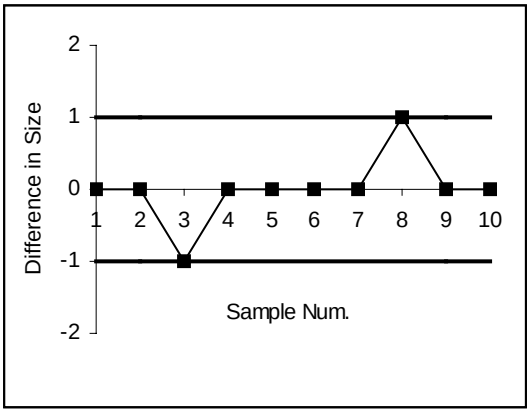


Figure 4. Results of comparative experiment between traditional and semi-automatic approaches.

Discussion

In this preliminary study, the decision support system showed promising results where the selected implant using this method was within acceptable range to results from traditional method. Combined with the smoothing operation, edge detection algorithm objectively identified the stem boundaries solely based on pixel intensity, which eliminates inadvertent intrusion of human errors and variances. In addition, compared to traditional templating method, it also saves time and labor of highly trained person. Other supporting features such as multi-level viewing allow surgeon to accurately identify reference points for size determination. However, further tests are needed on usability, inter- and intra- variance with larger sample population including multiple users and digitized images.

As another investigation, Gradient and Laplacian methods to enhance edge detection were investigated. As mentioned earlier, when an edge is present, a gradient is apparent on the pixel intensity of the X-ray image. At this point, the first derivative achieves a maximum and the second derivative becomes zero. Notice that in Figure 2, it is difficult to determine the exact pixel corresponding to the edge of stem based on the gradient. There are many ways to enhance the signal to detect edge. The most popular techniques may be grouped into two categories, Gradient and Laplacian. Gradient method utilizes the first derivative. The Laplacian method, on the other hand, searches for zero-crossings in the second derivative of the image to locate edges. Figure 5 shows the result after applying Laplacian method to the results from Figure 2. The comparison between Gradient and Laplacian methods reveals that Gradient method seems to yield more clear correlation between peak and edge than Laplacian, as shown in Figure 5. The Laplacian method is more affected by the presence of noise. Therefore Gradient method was employed in edge detection in our study.

It should be pointed out that the quality of X-rays is crucial for accurate edge detection. In addition, due to various shapes of different prostheses, spatial orientations of edges vary. Any irregular shape of the edge will increase the complexity of edge detection. Although the current approach is in semi-automatic way, it could be extended to automated fashion by employing more sophisticated noise removal and image analysis algorithms.

Finally, the developed method can be applicable to other body joints such as knee, shoulder and elbow with small modification, which will be another task in the future.

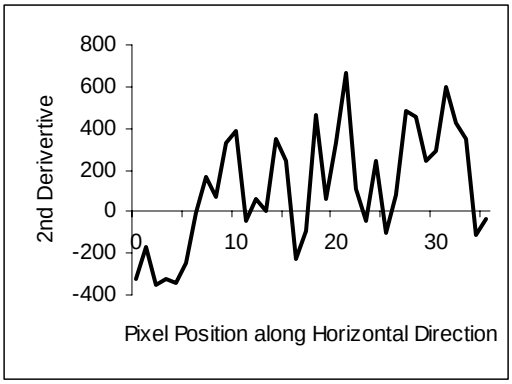


Figure 5. Results of Laplacian.

Conclusion

This paper has presented a semi-automatic, decision support method of determining an implant by making use of an edge detection algorithm. Compared to traditional templating method, it not only saves time and labor of highly trained person, but also reduces human errors and possibly inter- and intra variances. Overall, this method can become effective tools helping surgeons in surgery planning and decision making processes.

References

Kakeda, S., Moriya, J., Sato, H., Aoki, T., Watanabe, H., Nakata, H., Oda, N., Katsuragawa, S., Yamamoto, K., and Doi, K. 2004. Improved Detection of Lung Nodules on Chest Radiographs Using a Commercial Computer-Aided Diagnosis System. *AJR Am J Roentgenol*, Feb; 182(2): 505-10.

Dahab, G.M., Kheriza, M.M., El-Beltagi, H.M., Fouda, A.M., and El-Din, O.A. 2004. Digital quantification of fibrosis in liver biopsy sections: Description of a new method by Photoshop software. *J Gastroenterol Hepatol*, Jan; 19(1): 78-85.

Wolf, G., Nicoletti, R., Schultes, G., Schwarz, T., Schaffler, G., and Aigner, R.M. 2003. Preoperative image fusion of fluoro-2-deoxy-D-glucose-positron emission tomography and computed tomography data sets in oral maxillofacial carcinoma: potential clinical value. *J Comput Assist Tomogr*, Nov-Dec; 27(6): 889-95.

Shrout, M.K., Jett, S., Mailhot, J.M., Potter, B.J., Borke, J.L., and Hildebolt, C.F. 2003. Digital image analysis of cadaver mandibular trabecular bone patterns. *J Periodonto*, Sep; 74(9): 1342-7.

Seeberger, T.M., Matsumoto, Y., Alizadeh, A., Fitzgerald, P.G., and Clark, J.I. 2004. Digital image capture and quantification of subtle lens opacities in rodents. *J Biomed Opt*, Jan; 9(1): 116-20.

Sander, P., Gu, X., Gortler, S., Hoppe, H., and Snyder, J. 2000. *Silhouette clipping*. In Proceedings of the 27th annual conference on Computer graphics and interactive techniques. 327-334.

Gonzalez, R., and Woods, R. 1992. *Digital Image Processing*. Addison-Wesley Publishing Company.

Boyle, R., and Thomas, R. 1988. *Computer Vision: A First Course*. Blackwell Scientific Publications.

Davies, E. 1990. *Machine Vision: Theory, Algorithms and Practicalities*. Academic Press.

Vernon, D. 1991. *Machine Vision*. Prentice-Hall.

Haralick, R., and Shapiro, L. 1992. *Computer and Robot Vision.* Addison-Wesley Publishing Company.

Horn, R**.** 1986. *Robot Vision.* MIT Press.

Yitzhaky, Y. 2003. A Method for Objective Edge Detection Evaluation and Detector Parameter Selection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Oct; 25(10): 1027-33.

Intra and Extra-Generation Schemes for Combining Crossover Operators

Dana Vrajitoru
Intelligent Systems Laboratory,
IUSB, Computer and Information Sciences,
1700 Mishawaka Ave, P.O. Box 7111
South Bend, IN 46634
danav@cs.iusb.edu

Abstract

Several studies on the variations of the crossover operator have shown that each of them presents speci£c properties that are interesting under particular circumstances. The advantages of each operator over others are often contradictory and the best operator depends on the problem being solved. This paper is based on the assumption that a combination of several crossover operators can take advantage of their respective qualities and power. Thus, we propose several combination models based on four crossover operators: the 1-point, 2-point, uniform and dissociated. We test the performance of these models through an experimental approach using the problem of the Hamiltonian circuit.

Introduction

The crossover operator is one of the most important and powerful features of the genetic algorithms (GAs), and it has been the object of interest for many researchers in the £eld (Mao *et al.* 2002; Rana 1999; Suzuki & Iwasa 1999).

Holland (1975) and Goldberg (1989) have started the study of the probabilistic properties of the crossover through the schemata theorem. An interesting question that has been risen is between crossover and mutation, which one is more important for GAs (Mühlenbein & Schlierkamp-Voosen 1995; Wu, Lindsay, & Riolo 1997). The general conclusion is that mutation alone is not suf£cient for many problems, and that the role of crossover is essential.

In a different direction, Booker (1987) and Spears (1990) have used a second measure to evaluate the genetic operators : the exploratory power. They have shown that a highly disruptive operator is also highly productive. As the exploiting and the exploratory qualities are contradictory, each of these features could become an advantage for some particular search £elds, and a disadvantage for others.

Some research has been made to combine several crossover operators in the same application of the GA (Hong, Kahng, & Moon 1995; Spears 1995), and our paper follows similar ideas.

Generally, the number of options available for this operator is quite large. We can cite the nonuniform crossover (Maini *et al.* 1994), the spontaneous crossover (Mayer 1998), the selective crossover (Vekaria & Clack 1998). Some of these crossover operators can show a better performance on speci£c £tness landscapes, and be less able to solve other problems.

Considering these issues, it is dif£cult to choose one operator without prior experimenting for a given problem. The classical approach is to test several forms on a limited number of cases, and to base the future use of the crossover on them. The results obtained on a class of problems are in general hard to extend to another class, as the shape of the search space can be completely different. Thus, the choice of the form of crossover is often a subjective one.

We are interested in the situation where a single crossover operator must give good results for most of the problems to study, which is often the case. We would like to obtain a recombination model that shows a better performance than any single crossover on all of the problems we consider, and not only on some of them.

To achieve this goal, we present several models using not one, but four crossover operators (1-point, 2-point, uniform and dissociated) and combining them during the execution of the GA. We expect at least some of these models to show a better performance than each of the considered operators on the test problems. We are testing these models on the problem of the Hamiltonian circuit.

The next section introduces the general parameter settings, the test functions, and the four basic types of crossover that we have used for our experiment. The following section de£nes and tests several combination models based on these operators. The last section summarizes the entire experiment.

Experiment Description

This section presents the test functions and the parameter settings that we have used for the experiment.

Standard Functions Set This set contains ten functions that are used by many researchers to test GAs and their precise equations can be found in (Whitley *et al.* 1996). Each if them is a real function depending on 2 to 30 variables on a given interval. The goal is to minimize each of these functions, so lower £tness values are actually better ones. The genetic representation is a concatenation of the variables occurring in the functions. In our case we have use a discretization of the variables using 10 binary genes for each of them.

The Hamiltonian Circuit Problem The second test function that we have chosen is the problem of the Hamiltonian circuit.

Hamiltonian circuit (HC): Given an oriented graph, £nd a circuit that passes once and only once through each vertex. This problem is known to be NP-complete (Brassard & Bratley 1994).

We have performed our experiments with 11 HC problems with graphs having from 50 to 150 vertices and from 246 to 3136 edges. The direct representation of a HC problem for the GAs is dif£cult. De Jong and Spears (1989) have implemented a genetic representation using the transformation of an instance of HC into an instance of SAT, which is easier to represent in a genetic form.

A detailed description of the reduction of a HC instance into a SAT instance can be found in (Brassard & Bratley 1994) or (Vrajitoru 1999). For any given graph, a Boolean variable corresponds to each arch, and is given the true value if the arch belongs to the circuit. The SAT expression summarizes the fact that, for each vertex, one and only one of the entering edges and of the exiting edges must belong to the circuit.

For example, let us consider the graph in Figure 1. The conversion of the HC instance for this graph into a SAT instance would result in Equation 1, in which a Boolean variable with the same name as edge is true if the edge belongs to the Hamiltonian circuit. The symbols '$\otimes$' and '$\wedge$' represent the Boolean 'xor' and 'and' operators.

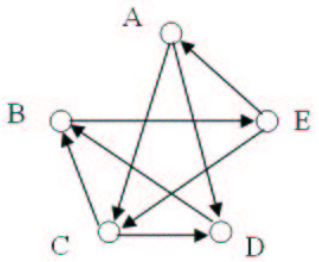


Figure 1: The graph for which Equation 1 represents the instance of SAT corresponding to its HC instance

$$\begin{array}{ll} out & (AC \otimes AD) \wedge BE \wedge (CB \otimes CD) \wedge \\ & \wedge DB \wedge (EA \otimes EC) \wedge \\ in & EA \wedge (CB \otimes DB) \wedge (AC \otimes EC) \wedge \\ & \wedge (AD \otimes AD) \wedge BE \end{array} \quad (1)$$

The £tness function for the HC problems is based on a fuzzy logic interpretation of the logical expression derived from the graph. Thus, each individual represents an assignment of truth values to each of the variables composing the logical expression. The logical and operator is interpreted as the average value of the operands, while the logical or operator is interpreted as the maximum of the values of the operands. If an individual represents a perfect Hamiltonian circuit, then its £tness will be 1.

The questions that we ask based on this problem are the following. First, can the combination of several crossover operators perform better than each of them individually? And second, how does the algorithm react to the problem scaling, especially when going from small problems to big problems?

Parameter Settings

Since we are studying the crossover operator, we have performed 50 runs of the GA for each problem with a crossover rate of 1 and a mutation rate of 0.0005. Previous tests have determined that this is a good setting for both our test problems. For each of the 50 trials, we generate one initial population and run the GA starting from it separately for each operator or combination model. This way, each of the operators has the same chance to £nd the optimal individual.

Each generation contains 100 individuals and the number of generations is limited to 1000. We have used the £tness proportionate selection (Goldberg 1989), and the monotone reproduction (Vrajitoru 1999). Our performance measure is the average of the best £tness value achieved in the last generation.

Crossover Operators

Among the existing variations and forms, we have chosen four crossover operators of comparable behavior: the 1-point, 2-point, uniform with a swap probability of 0.5, and dissociated crossover.

The 1-point crossover has little exploratory power, but can better exploit the knowledge contained in the initial population. On the contrary, the n-point crossover ($n >> 1$) is highly exploratory and is recommended when the population size is small. The uniform crossover has a great exploratory power, which can be controlled by the value of the swap probability. This operator has shown good performance even with a high exchange rate (Syswerda 1989).

The *dissociated* crossover has been introduced in (Vrajitoru 1999). We have used a variation on its original form that splits each parent in two at a different cross site, and swaps the resulting right hand sides of the parents by applying the logical 'or' operator on the overlapping portions for one child, and logical 'and' operator for the other child. More precisely, if $par_{1,2}$ are the parents in the crossover operation, $csite_{1,2}$ are two cross sites, then the children individuals are de£ned by the following equation in which the symbols $\vee$ and $\wedge$ represent the logical operators "or" and "and" respectively:

$$child_1(i) = \begin{cases} par_1(i), & i \leq csite_1 \\ par_1(i) \vee par_2(i), & csite_1 < i \leq csite_2 \\ par_2(i), & csite_2 < i \end{cases}$$

$$child_2(i) = \begin{cases} par_2(i), & i \leq csite_1 \\ par_1(i) \wedge par_2(i), & csite_1 < i \leq csite_2 \\ par_1(i), & csite_2 < i \end{cases} \quad (2)$$

Figure 2 illustrates the way the dissociated crossover builds the children, as speci£ed in Equation 2. This formulation is different from the original one presented in (Vrajitoru 1999) concerning the operator applied to the overlap-

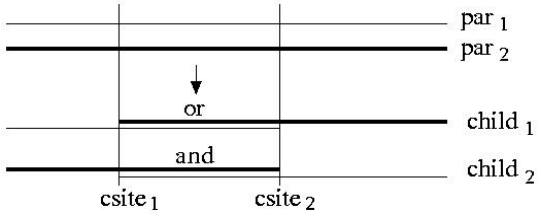


Figure 2: The dissociated crossover

ping part of the parents for the second child. We have experimented with various formulations and the 'and' operator seems to be the best one.

Before starting to combine the crossover operators, it is interesting to know how well each of them performs on the problems we have chosen. Tables 1 and 2 present the average over the 50 trials of the best performance in the last generation for each operator for the set of standard functions and for the HC problem respectively. The highest possible £tness in our case is 1, which would be achieved in the case of the optimal individual. The £rst column represents the graph itself, labeled by the number of vertices. The last column indicates the operator presenting the best performance for that graph.

Table 1: Average £tness of the crossover operators on the standard functions set (minimization problems)

# F	1-point	2-point	Uniform	Dissociated	Best
F1	0.066	0.044	0.344	0.383	uniform
F2	0.414	0.516	0.840	0.734	1-point
F3	12.22	12.34	12.12	10.44	dissoc.
F4	0.981	0.788	1.104	1.526	2-points
F5	5.206	8.175	4.596	5.949	uniform
F6	2.414	2.615	3.916	3.483	1-point
F7	608.37	614.82	670.62	708.84	1-point
F8	1.707	1.501	2.978	3.941	2-points
F9	0.197	0.184	0.243	0.196	2-points
F10	1.887	1.687	2.188	2.527	2-points

From these tables we see that the dissociated crossover presents the best performance on more than half of the problems, especially for the large graphs, and is followed by the 2-point crossover. However, the difference between the performance of these operators is not always visible because we only show the £rst 3 digits of the average £tness. Given the diversity in the results, we can deduce that, in general, we cannot predict with great accuracy that one of the operators will perform better than the others on a new problem.

Combining Crossover Operators

In this section we present several combination models for crossover operators and compare them with single crossover operators.

Table 2: Average £tness the crossover operators on the HC problem (£tness maximum of 1)

Graph	1-point	2-point	Uniform	Dissociated	Best
HC50	0.977	0.979	0.978	0.979	dissoc.
HC60	0.981	0.982	0.984	0.985	dissoc.
HC70	0.977	0.980	0.983	0.986	dissoc.
HC80	0.968	0.972	0.981	0.981	dissoc.
HC90	0.953	0.960	0.973	0.976	dissoc.
HC100	0.936	0.944	0.964	0.973	dissoc.
HC110	0.911	0.923	0.950	0.971	dissoc.
HC120	0.889	0.902	0.936	0.971	dissoc.
HC130	0.873	0.887	0.924	0.969	dissoc.
HC140	0.854	0.867	0.911	0.970	dissoc.
HC150	0.793	0.808	0.859	0.973	dissoc.

The attempts to combine crossover operators have been far less numerous than the developed forms of crossover. Schaffer and Morishima (1987) proposed a crossover scheme based on the n-point crossover where the number and position of the crossover sites where self-adapting over the GA run. Following the adaptation idea, Spears (1995) has developed a model combining the 2-point and uniform crossover, where the proportion between the two is also evolved by the GA. Finally, Hong et al. (1995) propose several adaptive combination strategies based on traditional, cycle, and geographic crossover. In all the cases, using more than one crossover seems to outperform each single crossover.

The three cited approaches use strategies that imply the use of several operators within the construction of each generation. We propose three new models, the £rst two applying several forms of crossover to build every new generation (*intra-generation* models). In the third model, a single crossover is used in each generation, and this operator changes after several generations (*extra-generation* model).

Model Description

Intra-generation schemes. The £rst combination model we propose aims to use several forms of crossover for building each generation. In general, for each crossover operation, the method chooses one of the four operators presented in the previous section by a random function following two schemes.

The £rst model, called *balanced combination*, gives each operator a probability to be selected of 0.25, which means that each operator has equal chances to be employed.

The second model we propose, named *adaptive*, starts the exact same way as the balanced one, by giving each of the four operators an equal chance to be selected. The probabilities associated with the basic operators are modi£ed after completing each generation the following way:

- When the population is too heterogeneous, or when the change in the £tness value is too dramatic from one generation to the next, we increase the probability associated with the conservative operators, which are the 1-point and the 2-point crossover.
- When the population is too homogeneous, or when the change in £tness from the previous generation is too small, we increase the probability associated with the more exploratory operators, which are the uniform and the dissociated crossover.
- Otherwise we may modify all of the probabilities by a small random amount.

Extra-generation Schemes. The third combination model uses a single crossover operator during a certain number of generations, then switches to another. The passage from one operator to the next is accomplished in a round-robin fashion, with the step equal to 50 or 100 generations. We have denoted these strategies by *RR50* and *RR100*.

For the moment, all the experiments start with the 1-point operator, followed in order by the 2-point, then by the uniform, and £nally by the dissociated crossover, and restarting the cycle if needed. The experiment can be extended to see whether the order and the starting point in the cycle are of any importance.

Experimental Results

We still expect the best crossover operator for each problem to be better that the combined models. In the situation we considered, the crossover form has to perform well on several different problems, and not only on a particular one. Thus, we are interested in comparing the combined models with the crossover operator that has shown the best general performance (see Table 2) for each individual problem. The last column marks the combination scheme that has presented the best performance so that it can be compared to the best single operator.

From this table, we can notice that the balanced combination method shows a better performance than the best single operator in most of the cases, except for the last 2 problems where the difference between them is very small. We believe that this scheme proves to be better than the other in most cases because it is the one taking advantage of the strength of each operator for every generation. When the population becomes more homogeneous in the later generations, the probability to spawn a better individual with this method is about equal to the maximum probability considering each operator separately. There is no signi£cant difference between the other combining methods, although they are in general better than most individual operators.

Next, we have thought interesting to observe the evolution of the best £tness through generations. Thus, Figure 3 shows the average performance for the 2-point, dissociated, combined balanced, and combined adaptive operators during the £rst 500 generations for the HC50 problem.

Table 3: Average £tness of the combined models, standard functions set (minimization problems)

# F	Balanced	Adaptive	RR5	RR10	Best
F1	0.066	0.044	0.024	0.025	2-point
F2	0.523	0.361	0.473	0.393	2-point
F3	10.4	11.74	10.6	10.92	dissoc.
F4	0.555	0.272	0.538	0.373	dissoc.
F5	1.581	2.409	2.39	2.985	balanced
F6	1.978	1.374	2.133	1.546	adaptive
F7	607.547	504.569	537.98	521.3	adaptive
F8	1.119	0.664	0.808	0.634	2-point
F9	0.165	0.139	0.154	0.156	2-point
F10	1.453	1.062	1.438	1.269	2-point

Table 4: Average £tness of the combined models, HC problems (£tness maximum of 1)

Graph	Balanced	Adaptive	RR5	RR10	Best
HC50	0.981	0.980	0.980	0.980	balanced
HC60	0.987	0.984	0.987	0.986	RR5
HC70	0.990	0.986	0.990	0.988	RR5
HC80	0.990	0.983	0.989	0.990	balanced
HC90	0.989	0.974	0.988	0.986	balanced
HC100	0.985	0.962	0.985	0.983	balanced
HC110	0.980	0.944	0.980	0.979	RR5
HC120	0.978	0.927	0.978	0.976	RR5
HC130	0.976	0.915	0.976	0.975	RR5
HC140	0.974	0.897	0.975	0.974	RR5
HC150	0.975	0.839	0.975	0.973	dissoc.

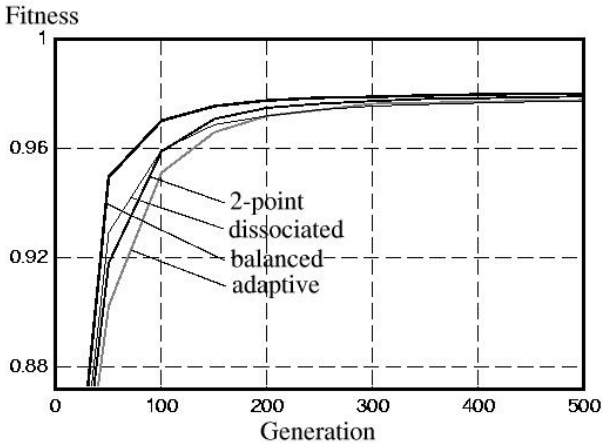


Figure 3: Average performance in 500 generations, HC50 graph

Table 5: Best £tness of 50 runs for the best combined method and the best single operator, HC problems (£tness maximum of 1)

Graph	Best single operator	Best combined method
HC50	0.995 (2-point)	0.991 (adaptive)
HC60	0.995 (dissoc.)	0.995 (RR10)
HC70	0.996 (dissoc.)	0.996 (RR5)
HC80	0.994 (uniform)	0.996 (balanced)
HC90	0.991 (uniform)	0.993 (RR5)
HC100	0.987 (uniform)	0.990 (RR5)
HC110	0.982 (dissoc.)	0.985 (balanced)
HC120	0.979 (dissoc.)	0.982 (RR5)
HC130	0.978 (dissoc.)	0.981 (RR5)
HC140	0.978 (dissoc.)	0.979 (RR5)
HC150	0.978 (dissoc.)	0.979 (RR5)

The Quality of the Solutions

For a different view of the quality of the solutions achieved by each method, we have searched for the best solution obtained in any of the 50 trials. Table 5 show the best solution found by a single crossover operator and by the best combined method for each problem. We can notice that the best solution found by the balanced scheme is in general better than the best solution obtained by any single operator, no matter which one it is.

The last experiment we have performed was intended to study the in¤uence of the mutation rate on the quality of the results for the various crossover operator and combining methods. Thus, we have plotted the average performance (£tness of the best individual) of the best single operator (dissociated), the best combined method (balanced), and the uniform crossover for a set of experiments without mutation and another set with a high mutation rate.

Figures 4 and 5 show these new results in 1000 generations with a mutation rate of 0.0 and of 0.01. The x axis represents the number of vertices in the graph. The size of the individual actually depends on the number of edges, which is signi£cantly larger than the number of vertices.

It is interesting to notice that the difference between the performance of the dissociated crossover and the other single operators increases with the problem size. In these experiments, the dissociated crossover is showing a better performance than the others and the difference increases with the graph size. Still, the performance of the combined balanced model is much closer to the dissociated crossover than the uniform operator.

The three models show a better performance without mutation rather than with a mutation rate of 0.01, which suggests that this mutation rate is too high for this class of problems. The mutation rate of 0.0005 that we have used before is a much better choice.

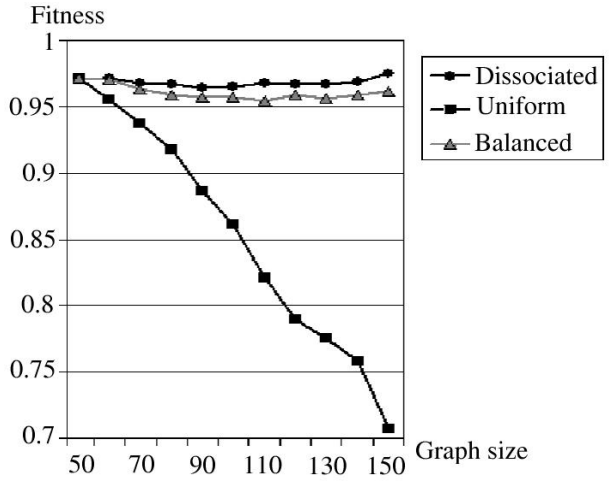


Figure 4: Average performance on HC problems without mutation

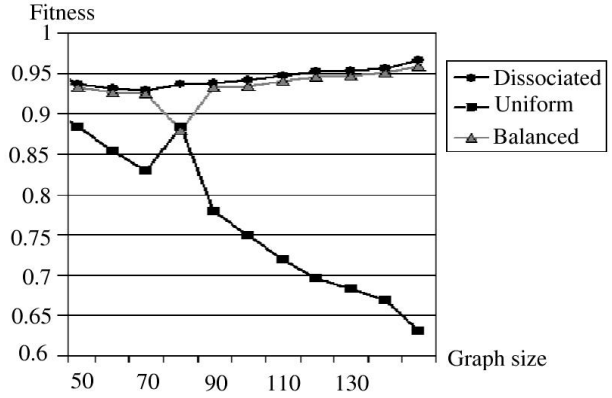


Figure 5: Average performance on HC problems with a mutation rate of 0.01

Conclusions

This paper has presented several combination methods for the crossover operator in an attempt to take advantage of the strength of each individual operator and produce better solutions to the problem in a more consistent way.

In the second section we have compared the four basic crossover operators, which are the 1-point, 2-point, uniform, and dissociated. The £rst experiments presented in Table 2 have shown that the best operator can be different from one problem to another.

In the third section we have de£ned several models combining the four crossover operators and we have tested their performance against the best single operator using the same test problems. The results from Table 4 show that a combined model can be consistently better than any of the single crossover operators. Further tests have con£rmed that the combined balanced model, that chooses between the four operators in a random fashion with equal probabilities for each of them, is also capable of generating solutions that are closer to the optimal one.

We can conclude that a model combining the power of several crossover operators is a more robust choice and can £nd better solutions under various circumstances..

References

Booker, L. 1987. Improving search in genetic algorithms. In Davis, L., ed., *Genetic Algorithms and Simulated Annealing*, 61–73. Morgan Kaufmann Publishers.

Brassard, G., and Bratley, P. 1994. *Fundamentals of Algorithmics*. Prentice-Hall.

De Jong, K., and Spears, M. 1989. Using genetic algorithms to solve NP-complete problems. In *Proceedings of the International Conference on Genetic Algorithms*, 124–132. Fairfax (VA): George Mason University.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading (MA): Addison-Wesley.

Holland, J. H. 1975. *Adaptation in Natural and Arti£cial Systems*. Ann Arbor: University of Michigan Press.

Hong, I.; Kahng, A.; and Moon, B. 1995. Exploiting synergies of multiple crossovers: initial studies. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, 245–250.

Maini, H.; Mehrotra, K.; Mohan, C.; and Ranka, S. 1994. Knowledge-based nonuniform crossover. *Complex Systems* 8:257–293.

Mao, J.; Hirasawa, K.; Hu, J.; and Murata, J. 2002. Increasing robustness of genetic algorithm. In Publishers, M. K., ed., *Proceedings of the Genetic and Evolutinary Computation Conference*, 456–462.

Mayer, H. 1998. ptGAs–Genetic algorithms evolving noncoding segments by means of promoter/terminator sequences. *Evolutionary Computation Journal* 6(4):361–386.

Mühlenbein, H., and Schlierkamp-Voosen, D. 1995. Analysis of selection, mutation and recombination in genetic algorithm. In Banuhaf, W., and Eeckman, F., eds., *Evolution as a Computational Process*, 188–214. Berlin: Springer.

Rana, S. 1999. The distributional biases of crossover operators. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 549–556. Orlando (FL): Morgan Kaufmann Publishers.

Schaffer, J. D., and Morishima, A. 1987. An adaptive crossover distribution mechanism for genetic algorithms. In Greffenstette, J., ed., *Proceedings of the Second International Conference on Genetic Algorithms*, 36–40. Hillsdale (NJ): Laurence Erlbaum.

Spears, W. 1990. Using neural networks and genetic algorithms as heuristics for NP-complete problems. Master's thesis, Department of Computer Science, George Mason University, Fairfax, Virginia.

Spears, W. 1995. Adapting crossover in evolutionary algorithms. In *Proceedings of the Fourth Annual Conference on Evolutionary Programming*.

Suzuki, H., and Iwasa, Y. 1999. Crossover accelerates evolution in GAs with a Babel-like £tness landscape: Mathematical analyses. *Evolutionary Computation Journal* 7(3):275–310.

Syswerda, G. 1989. Uniform crossover in genetic algorithms. In Schaffer, J. D., ed., *Proceedings of the International Conference on Genetic Algorithms*. San Mateo (CA): Morgan Kaufmann Publishers.

Vekaria, K., and Clack, C. 1998. Selective crossover in genetic algorithms: an empirical study. In et al., E., ed., *Proceedings of Parallel Problem Solving from Nature V*, 438–447. Springer-Verlag.

Vrajitoru, D. 1999. Genetic programming operators applied to genetic algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 686–693. Orlando (FL): Morgan Kaufmann Publishers.

Whitley, D.; Mathias, K.; Rana, S.; and Dzubera, J. 1996. Evaluating evolutionary algorithms. *Arti£cial Intelligence* 85:245–276.

Wu, A.; Lindsay, R.; and Riolo, R. 1997. Empirical observations on the roles of crossover and mutation. In Bäck, T., ed., *Proceedings of the 7th International Conference on Genetic Algorithms*, 362–369. San Francisco: Morgan Kaufmann.

A Differential Evolution Algorithm for Automatically Discovering Multiple Global Optima in Multidimensional, Discontinuous Spaces

Zachary V. Hendershot

Miami University Computer Science and Systems Analysis Department
230 Kreger Hall, Oxford, OH 45056
{henderzv, moorefw}@muohio.edu

Abstract

This paper presents multiDE, an extension of Price and Storn's differential evolution (DE) algorithm that consistently outperforms state-of-the-art search techniques for identifying multiple global optima in multidimensional, discontinuous solution spaces. MultiDE automatically determines appropriate values for control parameters, and periodically updates those values at run-time. MultiDE requires little expert knowledge of the solution space, and is capable of searching both discontinuous and differentiable solution spaces. Innovative use of multiple subpopulations, minimum spanning distances, subpopulation expiration, and precision control contributes to multiDE's speed and effectiveness. Results from several benchmark problems reveal MultiDE's extraordinary power.

Introduction

DE (Price and Storn 1997) is a powerful and efficient technique for optimizing nonlinear and non-differentiable continuous space functions. Its simple yet powerful algorithm is illustrated by Fig. 1. DE was designed as a replacement for traditional methods of solving differential equations, such as simulated annealing (Jones and Forbes 1995) and the well-known simplex algorithm (Nelder and Mead 1965). DE has distinguished itself as a fast and easy-to-use numerical optimization tool.

DE begins by generating a random population of candidate solutions in the form of numerical vectors. The first of these vectors is selected as the target vector. Next, DE builds a trial vector by executing the following sequence of steps:

1. Randomly select two vectors from the current generation.
2. Use these two vectors to compute a difference vector.
3. Multiply the difference vector by weighting factor F (see Fig. 1). It has been found that generally as the number of population members used by DE increases, the value of F should be decreased in order to aid in the convergence process (Storn 1996).
4. Form the new trial vector by adding the weighted difference vector to a third vector randomly selected from the current population.

The trial vector replaces the target vector in the next generation if and only if the trial vector represents a better solution, as indicated by its measured cost value. DE repeats this process for each of the remaining vectors in the current generation. DE then replaces the current generation with the next generation, and continues the evolutionary process over many generations.

DE's shortcomings become apparent when a researcher begins experimenting with problems that have

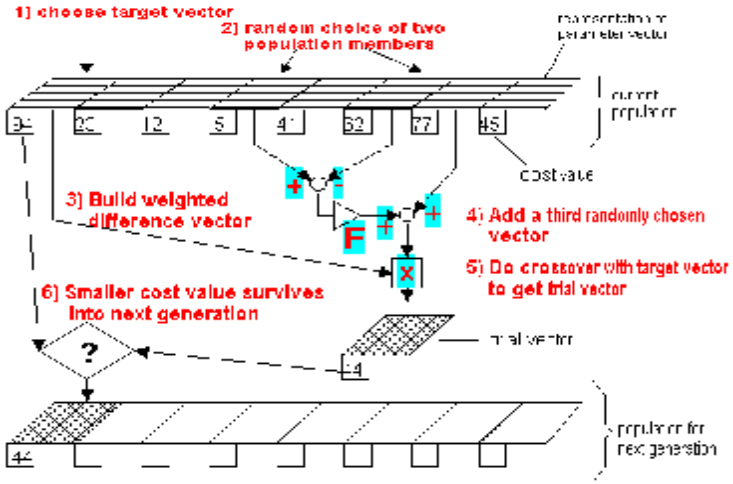


Fig. 1. The Traditional Differential Evolution Algorithm

large, complex solution spaces. Filter design, population are examples of problems that extend traditional assumptions concerning the formation of solutions. In such domains, it may be critical to enhance the problem-solving abilities of scientists and engineers by identifying as many globally optimal solutions as possible.

The New Approach

MultiDE incorporates several enhancements of the original DE framework, resulting in a fast and efficient method of convergence to multiple global optima. Researchers already familiar with DE can migrate to multiDE with minimal effort, and may begin exploiting multiDE's power to determine multiple global optima in complex search spaces.

First, multiDE assigns each candidate solution to one of several simultaneously evolving *subpopulations.* Each subpopulation consists of a user-specified number of vectors. Subpopulations evolve independently from one another, i.e., all three of the vectors used to create each trial vector are randomly selected from the subpopulation containing the corresponding target vector. MultiDE periodically transfers each current global optimum to a separate subpopulation known as "population 0", increases the number of subpopulations, creates a new set of candidate solutions for each subpopulation, and begins the evolutionary process anew. This technique significantly reduces the likelihood of premature convergence to a subset of global optima.

Second, multiDE uses a variable to specify the *precision* required for the current optimization problem. MultiDE considers real-valued solutions that differ by less than this variable's value to be equal. During evolution, each new trial vector is compared to each member of population 0; if the difference between a trial vector and any member of population 0 is less than the specified precision value, then multiDE considers the trial vector to have rediscovered a known globally optimal solution, and the trial vector is dropped from further calculations. This process accelerates the rate at which multiDE converges to new global optima. Reductions in the magnitude of the precision value force multiDE to recognize increasingly smaller differences between candidate solutions.

Third, multiDE dynamically adjusts the *minimum spanning distance* (MSD) between members of different subpopulations. MSD (Rumpler and Moore 2001) is initially calculated to be a factor of the user-specified maximum number of generations. When a new trial vector is created, multiDE computes its Euclidean distance from each of the target vectors in every other subpopulation of the current generation. If this distance is less than the MSD from at least one target vector, multiDE simply moves the trial vector a distance MSD in dynamics, and other applications of differential equations the opposite direction. This process is repeated until the trial vector is at least the MSD from every member of every other subpopulation. By preventing members of multiple subpopulations from gravitating to the same area of attraction, this technique encourages a more thorough search of the solution space, and thus increases the likelihood of finding larger numbers of global optima. Over many generations, multiDE slowly decreases the MSD until its value is less than or equal to the value of the precision variable described above. Each time the number of subpopulations is increased, multiDE resets the MSD to its initial value, and the process is repeated.

Fourth, multiDE introduces *subpopulation expiration* to accelerate the rate of convergence. MultiDE eliminates subpopulations from the currently evolving generation when they fail to discover new global optima after a specified maximum amount of computation. This technique results in a sizable linear increase in convergence speed. Shorter expiring times are most beneficial to researchers who want to quickly identify several solutions, but can forgo the opportunity of finding all possible solutions. A more comprehensive search can be made simply by increasing the expiring subpopulation variable to a much larger value.

MultiDE Solves Classic Optimization Problems

MultiDE's power can best be demonstrated by showing the speed and flexibility with which it solves several well-known multidimensional optimization functions. A "run", as tabulated below, was considered to be a complete execution of the multiDE algorithm with all parameters set and the function to be optimized loaded into the algorithm. Five test runs were made with each function. Each test run used a different random seed. A typical run used 60 vectors per subpopulation. Total computation time was recorded by executing multiDE via the Linux "time" command on a 3.06 GHz Intel Pentium 4 computer running Linux kernel 2.4.22.

The strategies listed below are designated using the standard format that Price and Storn set forth within the source code of their DE algorithm. For example, in DE/best/1/exp, the first value, "DE", indicates that a differential evolution technique has been used. The second parameter specifies the vector to be perturbed during the evolution process: "best" designates the best current population member, while "rand" designates a random population member. The third parameter represents the number of difference vectors to take from the perturbed vector to create a new trial vector. The fourth parameter specifies the crossover method to be used: "exp" designates an exponential method, while "bin" designates a binomial method.

The first equation analyzed was Branin's function (Branin 1972). This continuous function (Fig. 2) has six global optima within the specified range. DE/rand/1/exp was used as the strategy for optimizing this function. Initial test runs set the expiring subpopulation variable equal to the maximum number of generations. This value resulted in rapid convergence at the expense of a less comprehensive report of possible optimal solutions: the results from five test runs (Fig. 3) indicate that, on average, multiDE found 5.2 unique solutions in an average run time of 7.22 seconds. As few as six subpopulations were used, resulting in an average of 394,385 function evaluations.

Next, the value of the expiring subpopulation variable was increased by a factor of 10, and all five test runs were repeated under otherwise identical conditions. In each run, multiDE located all six global optima (Fig. 4). These results underscore a general rule: to increase the percentage of global optima found, the value of the expiring subpopulation variable must be proportional to the complexity of the solution space and number of solutions desired.

The second equation tested was Shubert's function (Fig. 5). This function has nine global optima within the specified range. Shubert's function lends itself to differential evolutionary optimization because, although its solution space is continuous and differentiable, it also offers sizable slopes that tend to drive adaptive techniques away from the global optima. A DE/rand-to-best/1/exp strategy was used to optimize this function. To increase the likelihood of converging on all possible solutions, the expiring subpopulation variable was increased by a factor of ten. The results of these tests (Fig. 6) illustrate multiDE's ability to rapidly converge to multiple optima. All five test runs determined all nine global optima solutions in an average execution time of only 5.47 seconds.

The third equation used to test multiDE was Rosenbrock's saddle (Rosenbrock 1960), shown in Fig. 7. This equation is an exception in this test suite: it has only one optimal solution. The reasons for its inclusion are to allow comparisons with more conventional optimizers, and to show that multiDE is flexible enough to solve problems having a single global optimum.

Using parameters similar to those used in previous tests and assuming that $n = 2$, multiDE quickly found the global optimum, in spite of the sizable overhead of multiDE's multi-pass algorithm. Note that after finding the single solution, multiDE continued executing many more iterations in an attempt to find additional solutions by increasing the number of subpopulations. Waiting for these additional subpopulations to complete introduced a function evaluation penalty that increased the total number of function evaluations (Fig. 8).

An interesting phenomenon arises when multiDE is

$$\textit{Minimize} \quad f(x) = (x_2 - (\frac{5}{4p^2})x_1^2 + (\frac{5}{p})x_1 - 6)^2 + 10(1 - \frac{1}{8p})\cos(x_i) + 10$$
$$\textit{Subject To} \quad -5 \le x_1, x_2 \le 15$$

Fig. 2. Branin's Function

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	5	8	782710	11.342	DE/rand/1/exp
2	5	6	233580	6.57	DE/rand/1/exp
3	5	6	227943	4.666	DE/rand/1/exp
4	5	7	501366	9.984	DE/rand/1/exp
5	6	6	226325	3.587	DE/rand/1/exp
Average	5.2	6.6	394384.8	7.2298	

Fig. 3. Branin's Function Test Results: Shorter Subpopulation Expiration Times

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	6	9	543094	41.190	DE/rand/1/exp
2	6	10	403493	36.243	DE/rand/1/exp
3	6	9	603912	42.302	DE/rand/1/exp
4	6	9	430934	37.584	DE/rand/1/exp
5	6	10	574833	44.238	DE/rand/1/exp
Average	6.0	9.4	511253.2	40.3114	

Fig. 4. Branin's Function Test Results: Longer Subpopulation Expiration Times

applied to the relatively simple equation illustrated in Fig.9. This equation has a total of sixteen global optima. When run with the same parameters specified for Shubert's function, multiDE quickly found up to ten unique solutions, but after multiple iterations failed to find the remaining solutions. Fig. 10 shows these results over five independent test runs.

To allow multiDE to solve problems having a larger number of global optima, the expiring subpopulation variable must be increased. After the expiring variable was increased by a factor of ten, multiDE used 22 subpopulations and 9,412,512 function evaluations to converge on all sixteen solutions in a single run in 15.72 seconds (Fig. 11).

MultiDE Outperforms Classic Optimization Algorithms

Efstratiadis (Efstratiadis 2001) defines effectiveness as follows:

> *[Effectiveness] indicated the probability of finding a global optimum starting from any random initial solution (or population of solutions)... A measure of the effectiveness of an algorithm in a specified problem is the number of successes out of a predefined number of independent runs.*

To demonstrate its effectiveness, multiDE was compared to the following classic algorithms: a multistart simplex technique (Torn and Zhilinskas 1989); a genetic algorithm-based method (Goldberg 1989); a shuffled complex evolution method (Duan, Gupta, and Sorooshian 1993); and a powerful annealing-simplex algorithm (Efstratiadis 2001) inspired by simulated annealing. Each algorithm was tested using Griewank's function and

$$\textit{Minimize} \quad f(x) = \left(\sum_{j=1}^{5} j \cos\left((j+1)x_1 + j\right)\right)\left(\sum_{j=1}^{5} j \cos\left((j+1)x_2 + j\right)\right)$$
$$\textit{Subject To} \quad -10 \le x_i \le 10 \quad \textit{for} \quad i = 1,2$$

Fig. 5. Shubert's Function

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	9	7	304930	4.546	DE/rand-to-best/1/exp
2	9	9	489445	6.549	DE/rand-to-best/1/exp
3	9	7	387545	5.528	DE/rand-to-best/1/exp
4	9	8	419430	5.762	DE/rand-to-best/1/exp
5	9	7	334309	4.984	DE/rand-to-best/1/exp
Average	9.0	7.6	387131.8	5.4738	

Fig. 6. Shubert's Function Test Results

$$\textit{Minimize} \quad f(x) = \sum_{i=1}^{n/2} 100\left(x_{2i} - x_{2i-1}^2\right)^2 + \left(1 - x_{2i-1}\right)^2$$
$$\textit{Subject To} \quad -10 \le x_i \le 10 \quad \textit{for } i = 1,\ldots,n.$$

Fig. 7. Rosenbrock's Saddle

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	1	7	449231	1.221	DE/rand-to-best/1/exp
2	1	7	449231	1.22	DE/rand-to-best/1/exp
3	1	7	449231	1.214	DE/rand-to-best/1/exp
4	1	7	453803	1.231	DE/rand-to-best/1/exp
5	1	7	465712	1.237	DE/rand-to-best/1/exp
Average	1	7	453441.6	1.2246	

Fig. 8. Rosenbrock's Saddle Test Results

$$\textit{Minimize} \quad f(x) = \left|(x-3)*(x-6)*(x-9)*(x-12)\right| + \left|(y-4)*(y-8)*(y-12)*(y-16)\right|$$
$$\textit{Subject To} \quad -20 \le x, y \le 20$$

Fig. 9. Simple Test Function with 16 Global Optima

Michalewicz's function. The calculated effectiveness of each function is defined as the percentage of global optima found.

Griewank's function (Fig. 12), as tested, has a ten-dimensional solution space of sufficient complexity to warrant extended convergence times and an increased number of function evaluations. This function is reported to have over 1000 optima in the range of interest (Efstratiadis 2001), and therefore presents an interesting benchmark for multiple global optima algorithms. Despite the multi-dimensional, non-convex nature of this function, the effectiveness of most of the algorithms in our test suite approached 100 (Fig. 14). MultiDE required only 7.35 seconds to find 100% of the solutions.

Michalewicz's function is illustrated in Fig. 13. Efstratiadis (Efstratiadis 2001) states that this function has more than 100 global optima in the specified range. This function, as tested, has a two dimensional solution space that proved to be very difficult for traditional optimization techniques: for the traditional four algorithms tested, the highest observed effectiveness rating was 58 (Fig. 14). In contrast, multiDE's effectiveness in solving Michalewicz's function was 96.

The importance of this finding is that, although the overhead of our multiple-pass evolutionary approach may be significant for simple test problems, multiDE's comprehensive search capabilities allow it to quickly converge on multiple globally optimal solutions in solution spaces that are too complex for the traditional optimization techniques included in this test suite. This result highlights the most important property of multiDE: its power to solve complex optimization problems quickly and reliably. MultiDE automatically adapts to the complexity of the given solution space, effectively utilizing the power of multiple simultaneously evolving subpopulations to find arbitrarily large numbers of global optima with less computational cost than traditional search techniques.

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	8	15	4147023	7.052	DE/rand-to-best/1/exp
2	10	13	3011941	5.443	DE/rand-to-best/1/exp
3	9	13	2999248	5.409	DE/rand-to-best/1/exp
4	9	13	3012578	5.442	DE/rand-to-best/1/exp
5	10	16	4762764	7.631	DE/rand-to-best/1/exp
Average	9.2	14	3586710.8	6.1954	

Fig. 10. Simple Multiple Optima – Results for Short Test Runs

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	16	22	9412512	15.722	DE/rand-to-best/1/exp

Fig. 11. Simple Multiple Optima – Results for a Longer Test Run

$$\textit{Minimize} \quad f(x) = \frac{1}{4000}\sum_{i=1}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$
$$\textit{Subject To} \quad -600 \le x_i \le 600$$

Fig. 12. Griewank's Function (Griewank 1981)

$$\textit{Minimize} \quad f(x) = -\sum \sin(x_i) * \sin^{20}$$
$$\textit{Subject To} \quad 0 \le x_i \le p$$

Fig. 13. Michalewicz's Function (Sun and Laio 2001)

Test Function	Dimensions (*n*)	Number of Optima	Multistart Simplex	Genetic Algorithm	SCE-UA	Annealing-simplex	multiDE
Griewank	10	>1000	100	89	100	99	100
Michalewicz	2	>100	35	31	44	58	96

Fig. 14. Test Results for Complex Functions: Effectiveness (%)

MultiDE was designed to be easy to use. Very few adjustments to control parameter values were necessary in order for the algorithm to reliably converge on multiple global optima; for example, to transition from testing Griewank's function to testing Michalewicz's function, the only values modified in the configuration file to reliably converge on multiple global optima were number of variables generated and the DE evolution strategy. (It is possible to change a larger number of variables within the configuration file, if the researcher desires finer control over the optimization process.) The nature of the algorithm lends itself to less reliance on the values of control parameters and more flexibility in convergence.

Conclusions

This paper described a new differential evolution algorithm that represents a significant contribution to the state-of-the-art in numerical optimization. Experimental results demonstrated multiDE's ability to consistently identify multiple global optima for complex functions (such as Michalewicz's function) that prove to be troublesome for such traditional optimization techniques as the multistart simplex method, the standard binary encoded genetic algorithm, the shuffled complex evolution method, and the annealing-simplex method. MultiDE allows researchers to experiment with a simple, flexible, and powerful differential evolution solver without a steep learning curve. MultiDE can be applied to a broad range of challenging optimization problems. Future research efforts will exploit multiDE's power to solve real-world problems with arbitrarily complex solution spaces.

References

Branin, F. H. Jr., 1972. "Widely convergent method for finding multiple solutions of simultaneous nonlinear equations", *IBM J. of Research and Development* 16(5): 504-522.

Duan, Q., V. Gupta, and S. Sorooshian, 1993. "A Shuffled Complex Evolution Approach for Effective and Efficient Global Minimization", *J. of Optimization Theory and Applications* 76(3): 501-521.

Efstratiadis, A., 2001. *Investigation of global optimum seeking methods in water resources problems*, M. Sc. thesis, Department of Water Resources, Hydraulic and Maritime Engineering, National Technical University of Athens.

Goldberg, D., 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley.

Griewank, A., 1981. "Generalized Descent for Global Optimization", *J. of Optimization Theory and Applications* 34: 11-39.

Jones, A. and G. Forbes, 1995. "An Adaptive Simulated Annealing Algorithm for Global Optimization over Continuous Variables", *J. of Global Optimization* 6(1): 1-37.

Michalewicz, Z. and T. Logan, 1994. "Evolutionary Operator for Continuous Convex Parameter Spaces", *Proc. 3rd Annual Conf. on Evolutionary Programming*, 84-97, World Scientific Publishing.

Nelder, J. and R. Mead, 1965. "A Simplex Method for Function Optimization", *Computer J.* 20(1): 84-85.

Price, K. and R. Storn, 1997. "Differential Evolution: Numerical Optimization Made Easy", *Dr. Dobb's J.*, April 1997, 18-24.

Rosenbrock H., 1960. "An Automatic Method for Finding the Greatest or Least Value of a Function", *Computer J.* 3: 175-184.

Rumpler, J. and F. Moore, 2001. "Automatic Selection of Sub-populations and Minimum Spanning Distances for Improved Numerical Optimization", *Proc., Congress on Evolutionary Computation 2001* 1: 38-43, IEEE.

Storn, R., 1996. "Differential Evolution Design of an IIR-Filter", *Proc. IEEE International Conf. on Evolutionary Computation*, Nagoya, Japan, 268-273, IEEE.

Sun, C-T. and Y-H. Liao, 2001. "An Educational Genetic Algorithm Learning Tool", *IEEE Trans. on Education* 44(2), IEEE.

Torn, A. and A. Zhilinskas, 1989. "Global Optimization", *Lecture Notes in Computer Science* 350, Springer-Verlag.

Directional Position of Objects in Image Understanding -a *fuzzy* landscape approach -

Prasanna Karunanayaka
ECECS Department
ML 0030
University of Cincinnati
Cincinnati, OH 45221, USA
prasanna@physics.uc.edu

Anca Ralescu
ECECS Department
ML 0030
University of Cincinnati
Cincinnati, OH 45221, USA
Anca.Ralescu@uc.edu

Abstract

The relationships between objects have been studied extensively in many different situations and the importance of a particular relationship depends on the context of the problem and the area of research. These relationships play a very important role in image understanding, by providing an effective tool for image processing and image segmentation. Among these relationships, directional relative position is extremely important since they provide information about spatial arrangement of objects in the scene. Such information is ambiguous and lack precise definitions, but humans have a rather intuitive and common way of understanding and interpreting them. Thus, in this context, fuzzy methods are very useful in providing a quantitative knowledge and a computational basis for interpreting imprecise spatial relationships. These methods provide a way of quantifying imprecise information expressed in a rather linguistic way. Thus, providing one with the knowledge to make accurate inferences based on such imprecise information. Several fuzzy approaches have been proposed in the literature and the aim of this paper is to develop a less computationally intensive approach that is consistent with other computationally expensive and accurate methods. The approach involves incorporating each pixel in the image space with quantified directional relative position information and a suitable aggregate that acts on a subset of this information. The method is even more appealing due to its simplicity and the ability to extend (or transfer) the same idea to handle imprecise information that arise in different contexts.

Introduction

The relationship between objects has been highlighted in very different contexts in different fields. These relationships mainly depend on the area of application and on the context in which they are being utilized in a particular application. In vision, for identifying shapes and objects, supporting spatial data and queries in database management systems and planning and reasoning in artificial intelligence to name a few of these diverse applications. According to the semantical hierarchy proposed in (Bloch & Ralescu 2003), spatial relationships can be mainly divided into topological and metric ones.

The spatial relationships play a very significant role in image understanding and processing. Since, distance and directional relative position constitute the metric relationships, these relationships fall under the second category. In this paper, we consider only directional relative position, which provides important information about the spatial arrangement of objects in the scene. The notion of directional relative position is rather ambiguous and clearly defies precise definitions. However, human beings have a rather intuitive and common way of understanding and interpreting them. As explained in Figure 1, the "all-or-nothing" definition leads to very unsatisfactory results even in situations of moderate complexity.

The limitations of purely qualitative reasoning have already been stressed in (Dutta 1991), and the advantage of adding semi quantitative extensions (as done in fuzzy set theory (Zadeh 1975) and (Dubois 1980)) to quantitative value in deriving useful and practical conclusions has already been mentioned, especially in relation to object recognition. The use of *fuzzy* approaches for representing directional relative position allows one to integrate both quantitative and qualitative knowledge fully utilizing the semiquantitative interpretation of *fuzzy* sets. As already mentioned in (Freeman 1975), this provides a computational basis for representation and interpretation of imprecise spatial relations, expressed in a linguistic way, that may include quantitative knowledge. Several *fuzzy* approaches have been proposed in the literature and the main purpose of this paper is to propose a computationally more efficient approach to complement the Histogram of Angles (the compatibility method)(Miyajima & Ralescu 1994b). The latter is more accurate but has the main disadvantage of being computationally very expensive. In this sense the methods proposed here are efficient (avoid repeated calculations) and capable of producing results that are comparable to more accurate methods.

The approach includes labeling each pixel with membership values corresponding to directional relative position information that are being considered (with respect to the origin) and an appropriate aggregation method that acts on a

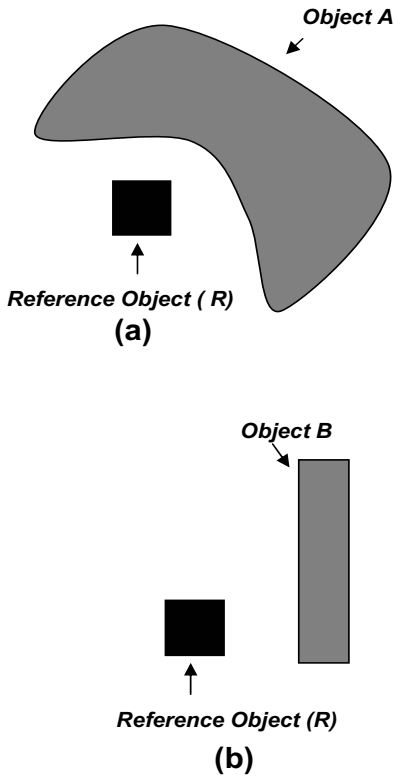


Figure 1: Two examples where the relative position of objects with respect to a reference object is difficult to define in a "all-or-nothing" manner.

subset of this information. Once the image landscape (*fuzzy*) is established, the degree to which an object satisfies a particular relationship can be calculated by a very simple mapping operation coupled with a suitable aggregation technique (we examine five). The main advantage of this method is that once the *fuzzy* landscape is established, evaluation of these relationships for other objects becomes computationally less expensive. Furthermore, once the relationship between the two objects in the *fuzzy* landscape is established (with respect to the origin), the proposed method may be extended to compute the spatial relationship between two arbirary objects in the image. The present paper only deals with the basics concept and the possible computational techniques associated with a *fuzzy* landscape.

Definitions of spatial relations using fuzzy sets

The main purpose of introducing fuzzy relative position definitions is to capture and describe spatial relations that do not have sharp boundaries. Most of the existing definitions for fuzzy relative spatial position for 2D objects rely on the measurement of angles between points of the two objects of interest (Miyajima & Ralescu 1994b) and (Keller & Wang 1995)]. In general, a fuzzy relationship is defined as a fuzzy set where the correspondence between the relation and the angle measurement is evaluated. There are other methods of defining fuzzy relations and one such method is based on linear cross sections of objects instead of only points (Matsakis & Wendling 1999). Finally, methods based on whole objects have been proposed, based either on learning from human evaluation (Keller & Wang 1996), on projections of the object, or on morphological approach (Bloch 1996) and (Bloch 1999).

Throughout this paper (fuzzy) spatial relations are defined in terms of angles. More precisely, given two objects, to begin with these are two points, a and b, the angle between the segment joining a and b and the *x*-axis of the coordinate frame(in 2D) as shown in Figure 2(a) is calculated. This angle denoted by $\phi(a,b)$, takes values in [-π,π], which constitute the domain on which the primitive directional relations are defined. As proposed in (Miyajima & Ralescu 1994b), Figure 2(b) illustrates four such relations *left (of)*, *right (of)*, *above* and *below* are defined as fuzzy sets using $\cos^2(\phi)$ and $\sin^2(\phi)$ for their respective membership functions. Although other functions are possible (e.g. triangular, or trapezoidal membership function), these simple functions proposed in (Miyajima & Ralescu 1994b) have very desirable properties in describing fuzzy relations. They define a fuzzy partition of [-π,π] and are invariant under rotation. (i.e. a rotation should correspond to a translation of the membership functions). Whatever the equations, the membership functions for these relations are denoted by their membership functions, μ_{left}, μ_{right}, μ_{up}, and μ_{below} that map f[-π,π] into [0,1]. In general, for an angle $\theta \in [-\pi, \pi]$ and a spatial relation $R \in \{right\ (of),\ above,\ left\ (of),\ below\}$, $\mu_R(\theta)$ should be read as *the degree to which θ satisfies the spatial relation R*.

Many applications rely on these four basic spatial relations, others being expressed in terms of these four. However as proposed in (Bloch & Ralescu 2003) the method can be extended to any direction. In 2D, a direction is defined by an angle α with the x-axis. Using this convention, the relationship *right (of)* corresponds to $\alpha = 0$. From $\mu_{right} = \mu_0$, we derive μ_α, describing the relationship *in direction* α, for any α as follows:

$$\forall \phi, \quad \mu_\alpha(\phi) = \mu_0(\phi - \alpha) \tag{1}$$

for instance,

$$\mu_0(\phi) = \begin{cases} \cos^2(\phi) & \text{if } \phi \in [-\pi, \pi] \\ 0 & \text{elsewhere} \end{cases} \tag{2}$$

This makes the definitions based on angle computation more general. Note that the relations obtained for $\alpha = \pi$, $\alpha = -\pi/2$, $\alpha = \pi/2$ are consistent with the definitions $\mu_{left}, \mu_{above}, \mu_{below}$ represented in Figure 2(b).

There are many ways of defining fuzzy relations as a combination of four operation. One such method as proposed in (Bloch & Ralescu 2003) is to express $\mu_\alpha(\phi)$ (ϕ in the direction α) as an aggregation of the degrees to which ϕ is *to the right of* and *above* weighted by a function to which α is right and above, respectively. The ratio between both weights is equal to $\tan(\phi)/\tan(\alpha)$. In general for any $\alpha \in [0, \pi/2]$, we have, for $\phi \in [0, \pi/2]$

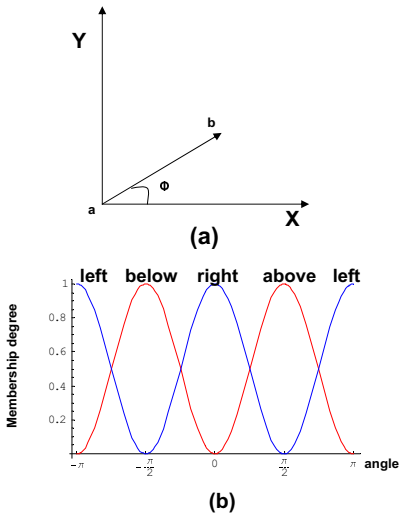


Figure 2: Definition of relative position of point *b* with respect to point *a*: (a) definition of angle ϕ for two object points *a* and *b* ; (b) definition of fuzzy relations representing relative position of *b* with respect to the point *a*.

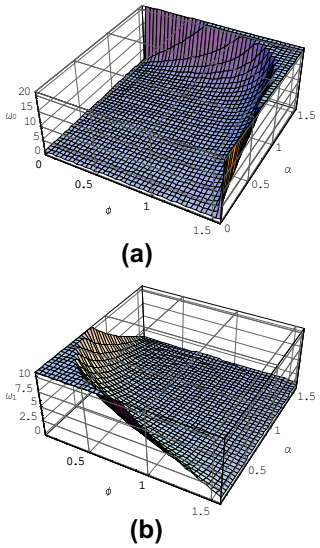


Figure 3: The behavior of the weights ω_0 and ω_1 with the angles α and ϕ. The behavior is rather interesting and the complements one another.

$$\begin{aligned}\mu_\alpha(\phi) &= \cos^2(\phi)\cos^2(\alpha)(1+\tan(\phi)\tan(\alpha) \\ &+ \sin^2(\phi)\sin^2(\alpha)(1+\cot(\phi)\cot(\alpha) \\ &= \mu_0(\phi)\mu_0(\alpha)\omega_0 + \mu_{\pi/2}(\phi)\mu_{\pi/2}(\alpha)\omega_1\end{aligned} \quad (3)$$

where

$$\begin{aligned}\omega_0 &= (1+\tan(\phi)\tan(\alpha) \\ \omega_1 &= (1+\cot(\phi)\cot(\alpha)\end{aligned} \quad (4)$$

and

$$\frac{\mu_0(\alpha)\omega_0}{\mu_{\pi/2}(\alpha)\omega_1} = \frac{\tan(\phi)}{\tan(\alpha)} \quad (5)$$

As shown in Figure 3, the weights ω_0 and ω_1 have an interesting behavior as the angles α and ϕ are varied. The behavior is complementary to each other and agrees very well with the intuition.

The histogram of angles method

Given a collection, Θ, of angles associated to one or more objects in the image, the histogram of angles method proposed in (Miyajima & Ralescu 1994a), (Miyajima & Ralescu 1994b) and (Miyajima & Ralescu 1997) computes the spatial relation(s) corresponding to these according tot he following steps:

1. Compute the histogram, H_Θ from Θ.
2. Interpret H_Θ as a fuzzy set. For simplicity of notation call this also by H_Θ. (Note that there are several ways in which this can be done. This study follows the original one in which the histogram normalized by its maximum value is interpreted as a fuzzy set.)
3. Match H_Θ to μ_R, where $R \in \{right(of), above, left(of), below\}$ and if necessary defuzzify.
4. Extract the result-set,

   $$\{\mu_R(\Theta); R \in \{right(of),\ above,\ left(of),\ below\}\}$$

   from which any question regarding the spatial relation of the region corresponding to Θ can be answered.

With respect to step (3) the match between H_Θ and μ_R is computed according to the *compatibility* of two fuzzy sets. The compatibility set $\mu_{C(H_\Theta,\mu_R}$ between H_Θ and the spatial relation R with membership function μ_R is defined, for any $u \in [0,1]$, following the extension principle for fuzzy sets, as shown in equation (6).

$$\mu_{C(H_\Theta,\mu_R)}(u) = \begin{cases} 0 & \text{if } \mu_R^{-1}(u) = 0 \\ sup_{v|u=\mu_R(v)} H_\Theta(v) & \text{otherwise} \end{cases} \quad (6)$$

The result is a fuzzy set - the *compatibility fuzzy set between* Θ *and* R. A global evaluation of the relation R can be extracted by defuzzifying $\mu_{C(H_\Theta,\mu_R)}$, using for example (as in (Miyajima & Ralescu 1994b)) the center of gravity of the compatibility fuzzy set, obtained acording to (7).

$$\mu_\alpha^R(A) = \frac{\int_0^1 u\mu_{C(H,\mu_\alpha)}(u)du}{\int_0^1 \mu_{C(H,\mu_\alpha)}(u)du} \quad (7)$$

It should be noted here, that steps (2) and (3) can be implemented in a number of other alternative ways as already indicated in (Bloch & Ralescu 2003).

Usually, in modeling (or learning) tasks there is a tradeoff between accuracy and computational efficiency. The Histogram of Angle method is very accurate as it takes into

account all the points (pixels) in the object(s) of interest. However, it is also computationally intensive: for deriving the spatial relations between two regions with m and n pixels respectively, it has to compute $m \times n$ angles (all the angles between pairs of points in the two regions). Moreover, if regions are repeated in subsequent queries, computations must be repeated. Therefore, improving upon the efficiency of the Histogram of Angle Method is one of the motivations for the current study and to investigate whether this improvement can be done so as to strike tradeoff (as needed) between accuracy and efficiency.

For example, for applications where the objects in the image move and change spatial relations and relative spatial relations (rapidly) the original Histogram of Angles method may not be suitable. Yet such systems have wide applicability in day to day life (e.g. weather tracking systems,radar systems and target selection systems). This paper proposes a simple technique to handle such systems based on the idea of a *fuzzy landscape* defined with respect to the origin. The goal is to minimize repeated calculations while providing meaningful results of acceptable accuracy. Another advantage of the fuzzy landscape approach is ability to extend it to handle modeling/recognition of other image characteristics that deal with concepts that can be represented using fuzzy sets.

The construction and some features of the fuzzy landscape

A fuzzy landscape is first defined with respect to the origin as a fuzzy set such that the membership value of each point corresponds to the degree of satisfaction of the spatial relation under consideration. In other words each pixel (Figure 4) in the image space is assigned a membership value for the relationship under consideration. This is simply making use of a spatial representation of a fuzzy set that already have proved to be very useful in image processing. More importantly, the fuzzy landscape is directly defined in the same space as the considered objects and unlike projection methods where the fuzzy area is defined in a 1D axis. This is a very desirable feature especially for image processing, because it allows us to extract other features from the image space if the need arises. A morphological method has been proposed (Bloch 1996), (Bloch 1999) using similar arguments but here the fuzzy landscape is defined as a fuzzy set around the reference object.

In order to evaluate how well an object matches with a the areas having high membership values (i.e. areas that are in the desired direction) an object is compared to the fuzzy landscape defined with respect to a point selected as origin. The fuzzy landscape together with an aggregation method either on the subset of angles Θ, or on the subset of membership values $\{\mu_R(\theta); \theta \in \Theta\}$, provides a global evaluation for the degree to which the object(s), represented by Θ, satisfies the spatial relationship R.

The main advantage of the proposed approach is the fact that in principle, any number of fuzzy relations can be assigned to a pixel and these relationships are computed once for all pixels. The degree to which an object (an object is simply a collection of pixels) is satisfying a particular spatial relationship can then be evaluated by a simple mapping operation avoiding repeated calculations.

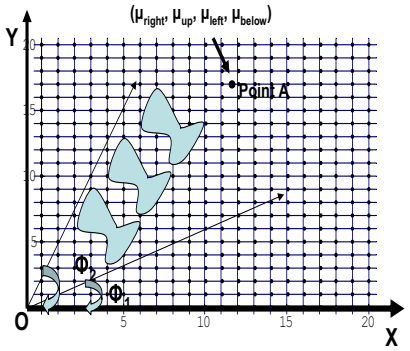


Figure 4: The fuzzy landscape defined for each pixel in the image space. This definition does capture the distance from the origin O in an implicit way.

As evident from Figure 4 the notion of a fuzzy landscape does capture the idea of distance from the origin to the object. This behavior depends on the position (in the fuzzy landscape) as well as the physical dimensions of the object. This feature can possibly be extended to capture the idea of relative position between objects in the fuzzy landscape.

The 2D (Figure 5) landscape that we have defined contains the membership values for the relationships *to the right (of)* and *above*(other values being zero because of the way we have chosen the coordinate system). The functions $\cos^2(\phi)$ and $\sin^2(\phi)$ represent fuzzy relationships as explained earlier. As clearly evident from the figure (Ref.5 different objects will have entirely different representations for spatial relations defined on the fuzzy landscape.

Computing the *fuzzy landscape* for objects in the image space

Using the mechanism outlined in the previous sections we compute the degree to which an object satisfies a certain fuzzy relationship according to five different aggregation methods. Success of a particular method can then be determined by comparing it with more accurate methods. But, even if it is not compared with other known methods the results should agree qualitatively with intuition of the spatial arrangement of objects in the image space. In most cases this is extremely valuable information and can be compared with the way humans understand and interpret these vague ideas. For the purpose of explaining the methods in computing fuzzy relationships the objects shown in Figure 6 are considered here.

Method 1 - Angle Aggregation: Average Angle

Method 1 involves aggregating all the angle values for a particular object once it has been mapped on to the fuzzy landscape. The aggregation method amounts to computing the average value (ϕ_{avg}) of these angles (the angle values can easily be included in pixels when the fuzzy landscape is initially defined). Once the ϕ_{avg} is calculated,

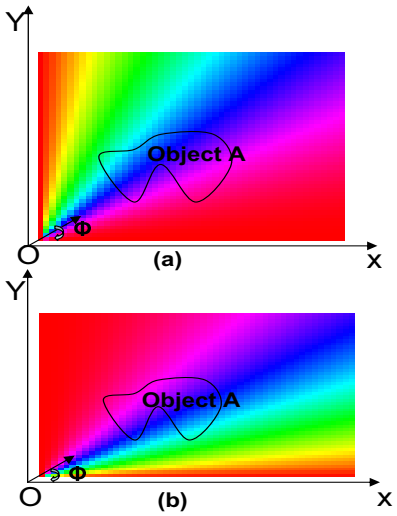
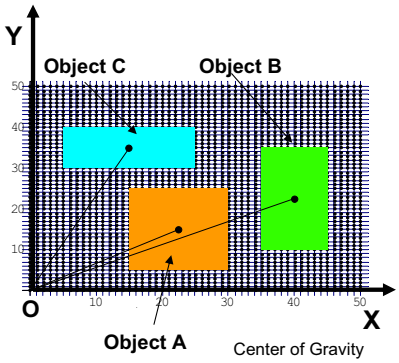


Figure 5: The fuzzy landscape for two relationships defined with respect to the origin O: (a) the relationship *to the right (of)* defined using $\cos^2(\phi)$; (b) the relationship *above* defined $\sin^2(\phi)$. Higher intensity corresponds to a higher degree to which a pixel(point) satisfies a spatial relationship.



Object A	(22.5, 15)
Object B	(40, 22.5)
Object C	(15, 35)

Figure 6: The objects (a collection of points) are mapped on to the fuzzy landscape defined for two relationships with respect to the origin O.

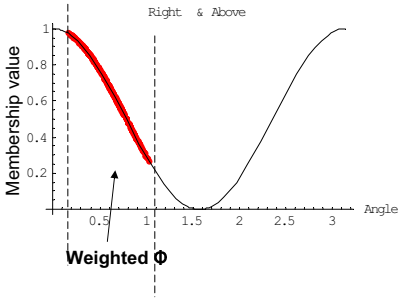


Figure 7: The calculation of $\phi_{wt.avg}$ for the relationship *to the right (of)* in method 3.

Object	Method 1	Method 2	Method 3	Method 4	Method 5
A	0 .70	0.68	0.75	0 .65	0.70
B	0.77	0.76	0.79	0 .76	0.73
C	0.15	0.17	0.22	0.19	0.20

Table 1: membership values obtained for the relationship *to the right (of)* for figures in Figure 6 using five different methods

using $\mu_{above}(\phi_{avg}) = \sin^2(\phi_{avg})$ and $\mu_{right}(\phi_{avg}) = \cos^2(\phi_{avg})$, the degree to which the object satisfies the relationships *above* and *to the right (of)* can be evaluated.

The obvious simplification of this method is to compute the physical center of gravity of the object under consideration and then computing the corresponding angle it makes with the x axis. Once the angle is known, the method can be straight forwardly extended to compute fuzzy relation as mentioned above. The following should be noted in connection with the use of Method 1:

- The average angle ϕ_{avg} may correspond to a point not in the actual object, just as the center of gravity of the object may lie outside of the physical object. Still, it is not clear if this will/should affect adversely the perception of the spatial relationship (for example, it seems that in the case of hollow regions, where it should not, it will not).
- The average angle value and the center of gravity are not uniquely associated to an object: for a given value of ϕ_{ave} or location of the center of gravity, there many be more than one region which has these caracteristics; the shapes of these different regions may affect the perception on spatial relations.

Method 2 - Membership Aggregation: Average Membership

Once the object is mapped on to the fuzzy landscape, the method aggregates all the membership values separately, for each fuzzy relationship in consideration. Here the aggregation method amounts to calculating the center of gravity of all membership values separately for each relationship. This method constitute the basic idea of a fuzzy landscape and the main objective here, is to minimize repeated calculations to make the computation more efficient.

Object	Method 1	Method 2	Method 3	Method 4	Method 5
A	0.29	0.31	0.41	0.33	0.35
B	0.23	0.24	0.29	0.26	0.26
C	0.85	0.83	0.86	0.80	0.75

Table 2: membership values obtained for the relationship *above* for figures in Figure 6 using five different methods

Method 3 - Angle Aggregation: Center of Gravity

Method 3 is simply an extension of Method 1 and involves computing a weighted average for ϕ using the following equation. Here the "weight" for each angle is the corresponding membership value for the relationship under consideration (in the following equation it is *to the right (of)*).

$$\phi_{wt.avg,right} = \frac{\int \phi \times \mu_{right}(\phi) d\phi}{\int \mu_{right} d\phi}. \quad (8)$$

The motivation for this approach is to compute the distributed for membership values for a particular fuzzy spatial relation. (Figure 7). Obviously if the object correspond to a considerably wide distribution of membership values Method 3 may be more appropriate than Method 1.

Method 4 - Membership Aggregation: Center of Gravity

Method 4 is an extension of Method 2 and involves computing a weighted average for the membership values (μ) obtained for a spatial relationship. The equation used is shown below.

$$\mu_{wt.avg,right} = \frac{\int \mu \times \mu_{right}(\mu) d\mu}{\int \mu_{right} d\mu} \quad (9)$$

The approach involves computing a membership function on membership values for a fuzzy spatial relationship (Figure 8. The main idea is to give more preference to membership values that has a higher frequency when considering a spatial relationship. In other words if there exist a membership value that corresponds to a relatively large number of points in the object, that membership value will be given a higher weight.

The computation of this new membership function, first includes calculating a normalized histogram (normalized by the total number of points) and then converting it to a fuzzy set using the standard method.

Method 5 - The Histogram of Angles -

Method 5 is the histogram of angles:compatibility method. As explained earlier, this is clearly the most accurate method for determining the spatial relations. The spatial relation of each object is computed with respect to the origin (effectively making origin the second object) to make a comparison with other results obtained by less accurate methods. This method involves calculating a histogram of angles, obtaining a fuzzy set and a fuzzy set matching operation (Figures 9 and 10).

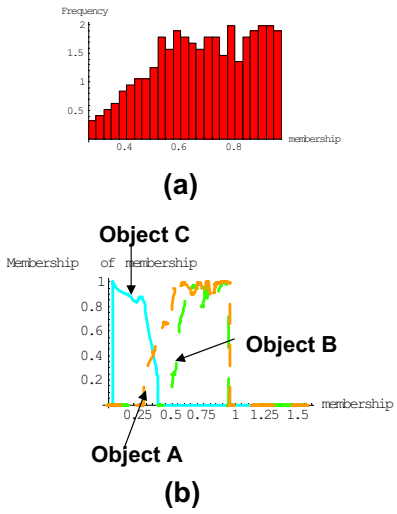


Figure 8: The calculation of a membership function on membership values for a particular spatial relation.: (a). the histogram calculated for the object A satisfying the relation *to the right (of)*; (b). the membership functions on membership values for the relationship *to the right (of)* for objects A, B and C.

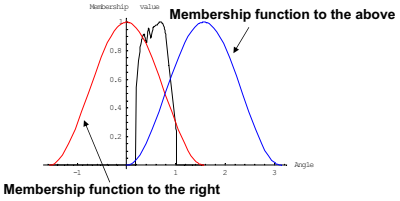


Figure 9: The matching of two fuzzy sets. The fuzzy set obtained by the histogram of angle method is matched with spatial relations *to the right (of)* and *above*.

In all these methods, by comparing the aggregated membership values for several objects, we can obtain information about the spatial arrangement of objects in the image space. The main feature of all these methods is that the relative direction is always given by two numbers *to the right (of)* and *above*).

Methods 1 through 5 are superior to the method of computing the physical center of gravity of an object and determining the spatial relations. They do not posses the main drawback mentioned earlier and it is always possible to find a point with the final membership value, that belongs to the physical object. This is a very desirable feature that can possibly, be extended to determine the spatial relations of objects with respect to each other.

Example 1 *In order to further examine the properties and the usefulness of defining a fuzzy landscape the objects shown in Figure 11 are considered. The objects overlap with each other making it more interesting than Figure 6. The spatial relations are computed using the five methods outlined above for the purpose of comparison (compared with Method 5).*

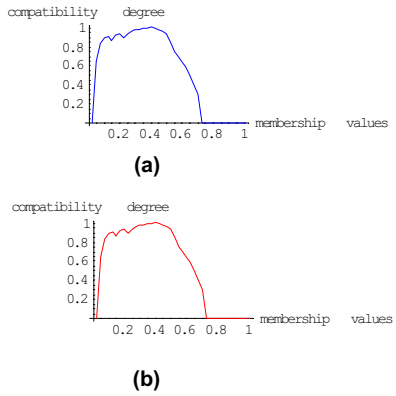
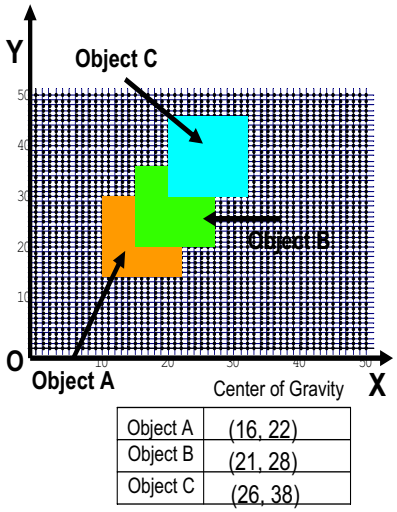


Figure 10: a). The compatibility set obtained by matching the relation *above*. b). The compatibility set obtained by matching the relation *to the right (of)*



Object A	(16, 22)
Object B	(21, 28)
Object C	(26, 38)

Figure 11: The image space for Example 1. The distribution of objects is much more complex than the Figure 6

All five method produce comparable results but Method 4 slightly out performs the other 3 (in both cases).

Conclusions

This study address the problem of directional position of objects in image understanding systems. The idea of a fuzzy landscape for an image was motivated by previous result, namely the Histogram:Compatibility method. Although the method is the most accurate, it has the main disadvantage of being computationally expensive. This become even more expensive if the image has many objects and the calculation has to be repeated for each spatial relation. The fuzzy landscape approach avoids this problem and possibly able to obtain comparable results. Five methods of implementing fuzzy landscape approach were discussed and compared for simple image spaces. All methods produced comparable results, but does not agree well with intuition in some cases because the method only deals with relative direction, not the relative position. The method can possibly be improved to overcome this problem by incorporating the notion of distance when the fuzzy landscape is initially defined. Although the fuzzy landscape approach is more transparent and easier to link with the image space, further research in the application domain is needed to decide whether the proposed approach can produce comparable results to that of more accurate ones.

Object	Method 1	Method 2	Method 3	Method 4	Method 5
A	0.35	0.35	0.41	0.40	0.42
B	0.36	0.36	0.40	0.38	0.40
C	0.32	0.32	0.34	0.31	0.34

Table 3: Membership values obtained for the relationship *to the right (of)* for figures in Figure 11 using five different methods

Object	Method 1	Method 2	Method 3	Method 4	Method 5
A	0.65	0.64	0.68	0.62	0.57
B	0.64	0.63	0.66	0.63	0.61
C	0.68	0.67	0.69	0.67	0.67

Table 4: Membership values obtained for the relationship *above* for figures in Figure 11 using five different methods

Acknowledgments

Both authors' work was partially supported by the Grant ONR N00014-03-1-0706.

References

Bloch, I., and Ralescu, A. 2003. Directional relative position between objects in image processing: a comparison between fuzzy approaches. *Pattern Recognition* 36:1563–1582.

Bloch, I. 1996. Fuzzy relative position between objects in image processing. *IEEE International conference on image processing.* 2:987–990.

Bloch, I. 1999. Fuzzy relative position between objects in image processing: a morphological approach. *IEEE Trans. Pattern Anal. Mach. Intell.* 21:634–642.

Dubois, D.; Prade, H. 1980. *Fuzzy sets and systems*. Academic press, New York.

Dutta, S. 1991. Approximate spatial reasoning: Integrating qualitative and quantitative constraints. *Int. J. Approximate reasoning* 5:307–331.

Freeman, J. 1975. The modelling of spatial relations. *Comput. Graph. Image Process* 4(2):156–171.

Keller, J., and Wang, X. K. 1995. Comparison of spatial relation definitions in computer vision. *ISUMA-NAFIP* 679–684.

Keller, J., and Wang, X. K. 1996. Learning spatial relations in computer vision. *FUZZ-IEEE* 118–124.

Matsakis, P., and Wendling, L. 1999. A new way to represent the relative position between areal objects. *IEEE Trans. Pattern Anal. Mach. Intell.* 21:634–642.

Miyajima, K., and Ralescu, A. 1994a. Spatial organization in 2d images. *IEEE International conference on fuzzy systems.FUZZ-IEEE*. 100–105.

Miyajima, K., and Ralescu, A. 1994b. Spatial organization in 2d segmented images:representation and recognition of primitive spatial relations. *Fuzzy Sets Syst.* 65:225–236.

Miyajima, K., and Ralescu, A. 1997. A fuzzy logic based method for describing the spatial relations in an image.

Zadeh, L. 1975. The concept of a linguistic variable and its application to approximate reasoning. *Inform. Sci.* 8:199–249.

ILLI-PAVE Based Pavement Moduli Backcalculation Using Artificial Neural Networks

Kasthurirangan Gopalakrishnan and Anshu Manik

Department of Civil and Environmental Engineering,
University of Illinois at Urbana-Champaign,
205 N. Mathews Ave., Room 3220, NCEL,
Urbana, IL 61801
kgopalak@uiuc.edu; manik@uiuc.edu

Abstract

In a mechanistic flexible pavement design procedure, the elastic moduli (resilient moduli) of the individual pavement layers must be known a priori for computation of critical pavement responses. The use of Falling Weight Deflectometer (FWD) data to backcalculate pavement layer moduli is a cost-effective and widely used method. Most of the commercial backcalculation programs do not account for the non-linearity of unbound granular materials and fine-grained cohesive soils and therefore do not produce realistic results. ILLI-PAVE is a structural axi-symmetric finite-element pavement analysis software which incorporates non-linear, stress dependent resilient modulus material models and failure criteria for granular materials and fine-grained soils. The goal of this research was to develop a procedure for backcalculating non-linear pavement layer moduli from FWD data using Artificial Neural Networks (ANN). A multi-layer, feed-forward network which uses an error-backpropagation algorithm was trained to approximate the FWD backcalculation function. The synthetic database generated with ILLI-PAVE was used to train the ANN. The study showed the potential for backcalculating asphalt concrete layer moduli and subgrade layer moduli using the ANN-based approach. Future studies would focus on applying the ANN-based backcalculation procedure to actual field data.

Introduction

A conventional asphalt concrete pavement is typically made up of three layers: a surface layer paved with Asphalt Concrete (AC) mix, a base or/and subbase layer made up of crushed stone, and a subgrade layer made up of natural soil. The deflection of a pavement represents an overall "system response" of the pavement layers to an applied load. When a load is applied on an asphalt pavement, the pavement layers deflect nearly vertically to form a basin. The deflected shape of the basin is predominantly a function of the thickness of the pavement layers, the moduli of individual layers, and the magnitude of the load. The Falling Weight Deflectometer (FWD) test is one of the most widely used tests for assessing the structural integrity of roads in a non-destructive manner. In an FWD test, an impulse load is applied to the pavement surface by dropping a weight onto a circular metal plate and the resultant pavement surface deflections are measured directly beneath the plate and at several radial offsets (see Figure 1). The FWD/HWD test tries to replicate the force history and deflection magnitudes of a moving truck tire.

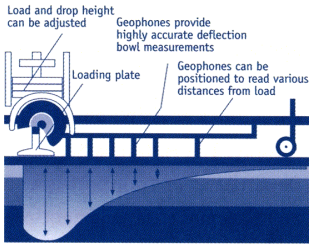


Figure 1. Illustration of FWD Deflection Basin

"Backcalculation" is the accepted term used to identify a process whereby the elastic (Young's) moduli of individual pavement layers are estimated based upon measured FWD surface deflections. As there are no closed-form solutions to accomplish this task, a mathematical model of the pavement system (called a forward model) is constructed and used to compute theoretical surface deflections with assumed initial layer moduli values at the appropriate FWD/HWD loads. Through a series of iterations, the layer moduli are changed, and the calculated deflections are then compared to the measured deflections until a match is obtained within tolerance limits. Most of the commercial backcalculation programs currently in use (e.g. WESDEF, BISDEF) utilize an Elastic Layer Program (ELP) as the forward model to compute the surface deflections. For example, WESDEF uses WESLEA and BISDEF uses BISAR.

The ELPs consider the pavement as an elastic multi-layered media, and assume that pavement materials are linear-elastic, homogeneous and isotropic. However, in reality, it has been found that certain pavement materials do not show linear stress-strain relation under cyclic loading. The non-linearity or stress-dependency of

resilient modulus for unbound granular materials and cohesive fine-grained subgrade soils is well documented in literature (Hicks 1970; Thompson and Robnett 1979). Unbound granular materials used in the base/subbase layer of an AC pavement show "stress-hardening" behavior (increase in resilient modulus with increasing hydrostatic stress) and cohesive subgrade soils show "stress-softening" behavior (reduction in resilient moduli with increased deviator stress). Therefore, the layer modulus is no longer a constant value, but a function of the stress state. Also, the ELPs do not account for the available shear strength of these unbound materials and frequently predict tensile stresses at the bottom of unbound granular layers which exceeds the available strength. Thus, the pavement layer moduli values predicted using ELP-based backcalculation programs are not very realistic.

ILLI-PAVE is a two-dimensional axi-symmetric pavement finite-element (FE) software developed at the University of Illinois at Urbana-Champaign (Raad and Figueroa 1980). It incorporates stress-sensitive material models and it provides a more realistic representation of the pavement structure and its response to loading. Several University of Illinois studies have supported the development of ILLI-PAVE based flexible pavement design procedures (Thompson 1992). The procedures have been successfully implemented by the Illinois Department of Transportation.

The goal of this research was to develop a tool for backcalculating non-linear pavement layer moduli from FWD data using Artificial Neural Networks (ANN). The reason for using ANN to accomplish this task is that once trained, they offer mathematical solutions that can be easily calculated in real-time on even the basic personal computers. Also, ANN can learn a backcalculation function that is based on much more realistic models of pavement response (e.g., ILLI-PAVE) than are used in traditional-basin matching programs. ANNs have been successfully used in the past for the backcalculation of flexible pavement moduli from FWD data (Meier and Rix 1993). However, they did not account for realistic pavement layer properties as ELP-generated synthetic database was used to train the ANN. Therefore, ILLI-PAVE was used in this study to develop the synthetic database which accounts for the nonlinearity in unbound material behavior. A multi-layer, feed-forward network which uses an error-backpropagation algorithm (Least Mean Square minimization) was trained to approximate the HWD backcalculation function.

Preparation of Training and Testing Data

To generate the training and testing data using ILLI-PAVE, a conventional flexible pavement section was modeled as a three-layered (AC layer, base layer, and a subgrade layer), two-dimensional, axisymmetric FE structure. A typical FWD test is performed by dropping a 9,000-lb load on the top of circular plate with a radius of 6 inches resting on the surface of the pavement. Deflections are measured at offsets of 0 (D_0), 6 (D_6), 12 (D_{12}), 24 (D_{24}), 36 (D_{36}), 48 (D_{48}), 60 (D_{60}) and 72 (D_{72}) inches from the center of loading plate. The effect of FWD loading was simulated in ILLI-PAVE.

The AC layer was treated as linear elastic material. Stress-dependent elastic models along with Mohr-Coulomb failure criteria were applied for the base and subgrade layers. The 'stress-hardening' K-θ model was used for the base layer:

$$M_R = \frac{\sigma_D}{\varepsilon_R} = K\theta^n$$

Where M_R is resilient modulus (psi), θ is bulk stress (psi) and K and n are statistical parameters. Based on extensive testing of granular materials, Rada and Witczak (1981) proposed the following relationship between K and n (R^2 = 0.68, SEE = 0.22):

$$\text{Log}_{10}(K) = 4.657 - 1.807n$$

The 'stress-softening' bilinear model was used for the subgrade layer:

$$M_R = M_{Ri} + K_1.(\sigma_d - \sigma_{di}) \quad for\ \sigma_d < \sigma_{di}$$
$$M_R = M_{Ri} + K_2.(\sigma_d - \sigma_{di}) \quad for\ \sigma_d > \sigma_{di}$$

Where M_R is resilient modulus (psi), σ_d is applied deviator stress (psi), and K_1 and K_2 are statistically determined coefficients from laboratory tests.

In this, the eight deflections computed at radial offset values which define the deflection basin, the thickness of AC surface layer, and the thickness of the base layer together form the *ten* input features. The natural subgrade is assumed to be of infinite thickness and is not considered. The modulus of the AC layer, E_{AC}, the modulus of the base layer K_b, and the elastic modulus of the subgrade layer, E_{Ri} represent the *three* output vectors.

A total of 1,500 data sets were generated by varying the pavement layer thicknesses and moduli values (note that K and n are related in the case of base layer). Of the total number of data sets, 1,125 data vectors were used in training the ANN and the rest 375 data vectors were utilized for the testing the network after the training was completed. The range of layer properties used in training the ANN are summarized in Table 1.

Table 1. Range of Layer Properties Used to Train the ANN

Pavement Layer	Thickness (inches)	Elastic Layer Modulus (psi)
Asphalt Concrete	3 – 15	100,754 – 1,995,419
Base	4 – 22	K_b: 3,014 – 14,000 n_b: 0.2 – 0.6
Subgrade	∞	1,012 – 14,000

Network Architecture

A generalized *n-layer feedforward artificial neural network* which uses an error-backpropogation algorithm (Haykin 1994) was implemented in the *Visual Basic* (VB 6.0) programming language. The program can allow for a general number of inputs, hidden layers, hidden layer elements, and output layer elements. Two hidden layers were found to be sufficient in solving a problem of this size and therefore the architecture was reduced to a four-layer feedforward network. A four-layer feedforward network consists of a set of sensory units (source nodes) that constitute the input layer, two hidden layer of computation nodes, and an output layer of computation nodes. The following notation is generally used to refer to a particular type of architecture that has two hidden layers: (# inputs)-(# hidden neurons)-(# hidden neurons)-(# outputs). For example, the notation 10-40-40-3 refers to an ANN architecture that takes in 10 inputs (features), has 2 hidden layers consisting of 40 neurons each, and produces 3 outputs.

Using the ILLI-PAVE synthetic database, the ANN was trained to learn the relation between the synthetic deflection basins (inputs) and the pavement layer moduli (outputs). To track the performance of the network a Root Mean Squared Error (RMSE) at the end of each *epoch*

$$RMSE = \sqrt{\frac{\sum_{j=1}^{N}\left[d_j - Y(X_j)\right]^2}{N}}$$

was calculated. An epoch is defined as one full presentation of all the training vectors to the network. The RMSE at the end of each epoch defined as:

Where d_j is the desired response for the input training vector X_j, and N is the total number of input vectors presented to the network for training. In order for the network to 'learn' the problem smoothly, a monotonic decrease in the RMSE is expected with increase in the number of epochs.

Parametric analyses were performed by systematically varying the choice and number of inputs and number of hidden neurons to identify the best-performance networks. As it was found that the prediction accuracy of the network remained the same for hidden layers greater than or equal to two, the number of hidden layers was fixed at two for all runs. The learning curve (RMSE Vs number of epochs) and the testing RMSE were studied in order to arrive at the best networks.

A range of (-0.2, +0.2) was used for random initialization of all synaptic weight vectors in the network. For this problem, an *asymmetric hyperbolic tangent* function (tanh) was chosen as the nonlinear activation function at the output end of all hidden neurons. Since, the final outputs (layer moduli) are real values rather than binary outputs, a *linear combiner* model was used for neurons in the output layer, thus omitting the nonlinear activation function. A smooth learning curve was achieved with a learning-rate parameter of 0.001.

Discussion of Results

Figures 2, 3 and 4 compare the target and ANN-predicted moduli of the AC, base and subgrade layers, respectively for the 375 test data vectors using the 10-40-40-3 network architecture. The R^2 values and the Standard Error of Estimate (SEE) values are reported in the plots.

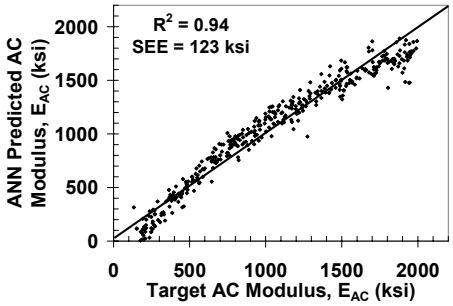


Figure 2. ANN Prediction of AC Modulus

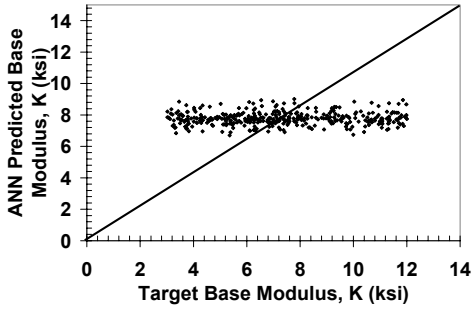


Figure 3. ANN Prediction of Base Modulus

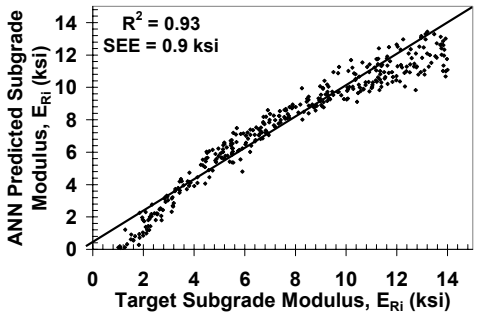


Figure 4. ANN Prediction of Subgrade Modulus

The base layer moduli were the hardest to predict. The difficulty associated with backcalculating the base layer modulus is a well recognized problem. It is sufficient to predict either 'n' or 'K' as there is a relation between the two. It is suggested that by including the ANN-predicted E_{AC} and E_{Ri} values as inputs to the network (i.e. 12-40-40-3 architecture), the chances of predicting K accurately will increase. Very good agreement is found between the target

and ANN-predicted layer moduli for AC and subgrade layers.

In a previous University of Illinois study, Thompson (1987) developed direct plot procedures and algorithms for backcalculating pavement layer and subgrade moduli from 9-kip FWD test results using comprehensive ILLI-PAVE databases. The AC thicknesses ranged from 1 to 8 inches and the base thicknesses ranged from 4 to 24 inches. For a conventional AC pavement, the regression algorithms are:

Subgrade Modulus

$$E_{Ri} = 24.1 - 5.08 * D_{36} + 0.28 * D_{36}^{2}$$

$$R^2 = 0.97 \qquad SEE = 0.76$$

AC Modulus

$$LOG\, E_{AC} = 1.48 + 1.76 * LOG\left(\frac{AREA}{D_0}\right) + 0.26 * \left(\frac{AREA}{T_{AC}}\right)$$

$$R^2 = 0.95 \qquad SEE = 0.110$$

Where
E_{AC} = AC modulus (ksi)
E_{Ri} = Subgrade soil resilient modulus at a repeated deviator stress of 6.2 psi (ksi)
D_0 = surface deflection @ 0 inches from center of loading plate (mils)
D_{36} = surface deflection @ 36 inches from center of loading plate (mils)
T_{AC} = thickness of the AC layer (inches)
AREA = a deflection basin parameter (inches)
[AREA (in.) = $6(1 + 2(D_{12}/D_0) + 2(D_{24}/D_0) + (D_{36}/D_0))$]

Using the ANN testing set, E_{AC} and E_{Ri} values were determined with the ILLI-PAVE regression algorithms. To avoid extrapolation, only those inputs, which fall within the input ranges for which the algorithms are valid, were considered. The E_{AC} and E_{Ri} values predicted by the ILLI-PAVE regression equations are shown in Figure 5 and Figure 6.

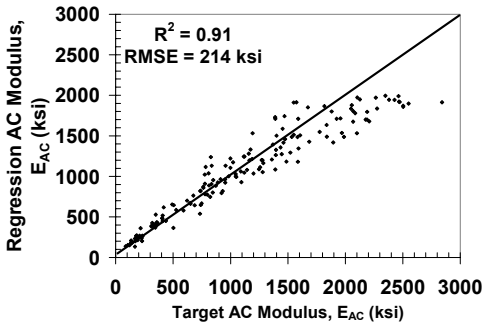


Figure 5. Regression Prediction of AC Modulus

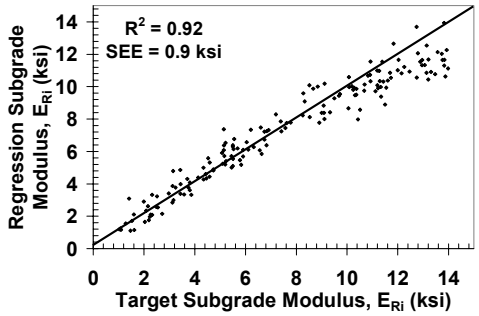


Figure 6. Regression Prediction of Subgrade Modulus

Thus, the ANN-predicted results are at least as good as those predicted by the ILLI-PAVE algorithms. In the case of AC moduli, the ANN-based approach shows better results. It is also noted that in the ANN prediction, 375 data points were used whereas only 164 data vectors were considered in using the ILLI-PAVE algorithms. A significant advantage of using the ANN-based approach over the regression approach is that the functional forms of the relationships are not needed a priori. Also, the robustness of the network can be improved by including the field data sets in the training process, as they implicitly incorporate noise and errors seen typically in field measurements.

Summary

In a mechanistic flexible pavement design procedure, the elastic moduli (resilient moduli) of the individual pavement layers must be known a priori for computation of critical pavement responses. The use of Falling Weight Deflectometer (FWD) data to backcalculate pavement layer moduli is a cost-effective and widely used method. In the FWD test, an impulse load is applied to the pavement surface by dropping a weight onto a circular metal plate and the resultant pavement surface deflections are measured directly beneath the plate and at several radial offsets. Backcalculation is the accepted term used to identify a process whereby the elastic (Young's) moduli of individual pavement layers are estimated based upon measured FWD surface deflections. The elastic moduli of the individual pavement layers are effective indicators of layer condition. They are also necessary inputs to mechanistic-based analysis and design of pavements. The ELP-based backcalculation programs do not account for the stress-dependency of unbound granular materials (used in the base and subbase layers) and fine-grained cohesive soils (used in the subgrade layer) and therefore do not produce realistic results. ILLI-PAVE is a pavement finite-element software that incorporates stress-sensitive material models and it provides a more realistic representation of the pavement structure and its response to loading. The goal of this research was to develop a procedure for backcalculating non-linear pavement layer

moduli from FWD data using Artificial Neural Networks (ANN). A multi-layer, feed-forward network which uses an error-backpropagation algorithm was trained to approximate the FWD backcalculation function. The ILLI-PAVE generated synthetic database was used to train the ANN. The study showed the potential for backcalculating asphalt concrete layer moduli and subgrade layer moduli using the ANN-based approach. The ANN-predicted results are at least as good as those predicted by the ILLI-PAVE algorithms. In the case of AC moduli, the ANN-based approach shows better results. Future studies would focus on applying the ANN-based backcalculation procedure to actual field data.

Acknowledgements/Disclaimer

We would like to thank Dr. Franco Gomez-Ramirez for generating the ILLI-PAVE synthetic database and for his valuable insights into the development of the ANN-based backcalculation procedure.

References

Haykin, S. 1994. *Neural Networks: A Comprehensive Foundation*. Macmillan College Publishing Company, Inc., New York.

Hicks, R. G. 1970. Factors Influencing the Resilient Properties of Granular Materials. Ph. D. diss., University of California, Berkeley.

Meier, R. W., and Rix, G. J. 1993. Backcalculation of Flexible Pavement Moduli Using Artificial Neural Networks. In *Proceedings of the 73rd Annual Meeting of the Transportation Research Board*, Washington, D.C.

Raad L., and Figueroa, J. L. 1980. Load Response of Transportation Support Systems. *Transportation Engineering Journal* 16(TE1).

Rada, G., and Witczak, M. W. 1981. Comprehensive Evaluation of Laboratory Resilient Moduli Results for Granular Material. *Transportation Research Record* 810.

Thompson, M. R., and Robnett, Q. L. 1979. Resilient Properties of Subgrade Soils. *Transportation Engineering Journal* 105(1).

Thompson, M. R. 1987. ILLI-PAVE Based Full-Depth Asphalt Concrete Pavement Design Procedure. In *Proceedings of the Sixth International Conference of Structural Design of Asphalt Pavements*, Ann Arbor, Michigan, USA.

Thompson, M. R. 1992. ILLI-PAVE Based Conventional Flexible Pavement Design Procedure. In *Proceedings of the Seventh International Conference on Asphalt Pavements*, Nottingham, U.K.

The impact of image partition granularity using Fuzzy Hamming Distance as image similarity measure

Mircea M. Ionescu

ECECS Department
ML 0030
University of Cincinnati
Cincinnati, OH 45221, USA
ionescmm@ececs.uc.edu

Anca L. Ralescu

ECECS Department
ML 0030
University of Cincinnati
Cincinnati, OH 45221, USA
Anca.Ralescu@uc.edu

Abstract

The choice of similarity measures affects greatly the performance Content-Based Image Retrieval (CBIR). Recently, the Fuzzy Hamming Distance, a fuzzy set based generalization of the well-known Hamming distance, from binary vectors to real valued vectors, has been proposed. Results so far, indicate that the Fuzzy Hamming Distance can be successfully used as image similarity measure. The Fuzzy Hamming Distance is suitable for this application because *it can take into account not only the number of different colors and the magnitude of this difference, but also its spatial distribution*. The current study reports on experiments on applying the Fuzzy Hamming Distance as a similarity measure between the color histograms of two images under different partition granularity and a weighted scheme of similarity aggregation.

Introduction

Large collections of images are accumulated every day. Finding a relevant set of images close to an example image is a major issue of Content-Based Image Retrieval (CBIR) systems. CBIR are the next evolution step of keyword-based systems in which images are retrieved based on the information of their contents. A survey of the functionality of current CBIR systems can be found in (Veltkamp & Tanase 2000).

CBIR systems are designed to allow users to query the image database in a natural way, by the image content. To achieve this goal, various features such as sketches, layout or structural description, texture, colors, are extracted from each image. As illustrated in the following scenarios for a CBIR system, where I denotes a generic image from a database of images D, and Q denotes a query image or pattern, query may simply specify an image or a pattern, or an image or a pattern in a certain location.

- **Scenario 1**:
  Find all images I in D which are similar to Q.
- **Scenario 2:**
  Find all images I in D which have the pattern Q at the top of the image.

The CBIR system would then find a subset of images that satisfy (are similar to) the query image.

Traditional measures of similarity between two images are based on L_p measures or weighted combination of L_p measures. More precisely, for two images with n features each, $X = (x_1...x_n), Y = (y_1...y_n)$ the L_p measure is defined as:

$$L_p(X,Y) = (\sum |x_i - y_i|^2)^p \quad (1)$$

One of the earliest content-based systems is QBIC, Query By Image Content, from IBM (Faloutsos 1994). For each image a set of features are extracted. These include: *color, texture, sketch* and, optionally, *objects defined inside the image*, in which case each object is determined manually by a contour and is described by its color, shape and texture. The color similarity is assessed computing the distance between the color histograms using (Enser 1993):

$$d^2(X,Y) = (X-Y)^t S(X-Y) \quad (2)$$

where an element s_{ij} of the matrix S is a similarity between colors i and j. It can be noticed that $d(X,Y)$ is the Euclidean distance when S is the identity matrix. For shapes, a weighted Euclidean distance between shapes attributes (area, circularity, eccentricity), as described in (Jain 1989), is used.

The main limitation of current CBIR systems is the difficulty of capturing semantic content of images and in taking into account spatial distribution of features. An image always contains some semantic information (for example, an image containing a *"seascape"*). Such semantic information can only be represented by some primitive features in present CBIR systems. On the other hand, the best way of describing semantic contents of an image is by using linguistic descriptions. Therefore, the main difficulty lies in establishing a *reliable* and *efficient* correspondence between such descriptions and image features, in a way that goes beyond use of key words. Some of the issues in this direction have been addressed in (Gasos & Ralescu 1997), (Ralescu 1995a).

Recently, a new similarity measure, the Fuzzy Hamming Distance, a fuzzy set-based generalization of the Hamming

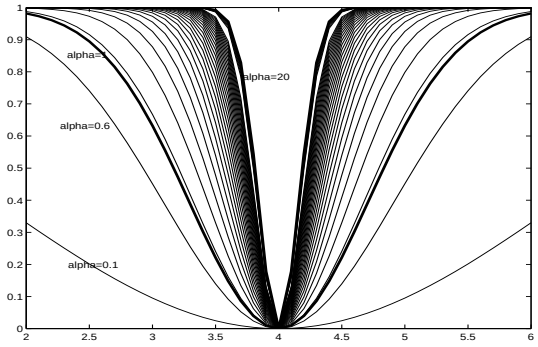


Figure 1: The membership function $d_\alpha(4, x)$ with $x = 2, \ldots, 6$ and α varying from 0.1 to 20 (some graphs for intermediate values of α are removed for clarity). (Ralescu 2003)

distance to real-valued vectors has been proposed (Ralescu 2003). Experiments to date, indicate that it can be applied successfully to measure the similarity between images (Ionescu & Ralescu 2004a), (Ionescu & Ralescu 2004b). The current study reports on further experiments with this approach to show how it can be used to capture, implicitly, semantic contents of the image, in addition to other image features (here color histograms). First, for the sake of completeness, the original definitions of (Ralescu 2003) are reproduced here.

The Fuzzy Hamming Distance

In a nutshell, given two real-valued vectors, FHD is the (fuzzy) number of components along which the two vectors are different. It is defined in the following steps:

Definition 1 *(***Degree of difference***(Ralescu 2003)): Given the real values x and y, the degree of the difference between x and y, modulated by $\alpha > 0$, denoted by $d_\alpha(x, y)$, is defined as:*

$$d_\alpha(x, y) = 1 - e^{-\alpha(x-y)^2} \tag{3}$$

The parameter $\alpha \geq 0$ *modulates* the degree of difference in the sense that for the same value of $|x-y|$ different values of α will result in different values of $d_\alpha(x, y)$. The (membership) function d_α defined in (3) has the following properties (Ralescu 2003):

1. $0 \leq d_\alpha(x, y) < 1$ with equality $\iff x = y$;
2. $d_\alpha(x, y) = d_\alpha(y, x)$;
3. for $x = a \pm c$, $d_\alpha(x, a) = e^{-c^2}$;
4. $d_\alpha(x, y) = d_\alpha(0, |x - y|)$.

Figure 1 illustrates $d_\alpha(4, x)$ when $x \in \{2, \ldots, 6\}$ for various values of α.

Using the notion of *degree of difference* defined above, *the difference fuzzy set $D(x, y)$* for two vectors x and y, is defined as:

Definition 2 *(***Difference fuzzy set for two vectors (Ralescu 2003)***): Let x and y be two n dimensional real vectors let x_i, y_i denote their corresponding ith component. The degree of difference between x and y along the component i, modulated by the parameter α, is $d_\alpha(x_i, y_i)$.*

The difference fuzzy set corresponding to $d_\alpha(x_i, y_i)$ is $D_\alpha(x, y)$ with membership function $\mu_{D_\alpha(x,y)} : \{1, \ldots, n\} \to [0, 1]$ given by equation 4:

$$\mu_{D_\alpha(x,y)}(i) = d_\alpha(x_i, y_i) \tag{4}$$

In words, $\mu_{D_\alpha(x,y)}(i)$ is the degree to which the vectors x and y are different along their ith component. The properties of $\mu_{D_\alpha(x,y)}$ are determined from the properties of $d_\alpha(x_i, y_i)$.

Since the (classical) Hamming distance is the **number** of components along which two bit vectors are different, a fuzzy extension of this concept must necessarily be based on the concept of cardinality of a (discrete) fuzzy set.

This concept has been studied by several authors starting with (Blanchard 1982), and continuing with (Zadeh 1983; Ralescu 1986; 1995b), etc. Here the definition put forward in (Ralescu 1986) and further developed in (Ralescu 1995b) is used.

Definition 3 *(***Cardinality of a fuzzy set (Zadeh 1983)***):Let*

$$A \equiv \sum_{i=1}^{n} x_i / \mu_i$$

denote the discrete fuzzy set A over the universe of discourse $\{x_1, \ldots, x_n\}$ where $\mu_i = \mu_A(x_i)$ denotes the degree of membership for x_i to A. *The cardinality, $CardA$, of A is a fuzzy set*

$$CardA \equiv \sum_{i=0}^{n} i / \mu_{CardA}(i)$$

where

$$\mu_{CardA}(i) = \mu_{(i)} \wedge (1 - \mu_{(i+1)}) \tag{5}$$

In (5) $\mu_{(i)}$ denotes the ith largest value of μ_i; the values $\mu_{(0)} = 1$ and $\mu_{(n+1)} = 0$ are introduced for convenience, and $\wedge$ denotes the min operation.

By analogy with the definition of the classical Hamming distance, and using the cardinality of a fuzzy set (Hamming 1950; Blanchard 1982; Zadeh 1983), the Fuzzy Hamming Distance (FHD) is now defined as follows (Ralescu 2003):

Definition 4 The Fuzzy Hamming Distance *Given two n dimensional real-valued vectors, x and y, for which the difference fuzzy set $D_\alpha(x, y)$, with membership function $\mu_{D_\alpha(x,y)}$ defined in (4), the* **fuzzy Hamming distance** *between x and y, denoted by $FHD_\alpha(x, y)$ is the fuzzy cardinality of the difference fuzzy set, $D_\alpha(x, y)$.*

$\mu_{FHD(x,y)}(.; \alpha) : \{0, \ldots, n\} \to [0, 1]$ denotes the membership function for $FHD_\alpha(x, y)$ corresponding to the parameter α. More precisely,

$$\mu_{FHD(x,y)}(k; \alpha) = \mu_{CardD_\alpha(x,y)}(k) \tag{6}$$

for $k \in \{0, \ldots, n\}$ where $n = |SupportD_\alpha(x, y)|$.

In words, (6) means that for a given value k, $\mu_{FHD(x,y)}(k;\alpha)$ is *the degree to which the vectors x and y are different on exactly k components (with the modulation constant α).*

Example 1 *Consider two vectors x and y with $|x-y| = (2,1,4,3,5)$. Their FHD is the same as that between the vector, 0 and $|x-y| = (2,1,4,3,5)$ (by property (4) of $d_\alpha(x,y)$).*

Therefore, the FHD between x and y is the cardinality of the fuzzy set $D(|x-y|,0) \equiv 1/0.9817 + 2/0.6321 + 3/1 + 4/0.9999 + 5/1$ obtained according to (5).

For $\alpha = 1$ this is obtained to be $FHD \equiv 0/0 + 1/0 + 2/0.0001 + 3/0.0183 + 4/0.3679 + 5/0.6321$.

In words, the meaning of FHD is as follows: the degree to which x and y differ along exactly 0 components (that is, do not differ) is 0, along exactly one component is 0, along exactly two components is 0.00013, along exactly three components is 0.0183, and so on. The difference fuzzy set and the fuzzy Hamming distance for this example are shown in Figures 2 and 3 respectively.

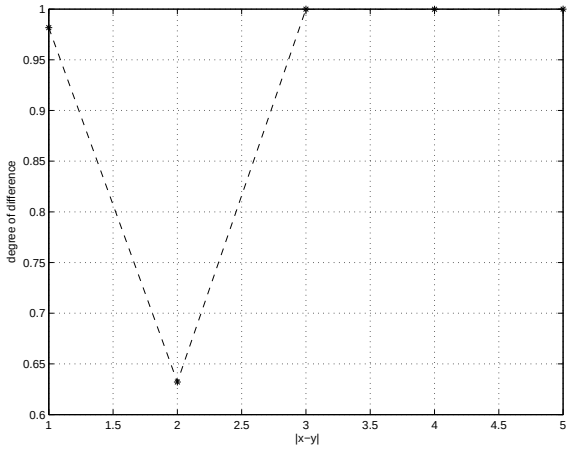


Figure 2: The difference fuzzy set for the data in Example 1.

Defuzzification of the Fuzzy Hamming Distance

As it is often the case with many fuzzy concepts, a non-fuzzy (crisp) approximation for them is useful. Several defuzzification techniques have been proposed in the literature on fuzzy sets.

Crisp Cardinality ($nCard$). For the defuzzification of the fuzzy cardinality fuzzy set A, the non-fuzzy cardinality $nCard(A)$, introduced in (Ralescu 1995b) is the only one which guarantees that the result is a whole number, and therefore satisfies its semantics, *"the number of ..."*. Starting from a set of requirements on $nCard(A)$, it is shown in (Ralescu 1995b), that $nCard(A)$ is equal to the cardinality (in classical sense) of $A_{0.5}$ the λ-level set of A with $\lambda = 0.5$. More precisely,

$$nCard(A) \equiv |\overline{\{x; \mu_A(x) > 0.5\}}| \tag{7}$$

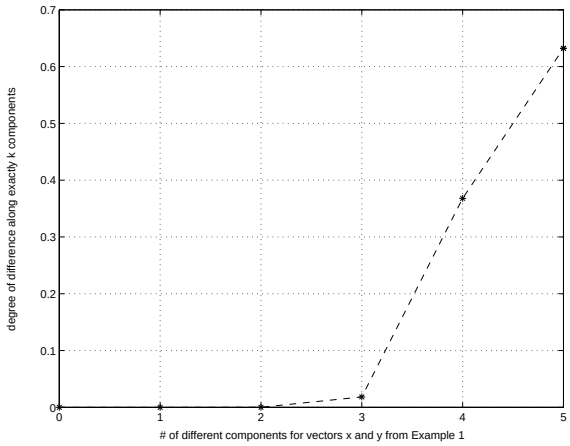


Figure 3: FHD for the data in Example 1 with the difference fuzzy set shown in Figure 2.

where for a set S, $\overline{S}$ denotes the closure of S. Since the fuzzy Hamming distance is the (fuzzy) cardinality of the difference fuzzy set $D_\alpha(x,y)$, its defuzzification leads to a non fuzzy **approximation** for it. It can be easily verified (Ralescu 2003), that with $\alpha = 1$ the defuzzification of FHD by taking its 0.5 level set is the standard Hamming distance. In other words, the defuzzification of FHD, $nFHD$ is the number of elements of $D(x,y)$ with membership greater or equal to 0.5. In the remainder of the paper $nFHD$ will denote the crisp quantity obtained according to (7) when A is the fuzzy set FHD defined in (6). That is, for the real vectors x and y,

$$nFHD_\alpha(x,y) \equiv nCard(D_\alpha(x,y)) \tag{8}$$

Center of Gravity (COG). Another defuzzification technique considered in this study is the center of gravity (COG) technique (COG) introduced in (Mamdani 1975) and very often used in defuzzification. In many cases (e.g. non-convex fuzzy sets), this technique does not give correct results (Ralescu 1995c). However, the Fuzzy Hamming Distance is a convex fuzzy set and in this case this technique is appropriate. According to this technique the defuzzification of the discrete fuzzy set $A \equiv \sum_{i=1}^{n} x_i/\mu_i$ is given by

$$COG = \frac{\sum_{i=1}^{n} x_i \mu_i}{\sum_{i=1}^{n} \mu_i} \tag{9}$$

For the data in the Example 1 the $nCard(A) = 5$ while the exact value for $COG = 4.5$. Strictly speaking the result must be integer. However in this study COG is not approximated as an integer so as to produce finer ranking of the final results.

The selection of the defuzzification technique depends to a large extent on the particular domain and problem to be solved.

Adapting the Fuzzy Hamming Distance

As already shown in (Ionescu & Ralescu 2004a), the modulator parameter α can be tuned to include a scaling factor

which controls FHD sensitivity to the extent of variation. For example, letting MAX denote the maximum value in the column domain, $0 \leq \beta \leq 1$ the percentage of MAX which would be considered a difference in the column values of two vectors, and

$$\alpha = \frac{ln\frac{1}{\epsilon}}{MAX^2}\frac{1}{\beta^2} \tag{10}$$

leads to

$$\mu_{D_\alpha(x,y)}(i) \equiv d_\alpha(x_i, y_i) \geq 1 - \epsilon \tag{11}$$

Setting for example, $\epsilon = 0.5$ equation (11) will assign a degree of difference greater than 0.5 (and hence included in the $nFHD_\alpha$) as soon as the difference between components of two vectors x_i, y_i, satisfies (12).

$$|x_i - y_i| \geq \beta MAX = \alpha\sqrt{\ln \epsilon} \tag{12}$$

For example, selecting $\beta = 0.1$ (10%), $MAX = 255$ and $\epsilon = 0.5$, means that if the difference between compared components of the vectors is greater or equal to 25 then the degree of change will be greater than $1 - \epsilon = 0.5$ (and therefore it will be counted in $nFHD$). The corresponding modulator value is $\alpha = 0.0011$. The experiments described in (Ionescu & Ralescu 2004a) show that in the case of two gray scale images, the modulator α can be tuned so that the corresponding value for $nFHD_\alpha$ is identical to the actual number of different pixels. Since β can be different for each column, FHD sensitivity can be controlled differently for each column (feature). Using different β for each feature means, in effect, weighing each feature thereby allowing a larger or smaller variation between feature values to be considered as a change.

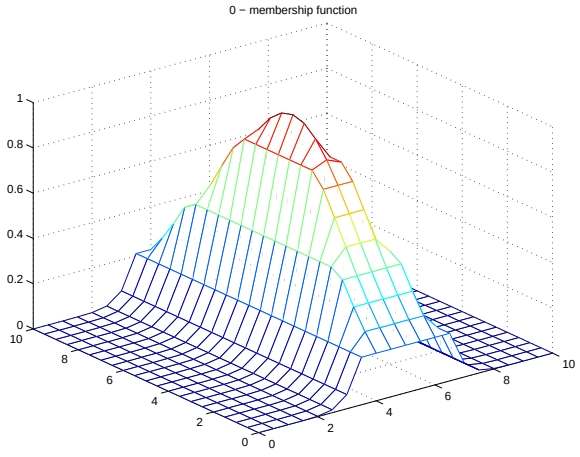


Figure 4: Membership function for the distance 0.

The Geometry of the Fuzzy Hamming Distance

To obtain a better understanding of FHD some geometrical aspects are considered next. Figures 4, 5, 6 and 7 show the membership function of the FHD distance from the point

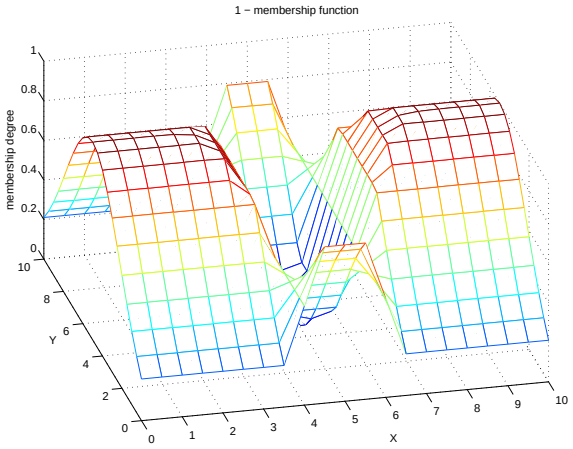


Figure 5: Membership function for the distance 1.

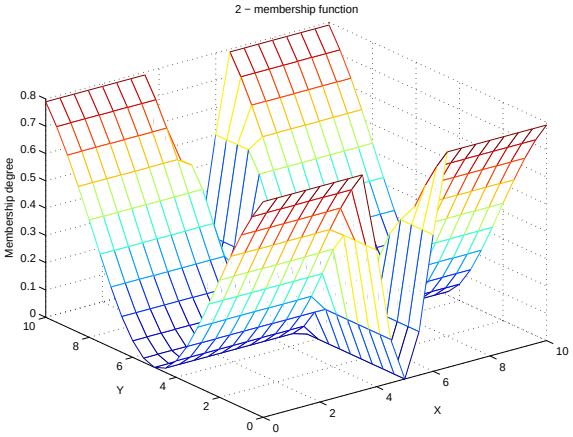


Figure 6: Membership function for the distance 2.

$(5, 5)$ and the point $(x, y) \in \{0 \ldots 10\} \times \{0 \ldots 10\}$, when $\beta_x = 0.1$ and $\beta_y = 0.3$, for all possibles values of the actual distance 0, 1 or 2 (since these points are in the 2D space). Figure 7 shows the distance distribution over the 2D space (x, y). At first glance, it may seem that FHD is much poorer than other distance measures, for example, the Euclidean distance ($\{0, 1, 2\}$ range of FHD versus $\Re_+$ for the Euclidean distance). However, a closer look reveals that this is not the case. Indeed, the components of the Euclidean distance appear, individually, in the degree of difference along each component (in fact, the Euclidean distance can be easily recovered from the difference fuzzy set). The effect of the components of the Euclidean distance on FHD can be seen from the in the plots from Figures 4, 5, 6 and 7.

It can be seen from these figures that μ_{FHD} measures how many components are changed and with what degree. In the middle, for points close to the center, $(5, 5)$, the distance is zero with high degree, while for points close to the margins (determined by the values for β_x and β_y) the degree for zero decreases and the degree of 1 is increases (passing $1 - \epsilon$ level). Beyond this margin, along both x and y, the degree

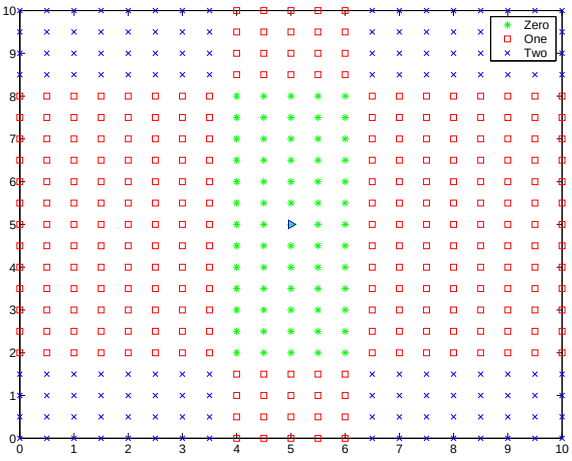


Figure 7: Distance distribution over $(x, y) \in \{0, \ldots, 10\} \times \{0, \ldots, 10\}$.

for the distance to be equal to 2 is increasing passing $\geq 1 - \epsilon$ and the degrees for 0 and 1 are again low. The distance distribution is represented in 2D space by rectangles (had β_x and β_y been equal these would have been cubes) and in nD by nCubes.

FHD for CBIR: System design

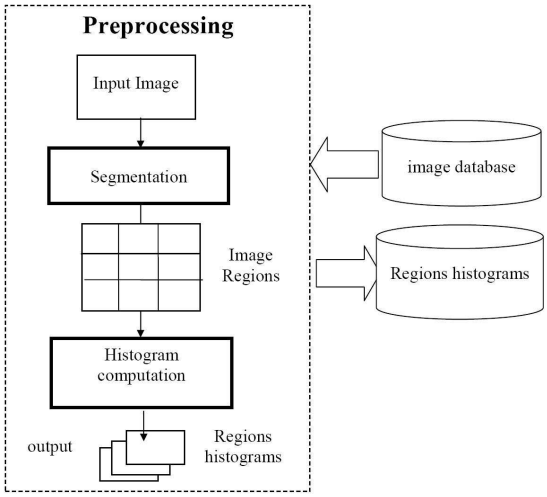


Figure 8: System architecture, preprocessing

The CBIR system used for this study is described in (Ionescu & Ralescu 2004b) and consists of the three modules as shown in Figure 8 and Figure 9. As it can be seen, the approach consists of three simple steps:

1. The **preprocessing module** extracts the information of interest (in this study the color histograms) from each of the images in the data base and the query image. The output of this module is a collection of color histograms.

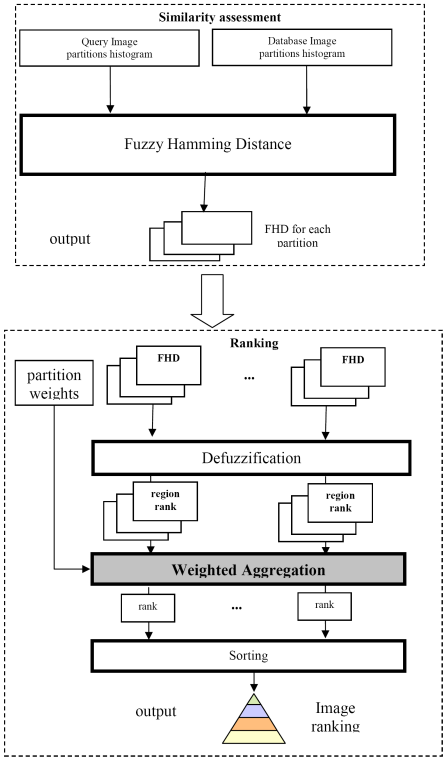


Figure 9: System architecture, similarity assessment and ranking

2. The **similarity assessment module** takes as input the information from the preprocessing module and computes the similarity (actually the FHD), between the query image and each image in the database. The output of this module is a collection of fuzzy sets (FHD).
3. The **ranking module** returns the outcome of similarity assessment ranked in decreasing order.

A detailed description of these modules follows.

The Preprocessing Module

The images of interest (data based and query image) are preprocessed according to the following steps:

1. **Partitioning:** Each image is partitioned in a $m \times m$ matrix of regions, where is $m = 1, 2, 3, 4, 8$.
2. **Compute color histograms:** The color histogram for each region is computed. Because the RGB color space can have 16M colors, quantization to 4096 colors, giving a color histogram of 4096 bins, is used. The number of pixels from each region that have the color corresponding to each of these bins is computed.

The output of the preprocessing module is the collection of $m \times m$ color histograms.

The Similarity Assessment Module

The collection of histograms corresponding to the image data base, and the color histogram of the query image, are

the input to this module. Each image is represented as a matrix of histograms, more precisely, image i is represented as $i = (H_i(k,l))_{k,l=1,\ldots,m}$. The query image q is represented as $q = (H_q(k,l))_{k,l=1,\ldots,m}$. The similarity function, $SimIm$, maps two images into a matrix whose entries are the FHD between corresponding regions of the argument images. More precisely, for two images, i and q,

$$SimIm(i,q) = (FHD_\beta(H_i(k,l), H_q(k,l)))_{k,l=1,\ldots,m} \quad (13)$$

where the FHD_β denotes the Fuzzy Hamming Distance parameterized by β (introduced in equation (10)), the percentage of difference in two colors (relative to the possible range) needed in order to consider these colors different. For the current experiments, the value $\beta = 0.1$, that is 10% of the maximum range, is used.

It is important to note here, that partitioning the image into regions and evaluation of the similarity between two images as a function of the similarities of their corresponding regions, assures that position information is also considered, along with color information. This way, two images that have the same color histogram but different semantic content (e.g. an image and the one obtained by rotation) will not necessarily be represented by the same vectors.

The Ranking Module

This module computes a score from the output of the similarity module and ranks images in order of the values of this score. To achieve this, in the current implementation ranking is done in two steps:

1. **Defuzzification:** The fuzzy sets returned by the similarity module are defuzzified. For comparison purpose three defuzzification methods were used: the center of gravity(COG), extracting the crisp version of the Fuzzy Hamming Distance, nFHD (defined in section), and the area under the curve (AUC).
2. **Aggregation and Ranking:** The results of the defuzzification step are aggregated into a final score using a weighted sum. The weighting is done by assigning value 10 to the most important regions and 1 to the rest. For example if the weight vector is $[10, 10, 1, 1]$, for 4×4 partitioning, the top regions will be considered the most important ones. Scores are ranked in nondecreasing order (images closest to the query have smallest number of different colors). The first n images, where n is a user-defined value, are returned as relevant to the query image.

Results

The approach described in the preceding sections is applied to 850 images in JPEG format from the Washington database (Images) (which contains 1000 images, the remaining 150 are in GIF format). Images are represented as $m \times m$ regions for all $m = 1, 2, 3, 4, 8$. For comparison purpose, a non-fuzzy similarity measure, based on the Euclidean distance, is also used.

For the similarity method using FHD we chose COG as defuzzification method because it output the best result-set as is shown in the previous study (Ionescu & Ralescu 2004b).

In the ranking module two weighting schemes are used for aggregation: (1) equal weights, and (2) regions in the right bottom portion of an image were assigned higher weights. The latter is important to enforce additional positional requirements (such as "make sure the retrieved image has a snow area like the one in the query image") for the retrieval procedure.

Varying Partition Granularity

Figures 10 and 11 show the result-set for different partition granularity, the images being partitioned from 1 region (1×1) to 64 regions (8×8) using $m = 1, 2, 3, 4, 8$. In this way the importance of the positional information increases.

It can be observed that for the selected query image, the relevance of the result decreases as m increases.

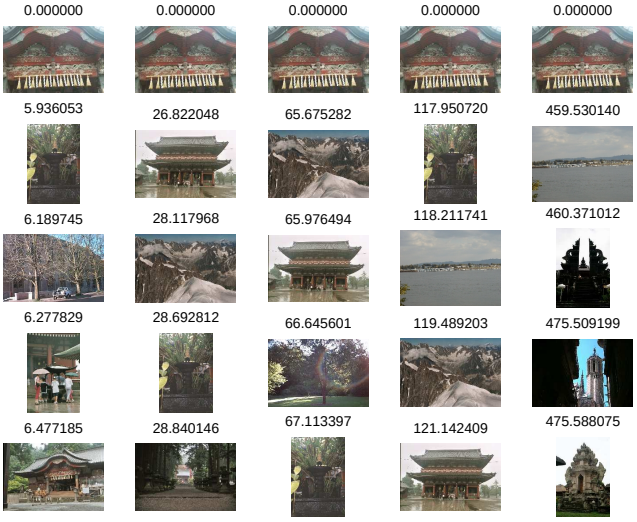


Figure 10: Top five query result-set using equal weights for each image region and COG defuzzification. Images have been divided into 1×1, 2×2, 3×3, 4×4 and 8×8 regions.

Looking closer at Figure 10, it can be seen that the query image represents a detail of a Japanese temple. In the database there is no other image that shows a similar detail, that is why position information fails to retrieve a relevant result, even though there is an image in the database showing the entire temple. In this case only using the histogram of the entire image (1×1 partitioning) brings the image with the entire temple in the result-set (line 5 column 1 of Figure 10). The same behavior is observed for Euclidean distance shown by Figure 11.

For other images, see Figure 12, the position information is valuable, and the relevance of the result-set increases when m increases. In this case the database contains more images similar to the query image, showing similar details and the position information helps to discriminate better between candidate images.

Figure 11: Top five query result-set using equal weights for each image region and Euclidean distance. Images have been divided into 1×1 2×2 3×3 4×4 8×8 regions.

Figure 12: Top five query result-set using equal weights for each image region and COG defuzzification. Images have been divided into 1×1, 2×2, 3×3, 4×4 and 8×8 regions.

Weighted aggregation

The result can also be improved by specifying some regions with higher relevance.

Figure 13 shows the result of assigning a higher weight to the right bottom portion of the image. The weight vector used for a 2×2 partitioning is $[1, 1, 1, 10]$. In other words we enforce the presence of the snow in that area. Each figure is divided in two groups of two columns, one group for each partitioning 2×2, 3×3, 4×4 and 8×8. The first column shows the ranking without weighting and the second with weighting for the corresponding partitioning.

It can be seen that the loss in performance given by the finer granularity (more regions) is overcome by the weighting schemes. For example, the fourth image of 8×8 partitioning, in Figure 14, is eliminated by this weighting. The same improvement is present for all partitioning schemes.

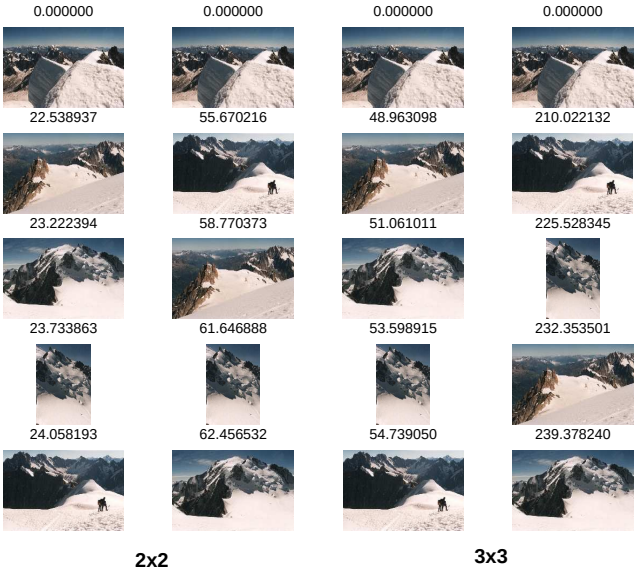


Figure 13: Top five query result-set using FHD with COG defuzzification. The results are grouped clusters of two columns. For each cluster the images have been divided into 2×2 and 3×3 regions. The first column of each cluster uses equal weights and the second uses a weighting schema where the regions in the right bottom portion of the image were assigned higher weights

Conclusions and future work

The relevance of the result-set of an CBIR system is hard to assess. It is based on what is relevant for a user, given the query image in a particular context. That is, relevance is both subjective and for the same user, it may be context dependent.

This study presented results of using a new similarity measure between images, Fuzzy Hamming Distance. The study shows that the relevance of the result-set can be improved by finding the proper the granularity of the image partitioning and/or by increasing the importance of some re-

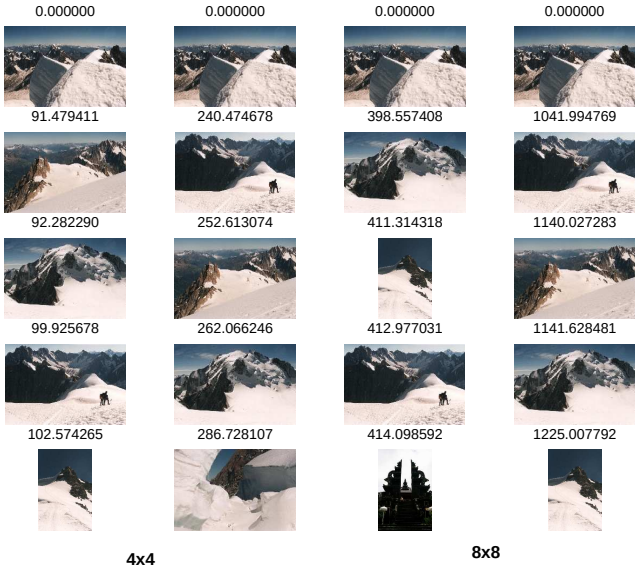


Figure 14: Top five query result-set using FHD with COG defuzzification. The results are grouped clusters of two columns. For each cluster the images have been divided into 4×4 and 8×8 regions. The first column of each cluster uses equal weights and the second uses a weighting schema where the regions in the right bottom portion of the image were assigned higher weights

gions of the image. Increasing granularity finds local solutions, because the position information becomes very important, and helps to distinguish between very similar images, and low granularity outputs a global solution where the position information is not important. Combining weighting and granularity can be used to optimize the query result-set.

Several mechanisms can be used to rank and aggregate the similarity measures in addition to the simple point-valued mechanisms illustrated here. These would include mechanisms that rank and aggregate directly fuzzy numbers without a (too early) defuzzification. Developing such mechanisms, experiments with other image features (HSV histograms, textures), and further experiments with the proposed methods belong to the future work in this direction. Another direction of improving the relevance of the result-set is to use an adaptive weighting algorithm. Using such an algorithm the user can specify the relevant or non-relevant images from the result-set, building iteratively the concept query.

Acknowledgment

Mircea Ionescu's work for this study was supported by a Graduate Fellowship from the Ohio Board of Regents. Anca Ralescu's work for this study was partially supported by a JSPS Senior Research Fellowship at the Brain Science Institute, RIKEN, Japan, and grant N000140310706 from the Department of the Navy.

References

Blanchard, N. 1982. Cardinal and ordinal theories about fuzzy sets. Fuzzy Information Processing and Decision Processes,149-157.

Enser, P. 1993. Query analysis in a visual information system. *Journal of Document and Text Management* 1:25–52.

Faloutsos, C. e. a. 1994. Efficient and effective querying by image content. *Journal of Intelligent Information Systems* 3,3/4:231–262.

Gasos, J., and Ralescu, A. 1997. Using imprecise environment information for guiding an image recognition system. *International Journal of Fuzzy Sets and Systems* 86.5:265–288.

Hamming, R. 1950. Hamming. error detecting and error correcting codes. *The Bell System Technical Journal* 16:147–160.

Images. Department of computer science and engineering, university of washington. http://www.cs.washington.edu/research/imagedatabase/groundtruth.

Ionescu, M., and Ralescu, A. 2004a. Fuzzy hamming distance as image similarity measure. accepted for presentation to IPMU-2004.

Ionescu, M., and Ralescu, A. 2004b. Fuzzy hamming distance in a content-based image retrieval system. IEEE-FUZZ-2004.

Jain, A. K. 1989. *Fundamentals of Digital Image Processing*. Prentice-Hall.

Mamdani, E. H.; Assilian, S. 1975. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies* 7(1):1–13.

Ralescu, A. 1986. A note on rule representation on expert systems. *Information Science* 38:193–203.

Ralescu, A. 1995a. Image understanding = verbal description of the image contents. *The Journal of the Japanese Society for Fuzzy Theory and Systems* 7(4):739–746.

Ralescu, D. 1995b. Cardinality, quantifiers, and the aggregation of fuzzy criteria. *Fuzzy Sets and Systems* 69:355–365.

Ralescu, A.; Hartani, R. 1995c. Some issues in fuzzy and linguistic modeling. Proceedings of FUZZ-IEEE/IFES'95, 1903-1910.

Ralescu, A. 2003. Generalization of the hamming distance using fuzzy sets. Research Report, JSPS Senior Research Fellowship, Laboratory for Mathematical Neuroscience, The Brain Science Institute, RIKEN, Japan.

Veltkamp, R., and Tanase, M. 2000. Content-based image retrieval systems: a survey. Technical report, UU-CS-2000-34, Utrecht University.

Zadeh, L. 1983. A computational approach to fuzzy quantifiers in natural languages. *Comput. Math.* 9:149–184.

Intelligent Content Based Title and Author Name Extraction from Formatted Documents

Eric G. Berkowitz, Mohamed Reda Elkhadiri, Tim Sahouri, Michel Abraham

Department of Computer Science
Roosevelt University
1400 North Roosevelt Boulevard
Schaumburg, IL 60173

Abstract

This paper describes the development of algorithms for extracting the title and the names of the authors from documents available on the World Wide Web. In this paper we describe several algorithms for doing so in a manner designed not to rely on specific stylistic dictates of any document formatting standard. Rather, they are designed to rely on a combination of overt and subtle cues that form a generalized, common standard for placing this information in a document and its easy extraction by readers.

Introduction

We began work on a system designed to aid students preparing papers in the social sciences and humanities perform literature reviews. The original concept was to design an automated literature review system. We thought such a system would provide on-the-fly research assistance not equal to that of a librarian but much better than that provided by a traditional Web search (Arms 2000). The system worked as follows. A student would use a traditional search engine such as Google or Yahoo to find a paper relevant to the topic being researched. The student would then pass the document to the automated system. Using the document's citations as a form of link, (Hitchcock et. Al 2002) the automated literature review system would then attempt to perform its own search for each of the documents cited in the bibliography, thereby aiding the student in collecting a range of relevant documents. Doing so required the system to be able to perform the search, download the first few documents returned by the search, and then scan the downloaded documents to determine if one of them is indeed the document being sought. In order to determine if a given document is indeed the one being sought, the system needed to be able to determine its title and the names of the authors. Thus we began to develop our first algorithms for intelligent title and author name extraction.

Background

Human readers determine the location of the title in a document from a combination of visual and contextual cues. Font size, location, isolation, letter case and grammar all combine to provide clues to the reader that a given segment of text is the title of the document. In specific instances, such as documents formatted for publication in a specific journal, predetermined formatting characteristics dictate, from a formatting and style perspective, exactly where and in what form the title should appear. We needed to develop an algorithm to determine the author of a document in an automated way by computer. Given the resources at our disposal, we needed to develop such an algorithm without reliance on visual cues that can only be determined through graphical analysis of a fully rendered document. Thus we began to develop our first context based algorithm.

Initial Development

We specifically wanted our algorithm to work on documents written by persons working in the social sciences and humanities. One of the issues we discovered very early in our attempts to develop the algorithm is that not only are there a plethora of document formatting styles, but that these styles are rarely adhered to exactly. Thus the extraction algorithm would need to rely on heuristics that are independent of the dictates of any given style. Another issue that plagues work in this field is that there is no provably correct answer. When a human reader looks at the document he relies on a preponderance of cues to determine what he/she believes is the title. The title page or header of a document is, to the human reader, a strongly structured document part (Buamann et. al. 1995). In fact, our early experiences in reading teaches us the skills we need to do so with a high degree of accuracy, yet still, it is impossible to "prove" that a segment of text is the title and not some other segment of text, particularly with many ornately formatted documents. Thus there is no way for an automated system to validate its own results. It can not "plug the answer back into the equation" as one might do in mathematics to verify a determination.

In order to avoid parsing a document formatting language, the first algorithm we developed relies on the program pdftotext that comes with the xpdf package to rerender a document in PDF format as a text document while attempting to preserve, as much as is possible, the original layout of the document.

The initial algorithm we developed assumes that the title of a document must come at or near the beginning of the text and thus is scans the first 100 lines of text in a document. It then scan these lines to find the title of the document and the names of the authors using a set of positive and negative heuristics. Positive heuristics are those that increase the likelihood that a line of text is part of the elements being searched for, that is, part of the authors' names or title. Negative heuristics are those that decrease the likelihood that a line of text is part of an element being searched for.

The algorithm first attempts to determine if the text being looked at is part of another element commonly found either before the title and authors' names in a document or in close proximity to them. A sample of these elements and the patterns used to form Java syntax regular expressions that constitute the heuristic test performed on a line of text to qualify it as probably being that element is shown in Table 1. The tests for an association assumes the author is associated with a school or institute. A line of text that matches one of these negative heuristics is determined not to be part of the title or authors' names. These lines are discarded by the algorithm.

Element	Pattern
e-mail address	"@"
Association	" [Uu]niversity of" "University" "Department" "Dept" "College\\s*\\z" "Institute"
City, State	"\\b[A-Z][A-Za-z]+,\\s+([A-Z]{2,})\\b" "\\b[A-Z][A-Za-z]+,\\s+([A-Z][A-Za-z]+)\\b"
URL	"http"
Phone number	"\\d{3,3}[\\s]*-[\\s]*\\d{3,}"
Volume Reference	"(\\b[Vv]ol[\\.:]\\s*\\d+)|(\\bp\\.\\s*\\d+)|(\\bpp.\\s*\\d+)|(\\b[Nn]o\\s*\\d+)"
End of Header	"Abstract"
Body Text	"\\b([a-z\\-]+[,\\.!\\?]*\\s+){4,}"
Copyright	"\\b[aA]ll [rR]ights [rR]eserved\\.*\\b"
Credits	"are a" "is a" "is the"
Date	"\\A\\s*\\w+\\s+\\d{4,4}\\s*\\Z"
Citation	"[Pp]resented\\sat\\sthe\\s.+?[Cc]onference" "\\b\\s*Journal of"
Blank	"^\\s*$"
Other Element	"[%\\|\\[{=\\$#!<>\\?]"
Too Short	"\\S+(\\s+\\S){2,}"
Prepostion or ucommon title word	" of ", " the ", " at ", " for ", " in "
Four+ lower case words	"\\b([a-z\\-]+[,\\.!?]*\\s+){4,}"

Table 1

The negative heuristics are followed by a set of positive heuristics that attempt to determine whether the text is part of the authors' names or part of the title.

Initial Results and Enhancements

Comment	Pattern or other Test
List of names	"\\S\\s+,\\S"
Title already found must be names	Title != null
Two digit numbers can only appear in the title	"\\d{2,}"

Table 2

The initial results of our algorithm were very promising and we were successful in extracting appropriate text from a wide variety of documents in a surprisingly large fraction of the test cases. The algorithm was, however far less successful at distinguishing between the title and the authors' names. While it was quite successful at excluding most parts of an address, it was including the city/state/country part of an address in a large number of cases, particularly in non-U.S. addresses where the format of <city>, XX where XX is a two letter state abbreviation, does not apply. Even with U.S addresses the algorithm was failing to remove the city/state part of an address if the author spelled out the state name instead of abbreviating it.

The algorithm was therefore enhanced with the ability to directly recognize many proper names and the names of all U.S states, Canadian provinces, and foreign countries. A list of most common surnames was retrieved from the U.S Census Bureau and a list of states and countries was created using data from the USPS and other sources on the Web.

The test for and address line that searched for data of the form <city>, XX was replace with a two step algorithm that test first for data of the form <city>, <state> where <city> and <state> can be of any length. If such text is found the second string, <state> is tested against the lists of countries, states, and abbreviations. If it is found in the list, this line of text is treated as part of an address and discarded. This significantly reduced the amount of address information slipping through the tests.

Distinguishing between names and title text was the next difficulty we tried to resolve. The positive heuristics already incorporated in the algorithm that were shown to be inadequate for making this determination were used as

the foundation of a scoring system. Each line of text that passed the earlier tests and was therefore determined to be either part of the title or the authors' names is passed through a sequence of tests. Two scores are maintained: one indicating the likelihood it is part of the title and the other indicating the likelihood it is the authors' names. The first scoring tests are the heuristics shown earlier. Next each line of text is run through a spell checker. The ratio of misspellings to the total number of words is calculated. If more than 80% of the words in the line are determined not to be correctly spelled English words, the score for being more likely names is incremented. While this test particularly stands out as limiting the algorithm to papers written in English, a review of the previous heuristics will show that, they too, assume English text and so using only an English dictionary does not reduce the scope of the algorithm's applicability. The next scoring test is to check whether the line starts with a surname from the list retrieved from the Census Bureau. If it does the names score is incremented again. Next the system checks if any previous lines were determined to be part of the title. If not, the system assumes that the title will precede the authors' names in the paper and increases the title score. If however, some text has been determined to be part of the title and there has been at least one blank line encountered since that text was found, the name score is incremented. A check for punctuation is performed. If commas are present in the line, given that addresses are presumed to already have been excluded, the name score is incremented. The final scores are then used to determine the nature of the text being checked. This algorithm is now named algorithm N-1 and the success of this algorithm will be shown in the section comparing algorithm performance at the end of this paper.

A Format Based Approach

Algorithm N-1 uses a long list of heuristics that rely on content alone to make determinations about the text. As a next stage in development we decided to take a format based approach. In a large number of document styles, the title is required to have the largest font on the first page of the document. Even in documents that do not adhere specifically to any given style, it has become a ***de facto*** standard to display the title in larger font than the rest of the document. One significant exception to this is the title page of a thesis or dissertation; yet still we decided to test the success of such an approach.

We began using components from the free PDFbox project, a freely available PDF parsing engine written entirely in Java. Instead of reducing the document to text we started working directly with the PDF formatted documents. First we developed a system to produce an XML formatted description of text segments and font information from a PDF, filtering out the rest of the PDF information in which we were not interested. Figure 1 is an excerpt of XML from analyzing a PDF document.

```
<?xml version="1.0"?>
<document
name="/home/eric/crawler/docdirs/a/NoTitles/www.northwestern.edu_ipr_publications_papers-mcc.pdf"><page
number="1">
<text fontName="ZapfDingbats"
fontSize="18.0">t</text><newline/>
<text fontName="Courier"
fontSize="24.0">Program on Community
Development</text><newline/>
<text fontName="Courier-Bold"
fontSize="30.0">Mapping
Community</text><newline/>
<text fontName="Courier-Bold"
fontSize="30.0">Capacity</text><newlin
e/>
<text fontName="Courier"
fontSize="13.0">NORTHWESTERN
UNIVERSITY<newline/>
...
</page>
<page number="2">
<text fontName="Helvetica-Bold"
fontSize="18.0">Mapping Community
Capacity</text><newline/>
<text fontName="Helvetica"
fontSize="12.0">by</text><newline/>
...
</page>
</document>
```

Figure 1

The next step was to create a parser for the XML data that would return one or more segments of text based on a set of criteria passed in. Our initial attempts to use this system were not highly successful. We determined that PDF documents incorporate elements other than text that are stored internally as text such as embedded graphics. In algorithm N-1 these elements were removed when the document was rerendered as text so we had not encountered them before. In order to avoid returning the textual gibberish that we were getting from searching for the largest font, we developed a two part test. The test searched for the largest font used on a "reasonable" segment of text where a reasonable segment of text is one that contains at least two three-letter words ("\\w{3,}\\s+\\w{3,}"). The test determines this font and then returns all of the text that uses the same[1] font size on the first page. This enhancement yielded significantly improved results. This algorithm however, has no mechanism for

[1] The PDF format uses floating point font sizes so "same" is within a margin of tolerance.

finding the names of the authors since we could not determine any generalized formatting convention for them. Thus it is only applicable for finding the title of a document. This algorithm appears as algorithm R-1 in the comparison at the end of this paper.

An Improved Content Based Approach

Given both approaches described above, the obvious best approach would be to combine their title extraction abilities and therefore increase the probability that we would find the title of the document. Unfortunately, the fact that given a segment of text one believes is the title of a document, there is no way to prove its correctness left us with no real way to perform this union. The algorithms use completely different approaches to finding the title and in the case where one returns information and the other does not, it is simple to use the only information available. But what of the more common case where information is returned by both and the information returned is not the same but overlaps or completely differs? How can one result be demonstrated programmatically to be superior to the other? It was therefore decided to perform a manual study of cases where algorithm R-1 performed better that algorithm N-1 and directly engineer into algorithm N-1 heuristics and mechanisms to handle these cases along with its current heuristic methods yield a single result. The result of this work was algorithm N-2.

The first lesson we learned was that while N-1 would occasionally truncate a title, R-1 would usually not truncate the same title. Analysis revealed that this occurred in the case where the title extended onto a second or third line and the last fragment was on one or two words long. R-1 would find the text since it would use the same font as the rest of the title but N-1 would disqualify the fragment as being too short to be a title. N-2 therefore incorporates a new heuristic that requires the first line of a title to have more than two words but allows subsequent lines to be shorter.

We also learned through our work on R-1 that PDF documents contain meta-information about the document that may include a title element. If there is a meta-title in the document it may or may not be the actual title of the document. A review of several common tools used to produce PDF documents revealed an assortment of behaviors regarding the generation of a meta-title. All tools allow an author to specify a meta-title when producing the PDF. If the author does do so the meta-title will probably be the actual title of the document. The manner in which the tools handle the case where no meta-title is specified differs greatly. Some tools use the filename of the PDF file being generated as the meta-title. Others use the name assigned by the operating system to the job when the PDF producing code is running. Others retain the title from document to document so that if an author once specifies the meta-title for a document that title will persist as the title of all subsequently produced documents until it is changed. N-2 utilizes this meta information in the following way. It first determines if the document contains a meta-title. If it does it searches to determine whether the meta-title found actually appears on the first page of the document. If it does, it is assumed that this is the title of the document. If it does not appear on the first page it is ignored. In the case where the meta-title is used, N-2 still attempts to find the title in the document as part of the process of searching for the authors' names, however the title found using the heuristic method is discarded and only the authors' names are used. N-2's use of the meta-title is shown in Figure 2.

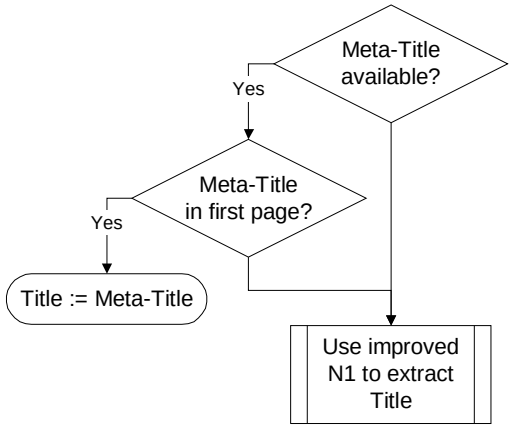


Figure 2

In reviewing the cases that N-1 missed the title and R-1 did not, we noticed that there is a large number of papers in which the title is not capitalized. R-1 was able to find the title via the formatting information while N-1 discarded the title assuming that a sequence of four or more lower case word signifies a regular sentence and not a title or list of authors' names. N-2 therefore does not automatically discard such sequences. Instead it only discards them if text from the title has been found and is detected to be set in initial-caps or if at least two blank lines have been detected since the ending text of a detected title. While this is not a perfect filter and lets some text through that it should not, it no longer excludes text that should be included as the earlier heuristic did. The full operation of N-2 is shown in Figure 3.

Performance Comparison

The following tables presents the relative performance of the algorithms on a random sample of 100 papers.

N-1	Exact Title	50.00%
	Title + Extra Text	11.00%

	Exact Title	50.00%
	Partial Title	13.00%
	Title+Author as Title	7.00%
	Missed Title	20.00%

R-1	Exact Title	53.00%
	Title+Extra Text	24.00%
	Partial Title	3.00%
	Title Missed	20.00%

N-2	Exact Title	43.00%
	Title+Extra Text	11.00%
	Partial Title	8.00%
	Title+Author as Title	5.00%
	Title Missed	32.00%

Intersection of Performance of All Algorithms	Exact Title	17.40%
	Title Missed	4.00%

Analysis

From the data in the performance table we can see that the changes to N-1 that yielded N-2 had an interesting effect. While reducing the overall chance of extracting the title text from a document, they also significantly increased the chances that if a title is returned, it will be the correct one. Accuracy was increased while overall effectiveness was decreased. Neither of the two heuristic algorithms has the effectiveness or accuracy of the formatting based algorithm, R-1.

What is perhaps more interesting is the intersection data. Only in 17.4% of the cases did all the algorithms extract the title correctly. Only on 4% of the cases did all three algorithms completely miss the title. An algorithm combining N-1, N-2 and R-1 should be able to achieve 96% efficiency although at varying degrees of accuracy.

Conclusion

The development effort that has gone into the creation of the algorithms described here has yielded interesting information both on the nature of formatted documents

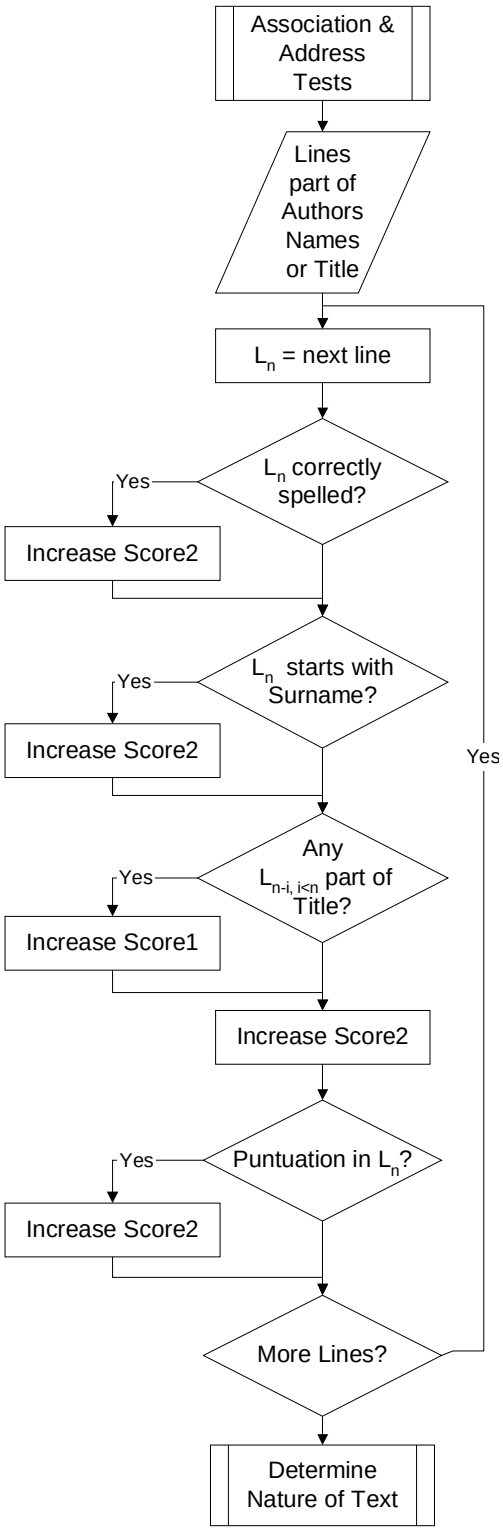


Figure 3

and appropriate methods for extracting information from them. The analysis shows that a hybrid approach should yield an algorithm capable of determining the title of a vast majority of documents without constraints on style and formatting. The major impediment to combining N-1, N-2 and R-1 is, as described earlier, the fact that we have no automated manner in which to verify correctness of title information. Thus, while human analysis can determine that, for example, in one case the title found by R-1 is correct while the title found by N-1 is partially correct, we have no method to make this determination programmatically, on the fly. Thus we are currently attempting to create a unified algorithm that combines elements of N-1, N-2 and R-1 but yields a single result.

There also remains a third approach to this analysis. While we have done much work on analyzing documents based on content and formatting, we have only begun to investigate analyzing them graphically using automated visual tools. Such analysis should yield a system that more closely mimics the activities of a human reader and should provide more insight into how to find desired information in documents.

References

William Y. Arms, "Automated Digital Libraries: How Effectively Can Computers Be Used for the Skilled Tasks of Professional Librarianship" D-Lib Magazine 6 (7/8) July/August 2000

Stephan Baumann et. al. "Document Analysis at DFKI: Part2: Information Extraction" Deutsches Forschungszentrum für Künstliche Intelligenz, RR-95-03, 1995.

Steve Hitchcock et. Al, "Open Citation Linking: The Way Forward" D-Lib Magazine, 8 (10) October 2002.

The Effects of Agent Topologies on Multiagent Planning Performance

Adam C. Langdon and Michael T. Cox

Department of Computer Science and Engineering

College of Engineering & Computer Science

Wright State University

Dayton, OH 45435-0001

{alangdon,mcox}@cs.wright.edu

Abstract

In multiagent distributed planning systems, agents can act cooperatively toward a common goal. Using multiagent goal transformations, the planning process can be distributed over an appropriate number of autonomous agents. Each agent will attempt to plan its part of the problem using its share of the knowledge. We will show how the distribution of knowledge affects the planning performance of the system. The factors we will examine in this process include the number of agents used in the system, which agent will initiate the planning process, and what knowledge, if any, will be duplicated among the agents. We will use examples from the logistics transportation domain to demonstrate how these factors affect performance.

1 Introduction

The level of control in a multiagent system can vary from complete centralized control to a fully distributed system. With centralized control, knowledge about all of the agents and their actions is known. The planning process as a whole can be monitored, and errors in the system can be detected. For example, if an agent is performing unnecessary steps to solve a problem, a control mechanism can correct the agent's planning. Alternatively, a fully distributed multiagent system relies only on the knowledge of each autonomous agent. While requiring less processing overhead, greater effort must be taken in its design to ensure proper coordination between the agents. This design process includes the development of an agent topology, which determines the number of agents in the system, how knowledge is distributed among them, and which agent will initiate the planning process. Because an agent depends only on what it knows, the structure of this topology may significantly impact its performance.

Cox, Elahi, and Cleerman (2003) first demonstrated a fully distributed multiagent system with multiagent goal transformations as its coordination method. In this system, the planning process is divided among multiple agents using domain-specific topologies. Elahi (2003) further investigated the performance of this system, showing how the number of agents affected performance. We will expound upon these findings and discuss how topological factors affect overall planning performance. Using experimental results, we will examine the impact of both the initiating agent and an instance of knowledge distribution overlap, in which two identical agents are used. We have used a planning and learning architecture called PRODIGY (Carbonell *et al.,* 1992; Veloso *et al*., 1995) as the underlying framework, and we have used Prodigy/Agent (Cox *et al.,* 2001) as individual agents.

The paper is organized as follows: Section 2 describes the method used for multiagent coordination. Section 3 defines an agent topology and its important factors. Section 4 continues with an example that illustrates the approach. Section 5 discusses the experimental results. Section 6 focuses on some future works still to be done. Finally, section 7 has concluding remarks.

2 Multiagent coordination

Coordination among agents in distributed planning can be performed by goal transformations. If an agent cannot achieve a goal by itself, it can transfer the goal into one that is achievable through co-operation. A multiagent goal transformation is defined in (Cox *et al.,* 2001) as follows: to solve a goal G, solve instead a goal G′ that generates a sub-solution, and then pass the remainder of the goal (i.e., G minus G′) to another agent. We distinguish between dynamic preconditions that may be adopted as subgoals because operators exist that modify the state, and applicability conditions (i.e., static preconditions) for which no operator of any agent exists to change the state. For example, no operator may exist to change the weather in a particular domain, yet some operators may require good weather for the operator to be applicable.

To achieve a particular goal state G, an agent must have an operator that can achieve the state. Now all open dynamic preconditions (those whose state is not true) are split into two sets. Set one contains states for which the agent has an operator, and set two contains states for which

it does not. The agent solves those preconditions in set one, and then it requests other agents to solve each state of set two. For example, a planner of a transportation domain might have two dynamic preconditions for the operator representing the action "Load Airplane P with object O at location L." One precondition requires that the object O be at the location L and the second precondition requires that the airplane P be at the same location. Suppose an agent A only has an operator to move objects but does not have an operator to fly airplanes. In this example, A will solve the first condition, and ask another agent to solve the second.

3 Topological Factors

A fully distributed multiagent system sacrifices the benefits of centralized control. Each agent acts independently without knowing the progress of the other agents. Therefore, the performance of this system will be affected by the properties of the agent topology and the properties of the planning problem. We have chosen to focus on the former and investigate how topological factors relate to performance.

The design of an agent topology can be difficult, and can generally be seen as a complex search problem in its own right (Durfee, 1999). To better examine this complexity, we will define an agent topology as the factors that make up its design. These three factors are the number of agents in the system, how knowledge is distributed among these agents, and what agent will initiate the planning process.

3.1 Agent distribution

The first step in designing an agent topology is deciding how many agents will be used. With a smaller number of agents, each one possesses a large amount of knowledge about the planning process. While less communication will be needed between agents, each agent will be responsible for solving a larger part of the plan. When more agents are used, each agent only possesses a small amount of knowledge. Consequently, it will only be required to solve a small part of the problem, but more communication will be needed.

Because multiple agents can work in parallel, it would seem best to favor more agents within the system. However, a larger number of agents requires a larger amount of knowledge distribution. This may negatively impact performance, as we will discuss as part of the next topological factor.

3.2 Knowledge distribution

Each agent possesses operators it can use to plan and must request the appropriate agent when it requires an operator it does not have. Therefore, the distribution of these operators is the next important topological factor. Using the original problem domain, the operators are divided according to the number of agents in the system. The complexity of this process is heavily domain dependent, affected both by the size and physics of the domain. For example, given a domain with six operators, there would be 5 possible sets of agent topologies, ranging from a 2 agent set to a 6 agent set. Each set, in turn, would have many possible topologies depending on the distribution of the operators. Using a segregated approach, with no knowledge overlap, the number of topologies in a 2 agent set would be the combination of all operator distributions (${}_6C_5$ + ${}_6C_4$ + ${}_6C_3$), or 61 possibilities. The sum of all sets in the domain would then be 483 possible topologies. If a non-segregated approach were taken in which operators were overlapped among agents, the number of possibilities would increase further. It is difficult to determine an optimal topology without enumerating all of these possibilities. However, general heuristics can be developed by initially comparing the performance of various types of topologies.

Goal loops. Due to the physics of a domain, certain distributions of operators may negatively impact performance. One such issue is the occurrence of goal loops. These arise in distributed multiagent systems when operators that depend on each other are divided among different agents.

A goal loop occurs in a planning problem when a precondition of a goal produces a subgoal that is already pending. Given a goal G that can be achieved by an operator OP1 with a precondition P1, if P1 produces a subgoal G' that requires an operator OP2 having the precondition P2 = G, then the resulting situation is a goal loop, because the achievement of G is still pending. PRODIGY can detect the goal loop if it exists within an agent, but if the operators exist in different agents, the goal loop will not be detected. Therefore, the first step to avoiding goal loops is to place these codependent operators in the same agent domain. However, if the operators are fully segregated and distributed, goal loops cannot be avoided in this way.

Agent Clones. Another similar factor that may hinder performance arises when an agent requests another agent when a request to that agent is already pending. For example, agent A does not have the necessary operator to plan part of a problem. Therefore, it makes a request to agent B, who has the operator, to plan this part. However, during its planning, agent B requires another operator it doesn't have. It must then request agent A to plan part of its sub-problem. Finally, agent A must make a second request back to agent B to achieve this goal it has received. However, because it is still waiting for a response from its original request to agent B, it cannot make a second request. This situation, referred to as a back-to-back request, is disallowed in order to prevent infinite goal loops.

However, a back-to-back request not part of a goal loop may be required to solve certain problems. This can be

addressed by creating a clone of one of the agents. A cloned agent is a specialized type of knowledge overlap in which two agents overlap by 100%, possessing identical sets of operators. When an agent attempts to make a second request to the same agent, it can instead request its clone. Once it receives a plan from the clone, execution can continue as necessary.

The use of cloned agents can be an effective way to increase the performance of a system. However, if there are a large number of agents, creating a clone for each one may be prohibitive. The best approach would be to create a clone dynamically, as it is needed within the system. This, too, may be difficult, depending on the implementation of the system. The final option would then be to create clones for only those agents that have the potential to receive sequential requests. Again, this is dependent on the domain, and may only be determined by examining the performance of the system using various topologies.

3.3 Initiating agents

The last important factor in an agent topology is the agent that initiates the planning process. In the original examinations of this system (Cox, Elahi, and Cleereman 2003, Elahi 2003), this factor was not initially taken into account. In this experiment, the starting agent was chosen arbitrarily for each topology. It was later discovered that planning the same problems with different starting agents yielded varying results. The number of problems solved and the amount of search varied depending on which agent initiated the planning. These findings demonstrate the effect the starting agent has on overall planning performance.

To begin the planning process, the problem must be passed to one of the agents in the system. This agent will attempt to plan the entire problem on its own, requesting the help of other agents as needed. Depending on the problem, the initiating agent may be able to begin the planning process, or it may need to immediately request another agent. Furthermore, the starting agent may not even be necessary to solve the problem. The selection of the starting agent will therefore affect problem-solving performance and the number of problems solved.

Before knowing if an agent can plan for a goal, it must choose an operator to solve this goal. If the agent does not possess the operator needed to begin planning, it will request the agent that does. The current state and modified goal are then passed to the next agent. This state and goal will be identical to the initial state and goal of the problem. Although no progress has been made, the planner has already expanded nodes in its search for a solution.

The selection of the starting agent will also affect the number of back-to-back request failures. If the initiating agent can only solve a small portion of the plan, it must pass the rest to another agent and planning will continue. The initiating agent must then wait until the rest of the plan is returned. Until this happens, the initiating agent cannot make a second request to that agent. Therefore, different starting agents will yield fewer failures than others and ultimately result in more problems solved than others.

4 The Logistics Domain

The logistics transportation domain (Veloso 1994) is a well-known benchmark domain in multiagent planning research. The basic elements of the domain are the infrastructure, the transportation units, and the packages. The infrastructure consists of a set of cities, and each city consists of a number of locations (e.g., post offices and airports). Trucks and airplanes are the transportation units: trucks can move freely within a city, but transport between cities can only be achieved by airplanes. The domain involves the movement of packages within cities by truck and between cities by airplane.

The logistics domain can be split into several sub-domains to fit in our distributed planning approach. There are six operators in the logistics domain: LOAD-TRUCK, UNLOAD-TRUCK, LOAD-AIRPLANE, UNLOAD-AIRPLANE, DRIVE-TRUCK, and FLY-AIRPLANE. To split the domain each operator is assigned to the agent one of the agents. If a goal requires operators from different subdomains, multiple planning agents must cooperate. Suppose an agent handles the moving of packages by truck within a city, and another agent handles the moving of packages by airplane between cities. If the goal is to transfer a package from the post-office in city1 to the airport in city2, then neither agent can achieve the goal on its own. The first agent must move the package to the airport in city1 and find a second agent to move it from there to the airport in city2. This intuitive explanation is implemented as a multiagent goal transformation.

5 Experimental Results

Experiments were performed in the logistics domain to demonstrate how different agent topologies affect performance in multiagent distributed planning. Several topologies were created by splitting the domain into multiple agent subdomains, varying from 2 to 6 agents. When possible, all of the agents possess an equal number of operators, and except for the use of clone agents, overlapping was not used in our examples. One set of topologies includes operator distributions designed to prevent goal loops, while another set of topologies was designed to allow the occurrence of goal loops. In the case of the logistics domain, the load-truck and unload-truck operators are codependent. Therefore, we placed these operators in separate agents to create goal loops. We then created a third topology by adding a clone agent to the first topology.

In the logistics domain, a problem may require the movement of a package within two cities. While only one plane would be needed to fly between the cities, two trucks would be required. Therefore, we created a clone of the agent or agents possessing the load and unload truck operators. For agent topologies with 2 to 5 agents, these operators were paired together, requiring one clone, while in the 6-agent topology, two clones were required.

We used a set of 100 randomly generated planning problems to test the performance of each topology. All of the problems contained three cities and a maximum of three goals. Performance was first measured by calculating the percentage of problems solved for each agent topology. Figure 1 shows the results our three sets of agents. Each topology was run using each possible initiating agent. The results for each initiating agent were then averaged to obtain a measure for that topology. The single-agent system is also included in our results. Because it is not a distributed topology, back-to-back requests and infinite goal loops are not a factor. The number of problems it solved is listed under both the non-goal loop and goal loop topologies, and the use of a clone would not be necessary for a single agent. The 6-agent topology is also a special case because there is only one way to distribute the operators. Therefore, its results will be the same for both the non-goal loop and goal loop topologies.

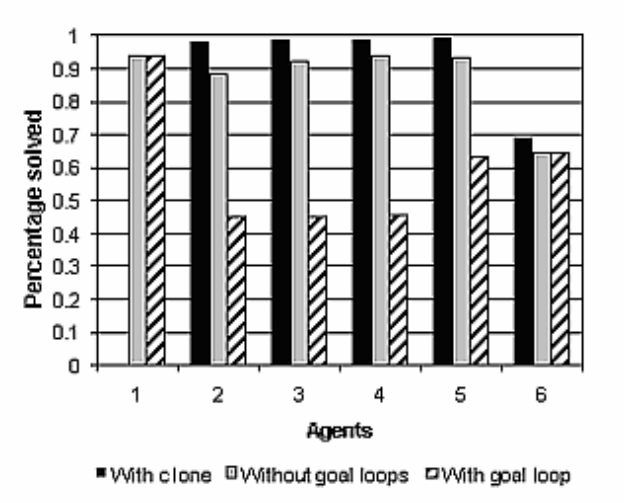


Figure 1: Percentage of problems solved by team topology

As we expected, the agent set with goal loops performed the worst, with only the 5 and 6 agent sets solving more than half the problems. The poor performance of the 2, 3, and 4 agent sets is due not only to the presence of goal loops, but also to back-to-back request failures. The further splitting of knowledge in the 5 and 6 agent sets reduces this type of failure. Because there are more agents, each agent possesses fewer operators. This makes it less likely that multiple requests will be made to the same agent.

The agent set with no goal loops solved the vast majority of problems, with performance peaking for the 4 and 5 agent topologies. This is because certain starting agents resulted in more back-to-back request failures. As more agents were added, these failures were minimized. The addition of a cloned agent to this topology eliminated all back-to-back request failures, resulting in excellent performance for the 2 to 5 agent sets. The use of a clone also improved problem-solving performance for the 6-agent topology, but because operators cannot be paired within the same agent, goal loops cannot be prevented.

Performance was also measured by the amount of search required to solve the problem. The average number of nodes expanded per agent was calculated for each problem, except for any problems that could not be solved by the system. The number of nodes expanded was then divided by the number of steps in the generated solution. Finally, this value was divided by the percentage of problems solved to normalize the results. Fewer nodes expanded means less search, thus a lower score is better. This method of measuring performance gives equal weight to the amount of search and the number of problems solved.

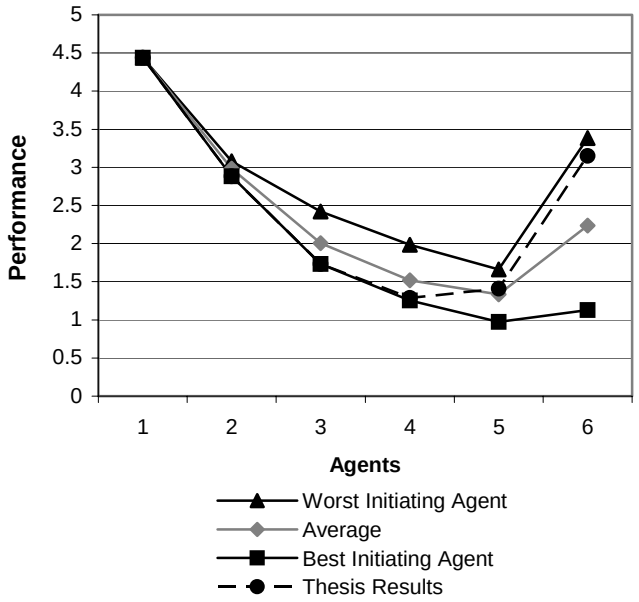


Figure 2: Planning performance as a function of team topology, non-goal-loop

Figure 2 shows the performance of topologies from 1 to 6 agents. These topologies were taken from the non-goal loop set, and compare performance using different initiating agents. The best and worst initiating agent for each is shown, as well as the average for all starting agents. We have also included the results compiled by Elahi (2003),

referred to as 'Thesis Results' in Figure 2. He employed a similar topology without goal loops, but chose the initiating agent arbitrarily. It can be seen that in all cases, the search performance improves as the number of agents increases, except for the fully distributed 6-agent topology. When the best starting agent is used, the decrease in performance for the 6-agent topology is minimized.

Finally, we compare the performance of the non-goal loop and goal loop topologies. Figure 3 shows the average planning performance when the results of both sets of topologies are combined. It also displays the combined results for the best and worst initiating agent for each topology. When both types of topologies are considered, the 4 and 5 agent sets performed the best over all possible initiating agents. If the best initiating agent is chosen, performance continues to improve slightly as the number of agents increases. The best agents in the goal loop topologies perform worse than those in the non-goal loop topologies, except for the 6-agent set, which is shared by both. Therefore, when the results are averaged, the 6-agent set yields the best performance. While there is an optimal performance that can be approached when using PRODIGY, there is no such upper bound. This is demonstrated in the results of the worst initiating agent. The occurrence of goal loops can yield inconsistent performance, shown most clearly in the 3-agent set.

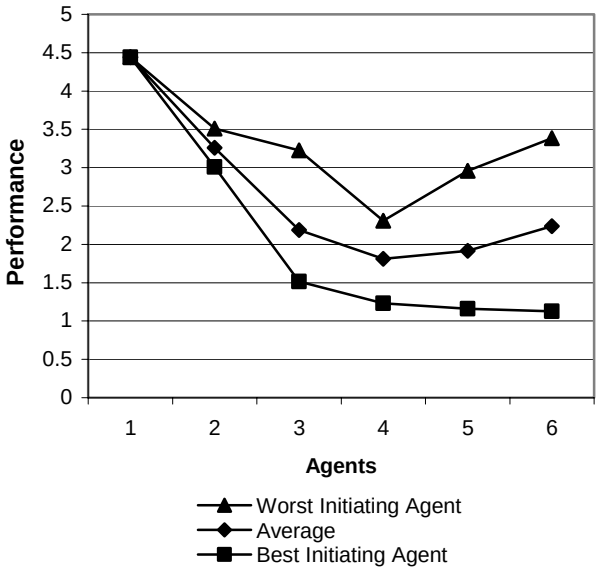


Figure 3: Planning performance as a function of team topology, non-goal-loop and goal loop combined

Both Figure 2 and 3 demonstrate the impact of the initiating agent on planning. The selection of this agent must be considered if optimal planning is to be achieved. By examining the performance of each initiating agent on a set of test problems, we can determine which agent in general will perform the best in the logistics domain.

6 Future Works

Using a relatively small set of topologies, we can see the effects their structures have on performance. To get a more complete idea of these effects, we would need to examine planning performance using all possible topologies within the logistics domain. This would require developing an automated method to split a domain into distributed domains. Furthermore, we would like to continue this work on larger domains, e.g. Extended STRIPS.

While topologies can be created that minimize the occurrence of goal loops, techniques need to be developed to better detect and correct for these occurrences. For example, a domain-independent method could be developed to determine what operators have the potential to create goal loops. When goal loops cannot be avoided, they would need to be excised. This would yield more relevant results for topologies in which goal loops prevent problems from being solved, such as a 6-agent topology in the logistics domain.

With more experimental results, we would also like to begin formulating procedures for designing optimal topologies for the domain examined. These procedures could then be generalized in a domain-independent manner. By examining the domain, the best topology could be selected without having to test its performance first.

7 Conclusions

In multiagent distributed planning, agents act autonomously according to the knowledge they possess. Structuring these agents and distributing knowledge among them is therefore crucial to the system's overall performance. We have discussed an approach used to coordinate and carry out distributed planning using multiple agents. Next, we have defined an agent topology to consist of three major factors: the number of agents in the system, the distribution of planning operators among them, and the agent initiating the planning. We then examined each factor and discussed how it affects planning. Finally, we have used an example to demonstrate empirically the impact the agent topology will have on performance.

Acknowledgements

This paper is supported by a grant from the Information Technology Research Institute (ITRI) at Wright State University. We also acknowledge Sylvia George for her feedback on our work.

References

Carbonell, J. G.; Blythe, J.; Etzioni, O.; Gil, Y.; Joseph, R.; Kahn, D.; Knoblock, C.; Minton, S.; Perez, A.; Reilly, S.; Veloso, M. M.; and Wang, X. 1992. *PRODIGY4.0: The Manual and Tutorial*, Technical Report, Computer Science Department, Carnegie Mellon University.

Cox, M. T.; Edwin, G.; Balasubramanian, K.; and Elahi, M. M. 2001. Multiagent Goal Transformation and Mixed-Initiative Planning Using Prodigy/Agent. In *Proceedings of* the *Fifth International Conference on Information, Systems, and Cybernetics*, Florida.

Cox, M. T.; Elahi, M.; and Cleereman, K. 2003. A distributed planning approach using multiagent goal transformations. In A. Ralescu (Ed.),Proceedings of the *14th Midwest Artificial Intelligence and Cognitive Science Conference*, 18-23. Cincinnati: Omnipress.

Cox, M. T.; and Veloso, M. M. 1998. Goal Transformations in Continuous Planning. In M. desJardins (Ed.), *Proceedings of the 1998 AAAI Fall Symposium on Distributed Continual Planning*, 23-30. Menlo Park, Calif.: AAAI Press / The MIT Press.

Durfee, E. H. 1999. Distributed Problem Solving and Planning. In G. Weiss (Ed.), *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, 121-164. Cambridge, Mass: The MIT Press.

Elahi, M. 2003. A Distributed Planning Approach Using Multiagent Goal Transformations, M.S. thesis, Department of Computer Science and Engineering, Wright State University

Sifting the Local Margins
-An Iterative Empirical Classification Scheme-

Dan Vance
ECECS Department
University of Cincinnati
Cincinnati OH 45221, USA
dvance@ececs.uc.edu

Anca L. Ralescu
ECECS Department
University of Cincinnati
Cincinnati OH 45221, USA
aralescu@ececs.uc.edu

Abstract

Attribute or feature selection is an important step in designing a classifier. It often reduces to choosing between computationally simple schemes (based on a small subset of attributes) which do not search the space and more complex schemes (large subset or entire set of available attributes) which are computationally intractable. Usually a compromise is reached: A computationally tractable scheme that relies on a subset of attributes that optimize a certain criterion is chosen. The result is usually a 'good' sub-optimal solution that may still require a fair amount of computation. This paper presents an approach which does not commit itself to any particular subset of the available attributes. Instead, the classifier uses each attribute successively as needed to classify a given data point. If the data set is separable in the given attribute space the algorithm will classify a given point with no errors. The resulting classifier is transparent, and the approach compares favorably with previous approaches both in accuracy and efficiency.

Keywords: classifiers, feature selection, decision trees, supervised learning.

1 Motivation for the Problem

A typical 2-class classification problem can be stated as follows, given a vector $X = [x_1, x_2, \ldots, x_n]$ classify it into one of two classes, based on the values of all or a subset of its components, $(x_i)_{i=1, \ldots, n}$.

In general, the components are called *attributes* and their values are called *features*. However, often these terms are used interchangeably.

A brute force algorithm that searches through all the subsets of $\{ x_i; i=1, \ldots, n\}$ in order to identify the best subset (a 'good' predictor for which the number of errors is minimized), has exponential complexity and therefore alternate approaches have been developed. A review, even concise of these approaches, exceeds the scope of this paper. It suffices to say that most often such approaches either identify a best subset of attributes, or construct new attributes from the given ones. The main idea underlying these methods is that of finding those attributes that can account for the classification for **all** (or nearly all) of the training set.

2 The Margin Algorithm

The work presented here departs from the previously described approach by allowing the **use of different attributes in different areas** of the feature space. Example 1 illustrates this approach on a small data set.

Example 1. Consider a simple artificial dataset: Class *A* consists of 34 points, Class *B* consists of 232 points. Each has a normal distribution: class *A* has $mean_A$ = (50.21, 52.73) and standard deviation s_A = (20.42, 23.12) and class *B* has $mean_B$ = (115.82, 111.73) and standard deviation s_B = (28.94, 27.79). In Figure 1, all the data points are plotted. An intuitive approach to separating the classes would be to find the mean of each cluster, create an axis connecting the two means, and find a line that crosses the line that *best* separates the clusters.

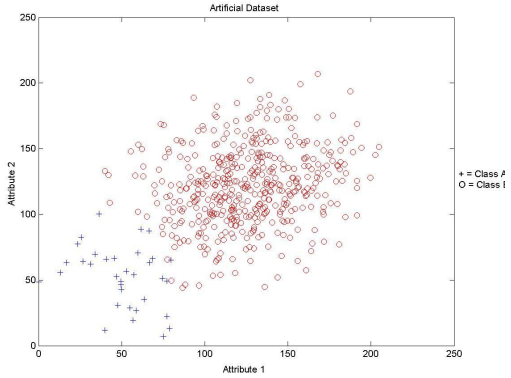


Figure 1: The data set for Example 1

In Figure 2, a vertical line, $x = x_1$, is drawn where x_1 is the maximum coordinate for points in *A* and another vertical line, $x = x_2$, is drawn where x_2 is the minimum *x* coordinate for points in *B*. The interval determined by these can be used to classify some of the points: those whose *x* coordinates fall to the left and right of the interval; however, the points with the *x* coordinate within the interval, are not classified using their *x* coordinates. The algorithm starts with $x_1 = \text{mean}_A$ and $x_2 = \text{mean}_B$. The lines are then *marched* towards the opposite mean until the shown final positions are reached. The final position forms

the *margin*, m_x, for this attribute. Points whose x coordinates fall in this margin cannot be classified by this attribute.

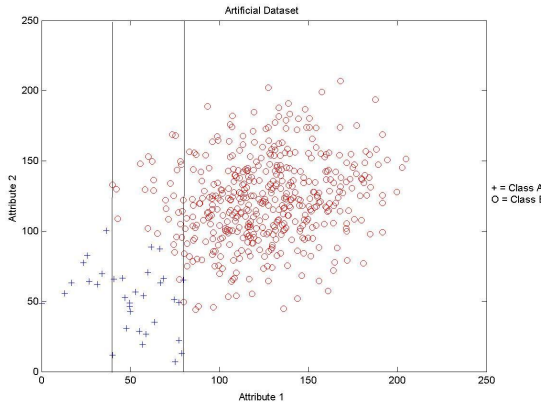


Figure 2: Margin for attribute x for the dataset in Example 1.

For these points a similar construction of a margin is done in the y attribute. In Figure 3, a horizontal line, $y = y_1$, is drawn where y_1 is the maximum y coordinate for the points from m_x that that are in A and another horizontal line, $y = y_2$, is draw, where y_2 is the minimum y coordinate for the points from m_x which are in B. The values y_1 and y_2 determine the margin, m_y, for the attribute y. The lines start at the means and are *marched* toward the opposite mean until the shown final positions are reached. The points with the x coordinates in m_x whose y coordinates fall outside m_y are classified either as A or B, while those whose y coordinates fall inside m_y cannot be classified (by this procedure).

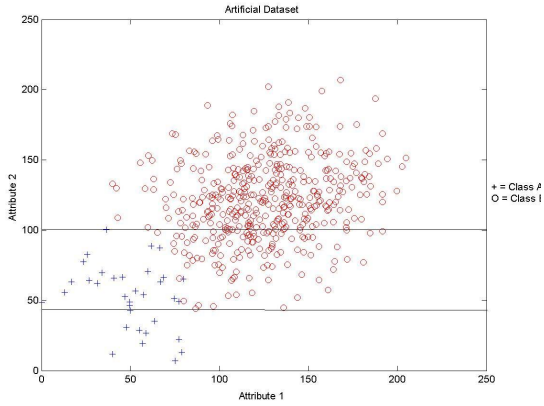


Figure 3: Margin for attribute y for the dataset in Example 1.

Figure 4 shows the combination of the results from figures 2 and 3. The points inside the rectangle are not classified; all other points can be classified correctly.

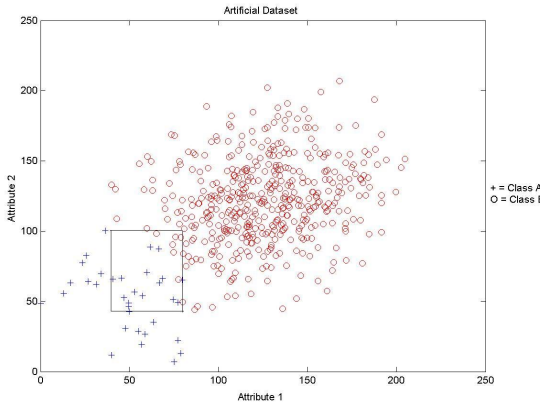


Figure 4: Combined margins for attributes x and y and for the dataset in Example 1.

Table 1 shows the results: few of the 266 points considered remain unclassified.

Table 1: Results for Example 1

	Class A	Class B	Total
Correct	22	228	250
Unclassified	12	6	18
% Correct	65%	97%	93%

Real datasets give a more interesting problem. There may be points that are outliers (or noise) are at the extremes of their distributions and are not in the overlap region. The decision is made *a priori* to accept these points as misclassified by the algorithm. In addition, there may be points such that if one stops *marching* the lines that form the margin at the first incorrect classification, one could miss large numbers of points that would be classified correctly.

The following remarks are useful at this point:

1. The lines are *marched* as before within 2 or 3 standard deviations from the means. This ensures that 98% or more of all data points are considered.
2. A fixed number of iterations is decided based on the value of increments of the standard deviation. For example, when this increment is 0.1, the number of iterations is 441. So the algorithm is deterministic.
3. The algorithm counts the number of correctly classified points.
4. The two standard deviations (one from each class), that give the highest accuracy of classification for each attribute are selected as final result. The dataset is classified by each attribute in turn, starting with the best classifier and so on.
5. In essence, the procedure truncates the extremes of each distribution in the area of overlap. Also some misclassification of the

training data are allowed in the hope of getting more points classified correctly.

As it can be seen from Example 1 when projected on a particular attribute, say x_1, the data points from the two classes may overlap. The region of overlap is called here *margin*. If the margin is known, then the data points that fall outside it can be classified exactly based on the values for the attribute x_1. Alternatively, it can be said that for the data points inside the margin, attribute x_1 is not useful. For these points the values of the attribute x_2 may be (and in the Example 1 are) considered for a complete classification: their margin (along x_2) is empty and the points are classified according to this margin. It is easily seen that the classification surface induced by such an algorithm is parallel to the coordinate axes (or, in higher dimensions, coordinate planes). It can also be seen that if the points are not separable by such surfaces (even though they may be separable, linearly or otherwise) the procedure outlined will not succeed in completely separating the training points and indeed may fail to separate **any** subset of these.

Therefore, in the remaining part of this paper it is assumed that the two classes are separable (to some extent) along at least some of the attributes and the algorithm aims at finding these attributes and the separating surfaces corresponding to them.

Intuitively, the concept of margin along attribute x_i is defined as the largest region in the domain for x_i which contains examples from the two classes. To express formally this concept the following notation is introduced:

Let x_i, $i = 1, \ldots, n$ denote the attributes for a two class classification problem. Let *TRAIN* denote the training set for this problem, that is *TRAIN* = $\{(x_1, \ldots, x_n, y)\}$ $y \in \{I, B\}\}$ where the values for y indicate the class (A or B) that the vector $(x_1, \ldots, x_n)$ belongs to. Let X_i denote the domain of the attribute x_i as represented in *TRAIN*. Let X^y_i the domain of the attribute x_i represented in class y. Obviously, $X_i = X^A_i \cup X^B_i$

Definition 1. The margin m_i along attribute x_i is defined as follows:

$$m_i \subseteq X^A_i \cap X^B_i$$

and

$$\forall S \subseteq X^A_i \cap X^B_i \text{ it follows that } S \subseteq m_i \quad (1)$$

Equation (1) states a property of maximality for the margin among all those subsets of the feature space that contain points from both classes. However, for practical purposes (in order to increase the number of correctly classified data points, at the expense of a fewer misclassified), this property is not always required.

2.1 Using the Margin along a Single Attribute

It can be seen that X_i can now be written as

$$X_i = (X^A_i \setminus X^B_i) \cup m_i \cup (X^B_i \setminus X^A_i)$$

Where \ denotes set difference, and therefore, knowledge of m_i can be used to correctly classify the data that do not fall within it.

Simple class complexity: The classification of data points is done as follows. The number of margins to be tried is deterministic. Each 'test' margin $m_i = [ai, b_i]$ for the attribute x_i is defined as follows:

$Max_A = m^i_A + \eta^j_A s^i_A$ (m = mean)
$Min_B = m^i_B - \eta^k_B s^i_B$ (s = standard deviation)

If $Max_A <= Max_B$
  $a_i = Max_A$
  $b_i = Min_B$ (2)
Else
  $a_i = Min_B$
  $b_i = Max_A$
Endif

In scanning the training data set in a given direction, all of the examples of class A appear to one side of the margin, followed by the margin, followed by all of the examples for class B (without loss of generality, assume that class A is to the left of class B). In this case, if $m_i = [a_i, b_i]$ and the scanning is *left-to-right* the class for the data point x is given by the rule:

$$\text{Class}(x) = \begin{cases} A & \text{if } x < a_i \\ B & \text{if } x > b_i \\ \text{none} & \text{otherwise} \end{cases} \quad (3)$$

The values x for which Class(x) = none are those which belong to the margin.

Increased class complexity For dataset $A = \{A_1 \cup A_2 \cup A_3 \cup \ldots \cup A_P\}$ with classes $A_1, A_2, A_3, \ldots, A_P$; an iterative approach to separating classes can be used.

Informally, rather than class A and class B, consider class A_1 and class B_1, where class $B_1 = \{A_2 \cup A_3 \cup \ldots \cup A_P\}$. Apply the algorithm to separate class A_1 from class B_1. Repeat the process by splitting B_1 as A_1 was split, and so on, until all classes are separated.

More formally, let $p_i = \cup^p_{j=1}[a_{ij}, b_{ij}]$ such that for each $j = 1, \ldots, p$ classes alternate. The approach can be reduced to the previous approach for binary classes by applying the classification rule iteratively in the space for the attribute x_i with the margin $[a_{ij}, b_{ij}]$, where $a_{ij} = \min\{a_{ij}; j = 1, \ldots, p\}$ and $b_{ij} = \max\{b_{ij}; j = 1, \ldots, p\}$.

The remaining part of this paper assumes simple class complexity.

2.2 Sifting the Margin

The quality of the margin determines the classification accuracy along the corresponding attribute. In practice the classifier has very little knowledge on the actual attribute values corresponding to each class. Therefore, the main part of deriving the classifier is to estimate the margins for each attribute. Informally, the procedure for learning the margin, starts with an estimate (around the class mean for training data) and iterates by updating these estimates; in these iterations more training data are allocated to their correct class.

More precisely, let m^i_A and m^i_B denote the means of the two classes as derived from TRAIN along attribute *i*. Again, without loss of generality assume that class *A* is to the left of class *B*, i.e., that $m^i_A < m^i_B$. Let s^i_A and s^i_B denote the standard deviation for class *A* and class *B*, respectively. In the current implementation, the margins are found for each of the attributes. Then the attributes are sorted by decreasing classification ability. The underlying class distribution is considered Normal (the Central Limit Theorem justifies this assumption for large classes). This means that with high probability the class values fall within two standard deviations from the class mean. The margin, $m_i = [a_i, b_i]$ for the *i*th attribute for the *j*th increment of η^j_A and the *k*th increment of η^k_B is updated according to the algorithm shown in Table 2.

The quantities η^j_A and η^k_B are the learning constants. Their values determine the accuracy and the speed of the convergence of the algorithm.

2.3 Using the Margin for Several Attributes

Let $X = \{x_1, \ldots, x_n\}$ denote the collection of attributes of interest (in general, this is the collection of attributes specified for all of the points in the training set, or alternatively all the attributes assuming that there are no missing attribute values). Further it is assumed that for all attributes the classes have simple complexity (or, alternatively, only that subset is considered for which this is true).

The iterative margin update procedure outlined in the previous sections for a single attribute can be extended to the set of attributes following the algorithm shown in Table 2.

Table 2: Iterative local margin pseudocode

```
TRAIN: the training data set
Initialize:
  max_correct = 0,   number_correct = 0

For i = 1 to k     (i = attribute number)
 For η^j_A = 0, …, 2 step 0.1  (A = class A)
  For η^k_B = 0, …, 2 step 0.1  (B = class B)
      For all x in the dataset,
        If x is correctly classified by
        rules (2) and (3)
           increment number_correct
        Endif
      Endfor
      If number_correct >max_correct
      then max_correct  = number_correct
      and m_i =   [a_i, b_i]
      Endif
  Endfor
 Endfor
Endfor
```

2.4 Computational Aspects

The complexity of the algorithm is determined by the complexity of the margin sifting portion and the complexity of the classification portion of the algorithm: The former is linear in the number of attributes and size of the training set, i.e., it is $O(C_1|X||TRAIN|)$ where the constant C_1 is determined by the step in the two 'for loops'. The latter is linear in the number of currently used attributes, that is, $O(C_2|A|)$ where $|X| \geq |A|$. In analyzing the algorithm using all the attributes two situations must be considered:

1. The two classes are separable to some extent along their attributes or a subset of these.
2. The two classes are not separable.

In case **(1.)** a more exact evaluation of complexity can be done by taking into account the size of the training subsets considered for each attribute and the probability of selecting, at each step, that attribute corresponding to a maximum reduction of this subset. Accuracy of the algorithm in this case does not depend on the order in which the attributes are considered, if there are no misclassifications, such as in carefully constructed artificial datasets. Since in fact this is not reasonable for real-world data, exploration of order of presentation of the attributes needs to be considered at a future date. It is expected that for a given attribute, the further apart the means of the 2 classes, and the tighter the clustering of each class, the closer one would come to this ideal artificial dataset. In short, we would expect to try a measure such as J3 to order the presentation of attributes to the algorithm.

In case **(2.)** the margin for any given attribute, that is the collection of training points not classified by the classification rule for that attribute, is not empty. In this case the order of attributes becomes more interesting and it is desired that attributes are considered in order of their classification accuracy or a measure such as J3.

Since, in general it is not known a priori that the classes are separable along attribute axes, attributes should be considered in increasing order of a measure calculated from the size of their domains and corresponding margin (e.g. smallest margin relative to the attribute domain).

3 Experimental Results

The approach outlined in the preceding sections has been tested on the artificial dataset described in Example 1 and two real datasets as follows:

Real Datasets

Two real datasets from the UC Irvine [1] repository are used to illustrate the performance of the algorithm.

The Wisconsin breast cancer dataset:

Of the 699 samples, 16 corresponding to missing attributes are removed (solely for simplicity of programming; the algorithm doesn't require their removal). The dataset is then split into a training set of approximately 1/3 the original number of points and the balance is used as

a test dataset. On a trial run the algorithm classifies the test data with 92.3% accuracy. Results over ten runs have 92.6% accuracy for the test set. Table 3 shows the results for each of these runs.

Table 3: Wisconsin Breast Cancer – 10 runs

Run #	% Correct of Test Data
1	92.3
2	93.0
3	93.0
4	93.0
5	91.5
6	91.9
7	92.1
8	92.3
9	93.4
10	93.4
Mean:	92.6

Pima Indians Diabetes dataset:

The Pima Indians Diabetes dataset is usually a difficult dataset to classify. It consists of 768 points which in the current approach are divided into a training set with 1/3 or 1/2 of the available data and the balance is used as a test dataset. On a trial run the algorithm classifies the test data with 73.6% accuracy, which compares favorably with other results in the literature. All points were classified, i.e., none were left in the margins. Note that the % classified correctly is slightly less than that of the best single classifier. On another run the test results are 75.7%, somewhat better than the single best classifier results of 71.1%. If the η^j_A and η^j_B for the single best classifier are used for each attribute (a 'global' approach), a slight decrease (0.5-2%) in accuracy occurs.

Table 4: Results of a typical training run for the Pima Indians Diabetes dataset

Attribute #	η^j_A	η^j_B	% Correct of Test Data
1	0.4	0	62.5
2	0.8	0.3	75.9
3	0	0	49.0
4	0.2	0	56.1
5	0.4	0.1	67.2
6	0.7	0	64.4
7	0.5	0	63.2
8	0.3	0.2	65.2

Figure 5 shows the result of averaging ten runs when the size of the training set varies from 10% to 70% of the entire dataset.

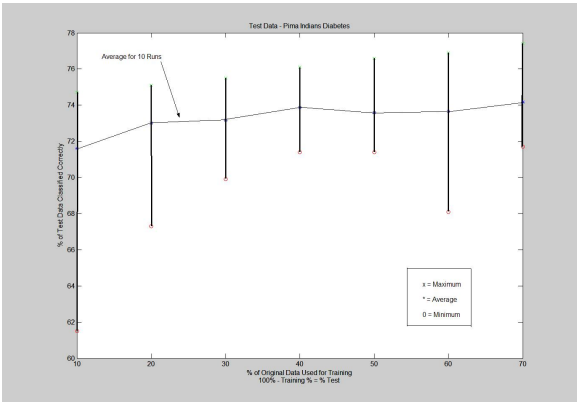


Figure 5: Results from averaging 10 runs for the Pima Indians Diabetes dataset.

In Table 5, results for 10 runs of training for Pima Indians Diabetes dataset is shown. On each of these runs approximately 1/2 of the entire dataset was used for training data and the other 1/2 as testing data.

Table 5: Typical results for the Pima Indians Diabetes data – 10 runs

Run #	% Correct of Test Data
1	71.4
2	73.7
3	74.7
4	75.8
5	74.2
6	75.3
7	73.4
8	75.3
9	75.3
10	73.7
Mean:	74.3

4 Comparison to other classifiers

As seen in Table 6, the Margin algorithm compares favorably with other methods. Pre-processing is not needed, as it is for some of these methods. Thus, these results are not a strict comparison.

Table 6: Accuracy: Margin vs. others

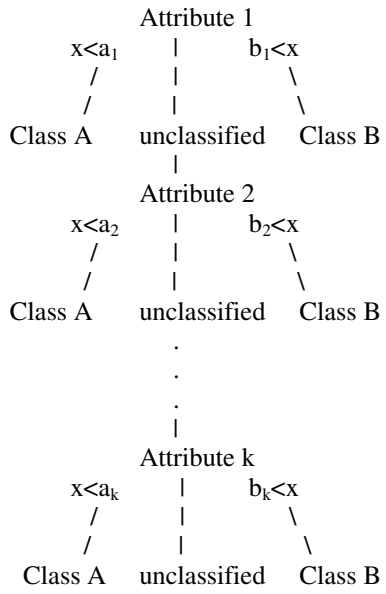
	Classifier	Pima Indians Diabetes	Wisconsin Breast Cancer
Neural Nets	Extracted Rules from Pruned Neural Nets [8]	-	97%
	Extracted Rules from Feedforward Networks [9]	-	97%
	Continuous Network [9]	-	97%
	Rprop [10]	76%	-
Genetic Programming	Forward Feed Neural Net in Genetic Programming [10]	77%	-
	Linear Genetic Programming [10]	76%	-
Linear Discriminant	Linear Discriminant (1) [11]	67%	-
Gaussian Process	Gaussian process (1) [11]	67%	-
SVM	SVM (1) [11]	64%	-
Bayesian Learning with Fuzzy Rules	Fuzzy Rules [12]	-	96%
Misc.	best of Statlog Project – compares 20 classification methods [13]	78%	-
	Bayesian estimator [13]	78%	96%
	LogitBoost [13]	-	97%
Ensemble Methods	best of Ensemble Methods [14]	74%	77%
Decision Trees	C4.5 [14], [6], [7]	68-71.5%	94-95.7%
	CVC4.5 [6]	72-73%	94%
	C4.5* (2) [6]	75%	96%
	CI2-2L [7]	70%	95%
	IB1 [7]	71%	96%
	Margin	**74%**	**93%**

(1) Reproduced in this paper from Seeger (2000).

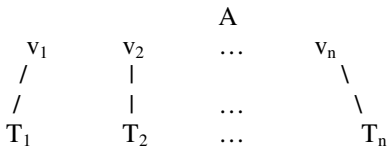
(2) Authors quote: "C4.5* results are optimistic upper bounds – we "cheated" and used the test set itself instead of cross-validation to pick the best value of the *m* parameter, then used the same test set to evaluate the induced tree."

Margin is a decision tree, as shown here:

Process of Margin:



Process of TDIDT:



A = 'Best' Attribute
v_i = A's values
T_i = subtrees

While at first glance, this may appear to be a Top-Down Inductive Decision Tree (TDIDT) algorithm, it is constructed in a different manner.

1. The criterion by which Margin's decision tree is constructed is not information theoretic as with ID3/C4.5, but, shall we say, more structural.
2. Margin uses the means and standard deviations to proceed through the data in stages of partitioning of data, while TDIDT uses statistical information to choose attributes directly.
3. Margin accomplishes most of the classification early in the 'sift' through the margin of each attribute. This is a very transparent approach: it can easily be seen why a particular attribute classified a point in class *A* or class *B*. TDIDT has a series of rules to make the classification

decision. TDIDT does allow a 'pruning', but it is not to be expected necessarily. Classification is assumed to be made after a series of decisions, thus later in general than by Margin.

These differences make the implementation of Margin very easy and straightforward compared to C4.5. Both C4.5 and Margin perform well on the Wisconsin Breast Cancer data (95% vs 93%) and the Pima Indians Diabetes data (72% vs 74%). Margin is quite efficient, being linear in n (the number of samples), both in training and testing the data.

5 Conclusions

An algorithm for finding classification surfaces parallel to the attribute axes has been proposed. This algorithm departs from the usual approach in which feature identification by allowing classification to be made according to different attributes in different regions of the feature space. Both the correctness of classification and the efficiency depend on the order in which the attributes are processed to some extent. Clearly the more points classified with 1 attribute leaves less processing for successive attributes. However, there is no guarantee that there is a best order of attributes with this particular filter, i.e., attributes are used in decreasing order of their classification ability by the 'margin'. Other filters need to be investigated.

A hypothesis that is currently being tested is that using a 'global' η_A and η_B for attribute $i = 1, \ldots, k$ would allow a tradeoff between the number correctly classified and the number incorrectly classified by any one attribute (and therefore not available for correct classification by successive attributes). This tradeoff may therefore give better classification performance. This can be accomplished without additional complexity. The 'local' current algorithm keeps a local 'best' η^i_A and η^i_B (the values for each attribute). By saving the global best (for all attributes) as the algorithm goes through the same exact steps, no complexity is added. This treats the set of points in a given class as 1 cluster and looks at the mean and standard deviation of this cluster, rather than the values for 1 attribute in a given class as 1 cluster and looks at the mean and standard deviation of a single attribute at a time. While the algorithm has been shown to work with these databases, a broader spectrum of databases needs to be tested.

Acknowledgements

Anca Ralescu's contribution to this work was partially supported by the grant N000140310706 from the Department of the Navy.

References

[1] C.J. Merz, P.M. Murphy (1998) UCI repository of machine learning databases. University of California, Irvine, Department of Information & Computer Sciences

[2] R. Caruana, V. R. de Sa (2003) "Benefitting from the Variables that Variable Selection Discards", Journal of Machine Learning Research, 2003, 1245-1264.

[3] I. Guyon, A. Elisseeff (2003) "An Introduction to Variable and Feature Selection", JMLR Special Issue, 2003, p 1157-1182

[4] H. Narazaki, A.L. Ralescu (1992) "Iterative Induction of a Category Membership Function", International Journal of Uncertainty, Fuzziness, and Knowledge-Based Systems, Vol. 2, No. 1, 1994, 91-100.

[5] H. Stoppiglia, G. Dreyfus, et al (2003) "Ranking a Random Variable for Variable and Feature Selection", JMLR Special Issue, 2003, p 1399-1414

[6] G. John (1994) "Cross-Validated C4.5: Using Error Estimation for Automatic Parameter Selection", Technical Report STAN-CS-TN-94-12, CS Department, Stanford University, October 1994

[7] Z. Zheng (1993) "A Benchmark for Classifier Learning", Technical Report 474, Proceedings of the 6th Australian Joint Conference on AI, p 281-286, World Scientific, 1993

[8] R. Setiono (1996) "Extracting Rules from Pruned Neural Networks for Breast Cancer Diagnosis", Artificial Intelligence in Medicine, Vol. 8, No. 1, February 1996, p 37-51

[9] J. Taha, J. Ghosh (1997) "Evaluation and Ordering of Rules Extracted from Forwardfeed Networks", Proceedings of the IEEE International Conference on Neural Networks, p 221-226, 1997

[10] A. Tsakonas, G. Dounias (2002) "A Scheme for the Evolution of Feedforward Networks using BNF-Grammar Driven Genetic Programming", Proceedings EUNITE-02. Algarve, Portugal, 2002

[11] W. Ju et al (2003) "On Bayesian Learning of Sparse Classifiers", September 2003

[12] C. Pina-Reyes, M. Sipper (1998) "Evolving Fuzzy Rules for Breast Cancer Diagnosis", Proceedings of 1998 International Symposium on Nonlinear Theory and Applications (NOLTA'98), Vol. 2, p 369-372, Lausanne, 1998

[13] P. Yau et al (2002) "Bayesian Variable Selection and Model Averaging in High Dimensional Multinomial Nonparametric Regression", Journal of Computational and Graphical Statistics, Vol. 12, No. 1, p23-32, March 2003

[14] D. Bahler, L. Navarro (2000) "Combining Heterogeneous Sets of Classifiers: Theoretical and Experimental Comparison of Methods", 17th National Conference on Artificial Intelligence (AAAI 2000), Workshop on New Research Problems for Machine Learning, 2000

[14] J. Aguilar, J Riquelme, M. Toro (2000) "Data Set Editing by Ordered Projection", 14th European Conference on Artificial Intelligence, August 2000

Genetic Algorithm Application to Clustering Problems

Mouin Hourani, Mufit Ozden, Frank Moore, and Pedrito Maynard-Zhang

Computer Science and Systems Analysis Department
Miami University, Oxford, OH 45056 USA
{houranm, ozdenm, moorefw, maynarp}@muohio.edu

Abstract

This study evaluates the efficacy of the continuous genetic algorithm in solving large-scale clustering problems. For these problems, thousands of entities must be grouped into a fixed, small number of clusters on the basis of similarity. Clustering of gene expressions reduces the unmanageable volume of data into a small number of sets. For this problem, the continuous genetic algorithm produces good results compared to alternative methods.

1 Introduction

The clustering problem is a search for natural groupings to simplify the description of a large set of multivariate data. Organizing data into sensible groupings is one of the most fundamental modes of human understanding and learning. Clustering is used in many diverse fields, including psychology, zoology, sociology, artificial intelligence, information retrieval, and botany.

For biologists, the availability of a nearly complete sequence of human genomes has resulted in a wealth of DNA sequence data. The stage has now been set for the next task, which is to identify the biological function of each gene in the sequence. Such knowledge will enable researchers to establish correspondence between genes and their functions in the cell, leading to therapeutic discoveries. A major difficulty is that the details of the phenomena taking place within cells are not fully understood. Identifying the phenomena and the genes involved and determining how the genes interact constitute an enormous challenge.

One approach to meeting this challenge is to identify groups of co-expressed genes. Clustering gene expression data into groups reduces the unmanageable volume of data into datasets that can be more easily handled. In our application, our goal is to determine the centroids of gene clusters in order to identify possible associations between the genes and their functions in cells ([1] 219-220). Our approach represents each gene with an expression profile (a time series of numerical values of its activities). We try to cluster genes in order to decide which genes exhibit similar behavior and understand the cause of a phenomenon. We assume that the number of clusters in this problem is given.

If we have *n* numerical values for each gene in its expression profile, the clustering problem will be *n*-dimensional. Every group has a centroid in this space. Positioning these centroids is our decision problem. Each gene is then assigned to the closest centroid. The optimal solution identifies a set of clusters with the minimum total distance between genes and their groups' centroids.

Genetic algorithms (GAs) are search techniques inspired by the biological process of evolution via natural selection. GAs can be used to construct numerical optimization techniques that perform effectively on problems characterized by huge search spaces. This study uses the continuous variable GA strategy to find the optimal solution for the clustering problem. This problem is particularly complex due to the large number of locally optimal solutions. The GA may get stuck in one of these locally optimal solutions instead of finding the globally optimal solution.

2 Continuous GAs

In each cycle, the standard GA [2] first determines the fitness of each candidate solution. Next, we perform selection, whereby a temporary population is created in which the fittest individuals (those corresponding to the best solutions contained in the current population) are likely to have a higher number of instances than less fit individuals. Crossover and mutation operators are then applied to the individuals in the temporary population. Finally, the temporary population replaces the current population. The GA repeats this process until a termination criterion is satisfied. GAs do not guarantee an optimal solution. The behavior of GAs is stochastic: different runs of the same algorithm may potentially result in different

solutions. For this reason, it is common to perform several independent runs when studying a particular problem, and retain the best solution. Selection chooses chromosomes to be used as parents for the next generation, according to their fitness values. Chromosomes with higher fitness values have a higher probability of contributing offspring to the next generation. The steady state selection mechanism used in this study sorts chromosomes from the population according to their fitness values. Chromosomes exhibiting higher fitness are then selected as parents, with the same probability for creating new offspring, while chromosomes with lower fitness values do not qualify to be parents.

The single point crossover used in this study selects a random crossover point within the two parent chromosomes, and then exchanges the genes at or below this point from each of the parent chromosomes to produce two new offspring. Mutation takes place after crossover is performed. Mutation randomly changes a selected gene in the selected offspring. The primary purpose of mutation is to prevent premature convergence to a locally optimal solution [5]. The value mutation for continuous GAs used in this study sets an entire gene from the selected chromosome to a new random value.

Continuous value encoding [5] can be used in problems where decision variables require more complex values such as real numbers. Every chromosome contains a sequence of these values. For clustering problems, each chromosome is represented by a string. For the example shown in Fig. 1, a chromosome contains 10 variables. Each variable length is 5, and the overall chromosome length is 50. To encode the first variable, we append the first five digits from the string (in this example, "13463") to the string "0.", resulting in the number 0.13463. All variables thus represented have values in the range [0,0.99999]. We can easily scale this range to any range we want.

3 The Clustering Problem and Gene Expression

The problem of clustering points in a multidimensional space is a combinatorial optimization problem where the goal is to partition a set of data objects into a predefined number of clusters in such a way that similar objects are grouped together while the dissimilar objects are put in separate groups [4]. The difficulty of clustering stems from the large number of unique partitions that may exist in a given search space. This number is formally defined as follows: given N data points in a D dimensional metric space, determine a partition of the patterns into K clusters, such that the number of possible of distinct partitions will be ([3] 90-91):

$$S(N, K) = S(N-1, K-1) + K\,S(N-1, K)$$

The boundary conditions on this equation are:

$$S(N, 1) = 1,\ S(N, N) = 1,\ S(N, K) = 0 \text{ if } K > N$$

The solution to the partial difference equation is called a Stirling number of second kind:

$$S(N, K) = \frac{1}{K!}\sum_{i=1}^{K}(-1)^{K-i}\binom{K}{i}(i)^{N}$$

For example, there are only 34,105 distinct partitions of 10 objects into four clusters, but this number explodes to approximately 11,259,666,000 if 19 objects are to be partitioned into four clusters. Clearly, an exhaustive search of all possible partitions is not computationally feasible, even for a small number of patterns.

3.1 K-means Method of Clustering

K-means clustering (Fig. 2) generates a specific number of disjoint, flat (non-hierarchical) clusters [6]. Frequently used for large problems, the K-means algorithm incorporates the following assumptions:

1. There are always K clusters.
2. There is always at least one item in each cluster.
3. The clusters are non-hierarchical and they do not overlap.
4. Every member of a cluster is closer to its cluster's centroid than to any other cluster's centroid.

13463	69785	32572	47681	02184	12443	49284	82572	25681	42184	.....
0.13463	0.69785	0.32572	0.47681	0.02184	0.12443	0.49284	0.82572	0.25681	0.42184	.....

Fig. 1. An example chromosome for a continuous GA

Partition dataset into **K** clusters according to some *ad hoc* procedure
DO
  Centroids ß means of the clusters
  FOR each data point
    FOR i=0 to **K**
      $\mathbf{d_i}$ ß **distance(**Centroid$_i$, point**)**
    END FOR
    Move point to the cluster that has the smallest $\mathbf{d_i}$
  END FOR
WHILE at least one point moved to another cluster
RETURN Centroids

Fig. 2: K-means Algorithm

Initially, the dataset is partitioned into K clusters according to some *ad hoc* procedure (e.g., picking K random data points as the centroids and assigning points to the closest centroids). Each centroid is updated to the mean of the points in its cluster. Each point is then clustered to the nearest centroid. This procedure is repeated until no data point is moved from one cluster to another. At this point, the clusters are stable and the clustering process ends. The K-means algorithm gives good results only when the initial partitioning is close to the optimal solution [3]. Fig. 3 shows how the K-means clustering algorithm works with points in 2-dimensional space ([1] 222).

3.2 GA Formulation of the Clustering Problem

This study applied GAs in four different ways to solve the clustering problem. Our test problem created a specified number of data points with a fixed number of attributes using Gaussian distributions around preset centroids. To facilitate comparisons between these applications, the clusters of this data were designed to be easily separated.

Each algorithm tries to group ***N*** points in a ***D*** dimensional metric space into ***K*** clusters. Every cluster has a centroid in this space. Finding these centroids is our decision problem. Each point should be assigned to the closest centroid. To get an optimal solution, we need to find a set of ***K*** cluster centroids with a minimum total distance to their clustering points. One of the most important choices in the clustering method is the objective function that evaluates the quality of clustering. A commonly used objective criterion is to minimize the sum of squared distance of each point to its cluster's centroid. The distance between a point and its cluster's centroid is measured by the Euclidean distance of the two feature vectors, which is calculated as ([3] 15):

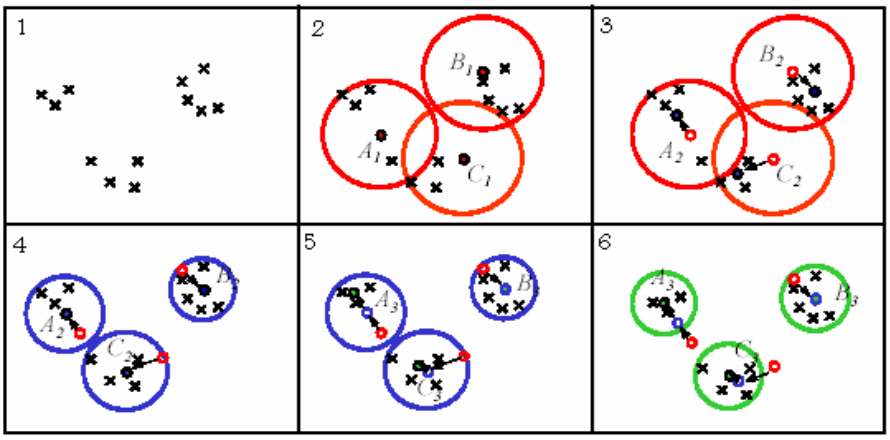


Fig. 3: An Example of K-means

$$d(p,c) = \sqrt{\sum_{d=1}^{D} (p_d - c_d)^2}$$

where p is a point and c is a centroid in D dimensional space. The fitness value for a GA chromosome is the mean squared error, which is the total distance between each pattern and the nearest centroid:

$$\text{Fitness value} = \frac{1}{ND} \sum_{i=1}^{N} d(p_i, c_p)^2$$

where ***N*** is number of data points.

3.3 Running GAs with the Clustering Test Data

1,500 points in a 5 dimensional metric space were generated to form 15 clusters, each containing 100 points. First, the centroids of the clusters were randomly created with the condition that the distance between any pair be at least 0.79. In this five-dimensional space, the diagonal distance is:

$$\sqrt{1^2 + 1^2 + 1^2 + 1^2 + 1^2} = 2.236$$

We used a Gaussian distribution with a standard deviation chosen randomly from a range [0.02, 0.04] to generate 100 points within the unit metric space for each of the centroids. Fig. 4 shows the centers of 1500 points in a 2-dimensional projection.

To determine the most effective GA for the clustering problem, a series of four GAs were applied to the test data. Each GA in the series enhances the previous approach.

3.3.1 Approach 1: Standard GA

First, a standard GA, designated GA1, was applied without any adaptation. GA1 was run with the following parameters:

- Population size = 100
- Selection method = Steady State Selection with 25% selection rate
- Crossover method = Simple point crossover with 65% crossover rate
- Mutation method = Value Mutation with 2% mutation rate
- Termination Criterion = Best Value Didn't Changed For 50 Generation

Number of runs = 10 with different initial generations. Table 2 illustrates the results. Actual centroids represent the means of the Gaussian distributions used to generate the points. The fitness value of the actual centroids for this data set is 0.003429.

3.3.2 Approach 2: Reordering Centroids before Crossover

When running GA1 to solve the clustering problem, each chromosome may have a different centroid order. That is, the naming of the clusters may be inconsistent from one solution to another. To accelerate convergence, GA1 finds associations among the cluster names and orders them accordingly in the chromosomes before crossover. Fig. 5 depicts the difficulty unless this association is made before crossover. In Fig. 5.a, when the clusters C1 and C2 in parent 1 are forced to crossover with C2 and C1 in Parent 2, respectively, the offspring will exhibit worse fitness than both of the parents. The fitness value of a solution stems from the positions of the centroids, rather than this naming order. Fig. 5.b shows the improvement derived from establishing the associated naming of the centroids in the parents based on the closeness of the centroids prior to crossover. We modified GA1 for clustering with the ordering of the centroids as described above, and ran the resulting program, GA2, on the same problem described in the previous section. Table 3 shows the new results. By comparing the time in Tables 2 and 3 with a 95% confidence interval, we have found that running time decreases after applying centroid reordering method. The results are:

Davg = 16.2; S^2 = 449.96; alpha = 0.05;

$t_{0.05/2,9}$ = 2.26; hw = 16.00; lo = 0.20;

up = 32.20

Since both lo and up have the same sign, we can say that GA2 is faster than GA1 with 95% confidence. GA2 is 1.3 times faster than GA1 on average. At the same time, GA2 is more accurate than GA1: the average percentage of correctly assigned points increased by 2% from 96.2% to 98.2%.

3.3.3 Approach 3: Approach 2 + the GA Applied on Sample Data

Clustering data usually contains many points that have close values. In this case, the GA may use only a sample of the data to find the clusters. Sampling may be the only feasible option for extremely large clustering problems. Our third approach, GA3, randomly selected a subset of data points. GA3 finds the optimal clusters for this subset, and then assigns the rest of the data points to those clusters. A final evaluation was done for the GA3 clusters over the entire

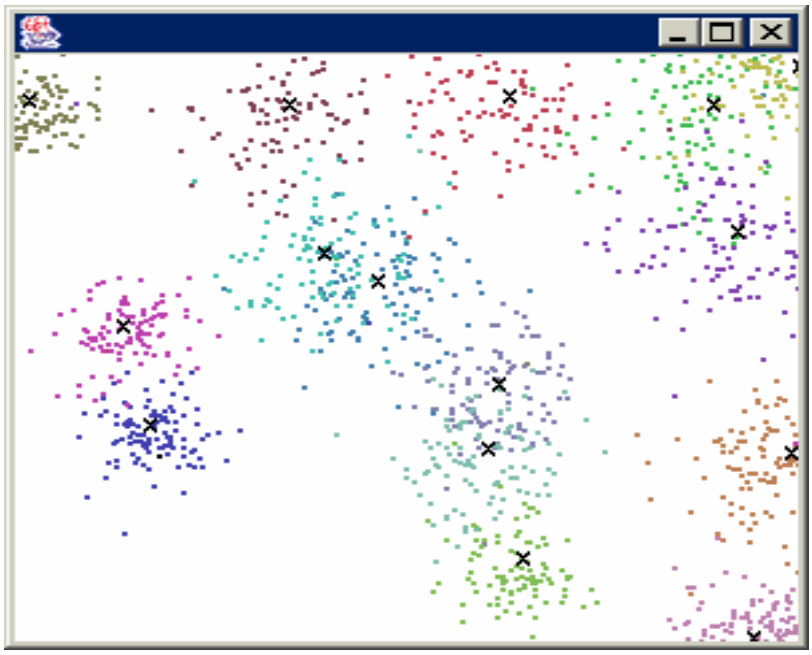


Fig. 4: Data Points with Fifteen Clusters Projected onto 2D

Table 2: Clustering Results after Applying a Standard GA (GA1)

Run Number	Time in Minutes	Number Of Generations	Solution Fitness	Solution Fitness Divided by Fitness of Actual Centroids	% of Correctly Assigned Points	Mean Squared Error between Calculated and Actual Centroids
1	98	1182	0.006341	1.85	90.6%	0.00419
2	55	1101	0.006342	1.85	90.6%	0.00587
3	73	1455	0.003799	1.11	100.0%	0.00011
4	70	1384	0.003641	1.06	100.0%	0.00010
5	76	1515	0.003535	1.03	100.0%	0.00010
6	52	1039	0.003807	1.11	100.0%	0.00013
7	64	1283	0.003829	1.12	100.0%	0.00015
8	90	1467	0.006721	1.96	90.8%	0.00299
9	72	1268	0.003748	1.09	100.0%	0.00012
10	87	1489	0.006609	1.93	90.4%	0.00435
Average	74	1318	0.004837	1.41	96.2%	0.00181
STDEV	15	170	0.001441	0.42	4.9%	0.00229
Max	98	1515	0.006721	1.96	100.0%	0.00587
Min	52	1039	0.003535	1.03	90.4%	0.00010

data set. This study used three sampling rates: 20%, 10% and 5%. As shown by Tables 3, 4, 5 and 6, decreasing the sampling rate reduces the average run time. Fig. 6 illustrates the linear relationship between the sampling rate and the average run time. Decreasing the sampling rate also causes a loss of accuracy: Fig. 7 shows the relationship between the average percentage of correctly assigned points and the sampling rate.

We used the 10% sampling percentage to compare GA3 with GA2. By comparing the time in Tables 3 and 5 with a 95% confidence interval, we have found that run time decreases after applying GA3 on a sample of data. The results are:

Davg = 49.77; S^2 = 189.17; alpha = 0.05;

$t_{0.05/2,9}$ = 2.26; hw = 10.37; lo = 39.40; up = 60.14.

Since both lo and up have the same sign, we can say that GA3 is faster than GA2 with95% confidence. GA3 is 7.44 times faster than GA2 on average. At the same time, GA2 is more accurate than GA3: the average percentage of correctly assigned points decreased from 98.2% to 96.53%. This slight loss of accuracy is reasonable compared to the amount of time saved using GA3.

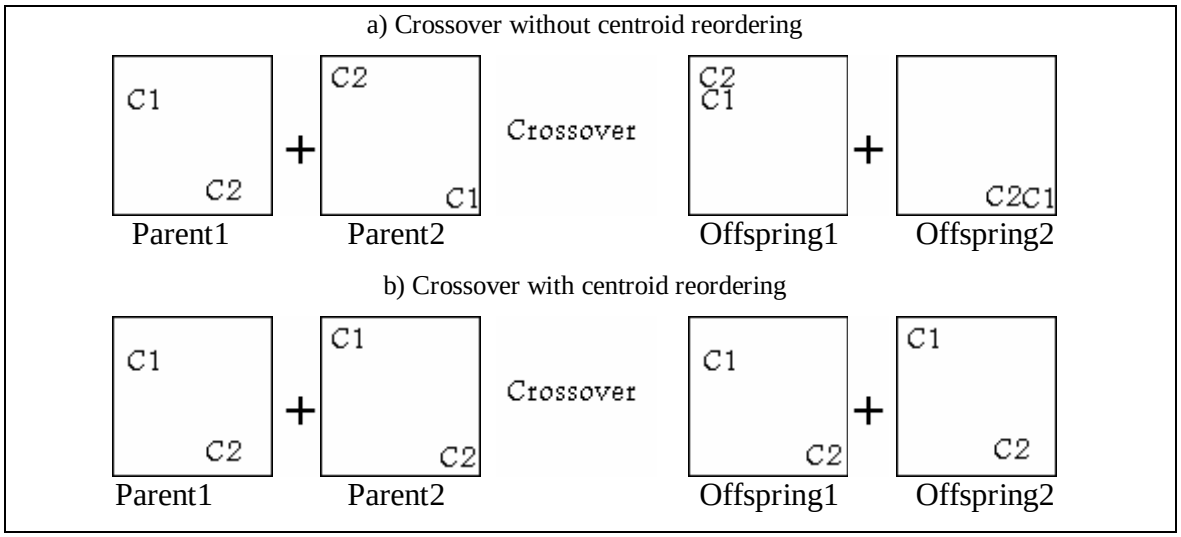


Fig. 5: An Example showing Crossover without/with Centroid Reordering

Table 3: Clustering Results after Reordering Centroids in the GA (GA2)

Run Number	Time in Minutes	Number Of Generations	Solution Fitness	Solution Fitness Divided by Fitness of Actual Centroids	% of Correctly Assigned Points	Mean Squared Error between Calculated and Actual Centroids
1	55	1092	0.00392	1.14	100%	0.00017
2	54	1060	0.00366	1.07	100%	0.00012
3	52	1035	0.00409	1.19	100%	0.00020
4	84	1695	0.00350	1.02	100%	0.00009
5	60	1181	0.00616	1.79	92%	0.00569
6	56	1105	0.00382	1.11	100%	0.00016
7	72	1420	0.00352	1.03	100%	0.00011
8	53	1042	0.00382	1.11	100%	0.00016
9	33	647	0.00730	2.13	90%	0.00323
10	56	1105	0.00360	1.05	100%	0.00010
Average	57	1138	0.00434	1.26	98.2%	0.00100
STDEV	13	271.4	0.00130	0.38	3.82%	0.00191
Max	84	1695	0.00730	2.13	100%	0.00569
Min	33	647	0.00350	1.02	90%	0.00009

Table 4: Clustering Results after Applying GA3 on 20% a Sample Data.

Run Number	Time in Minutes	Number Of Generations	Solution Fitness	Solution Fitness Divided by Fitness of Actual Centroids	% of Correctly Assigned Points	Mean Squared Error between Calculated and Actual Centroids
1	7.06	588	0.00754	1.09	92.20%	0.00323
2	11.89	1055	0.00397	1.41	100.00%	0.00018
3	12.38	1081	0.00385	1.21	100.00%	0.00016
4	13.38	1180	0.00382	1.08	100.00%	0.00013
5	15.92	1402	0.00376	1.12	100.00%	0.00014
6	14.93	1309	0.00386	2.05	100.00%	0.00014
7	12.01	1052	0.00673	2.23	90.60%	0.00436
8	9.60	842	0.00722	1.99	89.93%	0.00233
9	15.41	1352	0.00871	1.14	92.53%	0.00531
10	12.87	1129	0.00433	1.13	100.00%	0.00028
Average	12.55	1099	0.00538	1.45	96.53%	0.00163
STDEV	2.56	232	0.00184	0.44	4.31%	0.00192
Max	15.92	1402	0.00871	2.23	100.00%	0.00531
Min	7.06	588	0.00376	1.08	89.93%	0.00013

Table 5: Clustering Results after Applying GA3 on 10% a Sample Data.

Run Number	Time in Minutes	Number Of Generations	Solution Fitness	Solution Fitness Divided by Fitness of actual Centroids	% of Correctly Assigned Points	Mean Squared Error between Calculated and actual Centroids
1	7.22	1116	0.00742	2.16	91.67%	0.00191
2	4.41	680	0.00901	2.63	92.07%	0.00369
3	6.61	910	0.00788	2.30	91.87%	0.00478
4	5.30	783	0.00523	1.52	100.00%	0.00047
5	11.35	1713	0.00379	1.10	100.00%	0.00014
6	8.35	1261	0.00409	1.19	100.00%	0.00021
7	9.41	1377	0.00720	2.10	92.93%	0.00308
8	10.09	1463	0.00372	1.08	100.00%	0.00013
9	8.24	1218	0.00483	1.41	100.00%	0.00037
10	6.33	900	0.00959	2.80	93.27%	0.00340
Average	7.73	1142	0.00627	1.83	96.18%	0.00182
STDEV	2.06	310	0.00209	0.61	3.85%	0.00169
Max	11.35	1713	0.00959	2.80	100.00%	0.00478
Min	4.41	680	0.00372	1.08	91.67%	0.00013

3.3.4 Approach 4: Approach 3(10%) + Recalculating Centroids

Our fourth approach, GA4, fine tunes the new offspring after crossover and mutation. GA4 recalculates the centroids after the formation of clusters. After all data points are assigned to the closest centroids in the chromosome, the centroids are moved to the actual centers of the clusters. Using GA4, we obtained the result shown in Table 7. By comparing the time in Table 5 and Table 7 with a 95% confidence interval, we found that running time decreases when centroids are recalculated. The results are:

Davg = 6.87; S2 = 4.80; alpha = 0.05;

$t_{0.05/2,9}$ = 2.26; hw = 1.65; lo = 5.21; up = 8.52.

Since both lo and up have the same sign, we can say that GA4 is faster than GA3 (10% sampling percentage) with 95% confidence. GA4 is faster than GA3 by 8.9 times in the average. At the

same time, GA4 is more accurate than GA3: the average percentage of correctly assigned points increased from 96.18% to 100%. Even though additional calculations are performed by GA4, the total running time is less than GA3. GA4 decreases the search space, allowing it to find the solution in fewer generations. From the Tables 2, 3, 5, and 7, we found that GA4 gave the best results in terms of time performance and accuracy of the cluster assignments (Fig. 8).

3.4 Comparison of the Results of our GA and K-Means on the Clustering Test Data

We applied the K-means method described in section 3.2.2 to the test data, and compared its results those of GA4. The results of the K-means method are shown in Table 8. By comparing the percentage of correctly assigned points in Tables 7 and 8 with a 95% confidence interval, we have found that percentage of points correctly assigned by GA4 is higher than the percentage found using the K-means method. This result shows that GA4 performs better than K-means:

Davg = 0.17; S2 = 0.01; alpha = 0.05;

$t_{0.05/2,9}$ = 2.26; hw = 0.06; lo = 0.11; up = 0.220.

Table 6: Clustering Results after Applying GA3 on 5% a Sample Data.

Run Number	Time in Minutes	Number Of Generations	Solution Fitness	Solution Fitness Divided by Fitness of Actual Centroids	% of Correctly Assigned Points	Mean Squared Error between Calculated and Actual Centroids
1	5.80	1249	0.00405	1.18	100.00%	0.00017
2	6.20	1446	0.00461	1.34	100.00%	0.00032
3	3.55	843	0.00865	2.52	90.20%	0.00300
4	4.94	1227	0.00742	2.16	92.40%	0.00440
5	3.91	961	0.01035	3.02	83.13%	0.00459
6	3.33	804	0.00871	2.54	93.33%	0.00490
7	4.93	1189	0.00483	1.41	100.00%	0.00035
8	4.58	932	0.00791	2.31	89.80%	0.00274
9	4.69	998	0.00814	2.37	93.33%	0.00495
10	2.58	592	0.00905	2.64	90.20%	0.00262
Average	4.45	1024	0.00737	2.15	93.24%	0.00280
STDEV	1.06	240	0.00203	0.59	5.20%	0.00184
Max	6.20	1446	0.01035	3.02	100.00%	0.00495
Min	2.58	592	0.00405	1.18	83.13%	0.00017

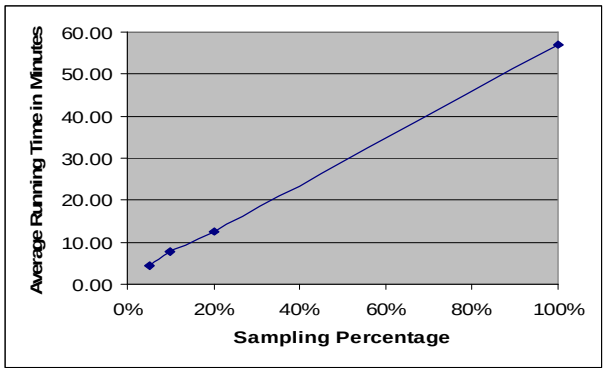


Fig. 6: Relationship between Average Running Time and Sampling Percentage (GA3)

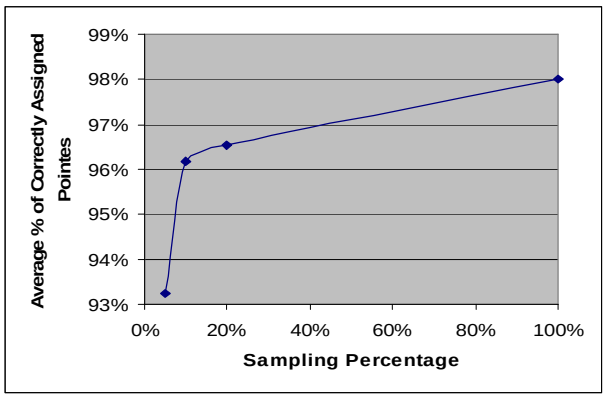


Fig. 7: Relationship between Average Percentage of Correctly Assigned Points and Sampling Percentage (GA3)

Table 7: Clustering Results after Applying the GA with Recalculating Centroid

Run Number	Time in Minutes	Number Of Generations	Solution Fitness	Solution Fitness Divided by Fitness of Actual Centroids	% of Correctly Assigned Points	Mean Squared Error between Calculated and Actual Centroids
1	0.91	53	0.00352	1.03	100.00%	0.00009
2	0.93	52	0.00367	1.07	100.00%	0.00011
3	0.87	53	0.00363	1.06	100.00%	0.00010
4	0.88	54	0.00366	1.07	100.00%	0.00011
5	0.86	53	0.00356	1.04	100.00%	0.00009
6	0.84	53	0.00364	1.06	100.00%	0.00009
7	0.82	53	0.00354	1.03	100.00%	0.00009
8	0.87	53	0.00354	1.03	100.00%	0.00009
9	0.83	53	0.00352	1.03	100.00%	0.00007
10	0.83	53	0.00364	1.06	100.00%	0.00011
Average	0.86	53	0.00359	1.05	100.00%	0.00010
STDEV	0.03	0.4	0.00006	0.02	0.00%	0.00001
Max	0.93	54	0.00367	1.07	100.00%	0.00011
Min	0.82	52	0.00352	1.03	100.00%	0.00007

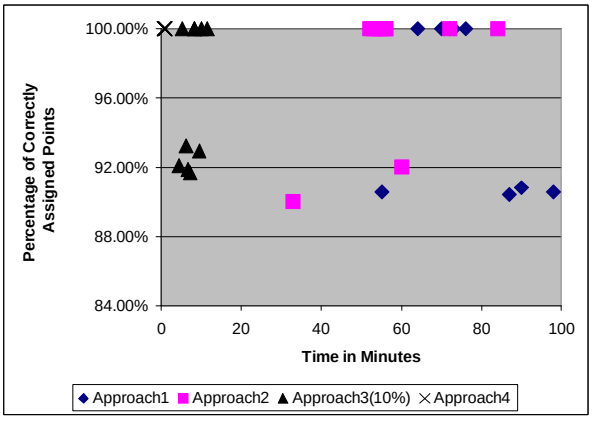


Fig. 8: Run Times vs. Percentage of Correctly Assigned Points

Table 8: K-means Clustering Results for Test Data

Run Number	Time in Minutes	Number Of Cycles	Solution Fitness	Solution Fitness Divided by Fitness of Actual Centroids	% of Correctly Assigned Points	Mean Squared Error between Calculated and Actual Centroids
1	0.017	12	0.00734	2.14	90.87%	0.00217
2	0.021	15	0.00921	2.69	81.13%	0.00458
3	0.017	14	0.00629	1.83	90.93%	0.00195
4	0.012	12	0.01225	3.57	73.40%	0.01195
5	0.014	13	0.00615	1.79	89.87%	0.00309
6	0.013	12	0.01141	3.33	83.33%	0.00722
7	0.016	15	0.00938	2.73	83.33%	0.00617
8	0.011	11	0.00667	1.95	89.60%	0.00295
9	0.009	9	0.00964	2.81	82.67%	0.00583
10	0.011	11	0.01582	4.61	68.07%	0.01049
Average	0.014	12	0.00942	2.75	83.32%	0.00564
STDEV	0.003	2	0.00292	0.85	7.30%	0.00327
Max	0.021	15	0.01582	4.61	90.93%	0.01195
Min	0.009	9	0.00615	1.79	68.07%	0.00195

Since both lo and up have the same sign, we can say that GA4 performs better than K-Means with 95% confidence. Related research (e.g., [7]) supports this conclusion. The average percentage of correctly assigned points for GA4 is 100% while the average percentage for K-means is 83.22%. However, K-means is faster than GA4: the average run time for K-means is 0.014 minutes while for GA4 it is 0.86 minutes. If time is more important than accuracy, we can use K-means method. Otherwise, GA4 will better suit our purpose.

4 Conclusions

This study investigated the use of GAs for continuous variable optimization in the clustering problem. To overcome the enormous computational complexity of this task, we implemented several innovative techniques, including centroid reordering, sampling, and centroid relocation. Each of these innovations resulted in significant performance enhancements.

We compared the results of our most effective algorithm, GA4, to those of the K-means method, which is a standard approach to solving large clustering problems. We estimated that GA4 produced better results than the K-means algorithm with 95% confidence. Further improvement may come by reducing the number of dimensions and the number of points used to determine the centroids, without changing the gene expression similarities. We believe the potential for additional progress in this area of research is strong.

5 References

[1] Fogel, G. and D. Corne 2002. *Evolutionary Computation in Bioinformatics*, Morgan Kaufmann.

[2] Goldberg D. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Company.

[3] Jain, A. and R. Dubes 1988. *Algorithms for Clustering Data*, NJ: Englewood Cliffs, Prentice Hall College Div.

[4] Kivijarvi, J., P. Franti, and O. Nevalainen 2003. Self-Adaptive Genetic Algorithm for Clustering, *Journal of Heuristics* 9: 113-129.

[5] Obitko, M. 1998. Introduction to Genetic Algorithms, Prague University of Applied Sciences. See <http://cs/felk.cvut.cz/~xobitko/ga/>.

[6] Predictive Patterns Software 2003. K-Means Clustering Overview, Kingston, Canada. See <http://www.predictivepatterns.com/docs/WebSiteDocs/Clustering/KMeans_Clustering_Overview.htm>.

[7] Werner, J. and T. Fogarty 2002. Genetic Algorithm in Clustering Data Sets, Technical Report, London South Bank University.

Linear Inequality Control Rules in State-Space Planning: Beyond the first order predicate calculus

Kevin Cleereman and Michael T. Cox

{kcleerem,mcox}@cs.wright.edu

Department of Computer Science & Engineering

Wright State University

Dayton, OH 45435-0001

Abstract

The general planning problem is known to be PSPACE-complete, and thus a planner must employ control rules to efficiently arrive at a solution to any non-trivial problem. These control rules are generally written in variants of the STRIPS or ADL languages, which are extensions of the first order predicate calculus. However, certain characteristics of a given domain may prevent efficient search due to cyclic dependencies in the bipartite relationships between candidate variable bindings. It is not possible for first order control rules to produce an efficient solution to many problems in the resulting pathological domain. We will outline our system of linear inequality control rules that produce efficient solutions in pathological domains by deviating from FOPC. We will further demonstrate that the use of linear inequality control rules will allow for greater flexibility in efficiently splitting a domain for a multi-agent system.

Keywords: Planning, search control, multi-agent systems, resource constraints, pathological dependency cycles.

1 Introduction

Effective planning comes about when a domain is engineered to provide an abstract model that retains the salient characteristics of the domain while removing unnecessary details, resulting in a compact search space. However, the computational complexity of domain-independent planning is known to be at least PSPACE-complete (Bylander, 1991; Chapman, 1987; Selman, 1994), so for any non-trivial problem search control is needed to further guide the planner through the search space. These control rules are generally written in variants of the STRIPS or ADL languages, which are extensions of the first order predicate calculus (FOPC). However, certain characteristics of a given domain may prevent efficient search due to cyclic dependencies in the bipartite relationships between candidate bindings and required variables in operator representations. This paper presents a new system of linear inequality (LI) control rules that eliminates the need to employ costly backtracking in binding assignment problems containing a pathological dependency cycle.

An impediment to efficient search control arises when a planning domain possesses several possible binding compositions for its operators. A "*binding composition*" refers to a set of domain objects whose candidate variable bindings are mutually constrained. For instance, consider the operator FIT that associates pegs and holes. Because one peg can fit into at most one hole, and because one hole can contain at most one peg, the process of fitting pegs to holes is simply a bipartite matching problem (Asratian, Denley, & Häggkvist, 1998).

State-space planners employing FOPC control rules are able to match pegs to holes with control rules of the form "if you have a square hole, then fill it with a square peg" or "if you have fit a peg into a hole, then do not try to fit the same peg into another hole." The task of fitting pegs into holes is much more complicated when, for example, a square peg is allowed to fit into either a square hole *or* a hex hole, while a round peg is allowed to fit into either a round hole *or* a square hole. It now becomes possible for the planner to make inappropriate binding choices by, e.g., fitting all of the square pegs into square holes and making it impossible to fill the hex-shaped holes. However, it is still possible to determine an absolute binding hierarchy in this domain, because all hex holes *must* be filled with square pegs and all round holes *must* be filled with round pegs. Therefore, a state-space planner employing FOPC control rules can avoid making mistakes by filling all hex and round holes before it fills square holes.

Unfortunately, it is no longer possible for FOPC control rules to impose an absolute binding hierarchy in a domain containing cycles in its bipartite graph representation due to particular binding compositions. We call these bipartite graph cycles *pathological dependency cycles*. A domain containing a pathological dependency cycle is called a *pathological domain*. Our non-pathological domain given above is transformed into a pathological domain if we also state that hex pegs may fit into hex holes *or* round holes, because then we can no

longer commit to any bindings without running the risk of having to backtrack over our binding decisions. State-space planners solve this relatively simple task of fitting pegs into holes in exponential time if the planning domain is pathological (Cleereman and Cox, 2004).

In the next section we will introduce our system of LI control rules in a non-pathological domain, and will compare them to FOPC control rules. In section 3 we will demonstrate that FOPC control rules are inadequate in a pathological domain, but that our LI control rules are sufficient for insuring that no backtracking will be required. In section 4 we will demonstrate the use of LI control rules in splitting a domain into three sub-domains for an asynchronous multi-agent system. In the fifth and final section we will offer our conclusions.

2 LI and FOPC Control Rules

The standard logistics (i.e., package delivery) domain (Veloso, 1994) is used by many researchers who study planning. This domain models the transfer of objects and vehicles between various locations. Standard operators include the following.

DRIVE (loc1 isa(LOCATION), loc2 isa(LOCATION)) – transfers a vehicle and its contexts from LOCATION loc1 to LOCATION loc2

LOAD (obj1 isa(OBJECT), carrier1 isa(CARRIER)) – loads an OBJECT obj1 onto a CARRIER carrier1

UNLOAD (obj1 isa(OBJECT), carrier1 isa(CARRIER)) – unloads an OBJECT obj1 from a CARRIER carrier1

The preconditions for these operators are fairly intuitive. For example, to UNLOAD an OBJECT from a CARRIER, the OBJECT must first be inside the CARRIER, and to LOAD an object into a CARRIER both the OBJECT and CARRIER must be at the same LOCATION.

To examine unusually hard problems, we modified the domain such that it would perform the basic operations as follows. We specify that once a CARRIER LOADs an OBJECT in a given problem, it may not LOAD another OBJECT for the remainder of that problem[1].

Control rules come into use when avoiding the various idiosyncrasies of planning algorithms. For example, in our domain outlined above the effects list of the LOAD operator would add the predicate `IS-USED(Carrier1)` when the planner binds an OBJECT to the CARRIER Carrier1, and we would include as a precondition to the LOAD operator `NOT(IS-USED(CARRIER))` to insure that a previously used CARRIER could not be loaded again. However, a non-linear planner using back-chaining may inadvertently bind the same CARRIER Carrier1 to multiple OBJECT objects with the UNLOAD operator, forcing it to backtrack when it finds that it can LOAD only a single OBJECT (and thus can UNLOAD only a single OBJECT). Rather than re-engineer the entire domain, it is often far simpler to write a control rule that will, in this case, reject the CARRIER objects that have previously been employed in an UNLOAD operator when attempting to bind a CARRIER to the UNLOAD operator.

Control rules are also needed to direct the flow of the planning cycle. For example, in a Blocksworld problem with initial state {`ON-TABLE(Block-A), ON-BLOCK(Block-B, Block-A), ON-TABLE(Block-C)`} and goal state {`ON-BLOCK(Block-A, Block-B), ON-BLOCK(Block-B, Block-C), ON-TABLE(Block-C)`}, then the problem can be decomposed into the goals `ON-BLOCK(Block-A, Block-B)` and `ON-BLOCK(Block-B, Block-C)`. However, uninformed search may first select the goal `ON-BLOCK(Block-A, Block-B)`, resulting in the state {`ON-TABLE(Block-C), ON-TABLE(Block-B), ON-BLOCK(Block-A, Block-B)`}. This goal will then have to be undone when the goal `ON-BLOCK(Block-B, Block-C)` is next selected, because Block-B cannot be moved onto Block-C until it is cleared. A control rule can insure that the goal `ON-BLOCK(Block-B, Block-C)` is selected first, thus improving the planner's efficiency.

The distinction between these two types of rules can be expressed as follows. If a particular problem can be expressed as a maximum matching problem on a bipartite graph, then the first type of control rule is used to insure that the rules of the graph matching problem are not violated. The second type of control rule is used to increase the efficiency of the search to locate a maximum matching. While the first type of control rule is still important to the planning process, we will focus our attention on the second type.

We further modified the logistics domain to increase the difficulty of the matching problem. The LOAD operators were altered so that certain CARRIER objects could only load certain types of objects.

1. LOAD-STD_TRUCK (obj1 isa(PRODUCE), carrier1 isa(STD_TRUCK))
2. LOAD-REFRIG_TRUCK (obj1 isa(OR(PRODUCE, DAIRY)), carrier1 isa(REFRIG_TRUCK))
3. LOAD-FREEZER_TRUCK (obj1 isa(OR(DAIRY, MEAT)), carrier1 isa(FREEZER_TRUCK))

[1] The only exception to this rule is if the planner backtracks over a previous binding commitment. In either case, a CARRIER will LOAD at most one OBJECT in the final plan.

The relations between these LOAD operators can be expressed as a bipartite graph. Arrows represent the valid binding pairs.

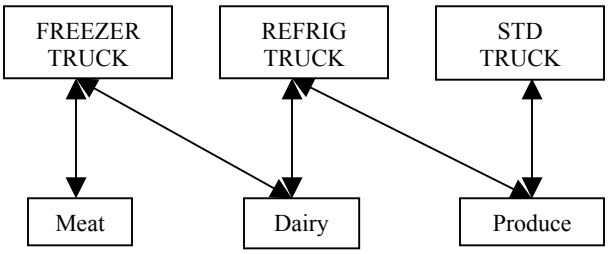


Figure 1. Bipartite Graph Representation.

Note that this graph is a simplified domain representation. Any particular problem has a given number of FREEZER_TRUCK, REFRIG_TRUCK, STD_TRUCK, MEAT, DAIRY, and PRODUCE nodes, but these nodes have edges as given by the domain graph (e.g., all MEAT nodes connect to all FREEZER_TRUCK nodes).

For any given problem, the number of FREEZER_TRUCK, REFRIG_TRUCK, STD_TRUCK, MEAT, DAIRY, and PRODUCE nodes is given by the variables *freezer*, *refrig*, *std*, *meat*, *dairy*, and *prod*, respectively[2]. Thus, a problem possesses a solution iff

1. *std* + *refrig* >= *produce*
2. *refrig* + *freezer* >= *dairy* + *meat*
3. *freezer* >= *meat*
4. *std* + *refrig* + *freezer* >= *produce* + *dairy* + *meat*

This is supported by Hall's Theorem (Nering, 1993). If we operate on the assumption that all of these inequalities are satisfied at the start of a problem, then we can write control rules that will insure that these inequalities are not violated during the course of planning.

Linear inequality control rules as shown in Table 1 consist simply of determining if one of these inequalities will be violated by a binding decision before committing to that binding decision.

The meta-predicate "cand-match" (short for candidate-match) returns true if there is a CARRIER object of the given type that is available to LOAD a PACKAGE of the given type, else it returns false. If the meta-predicate and all inequalities return true, then the CARRIER and PACKAGE are bound to the LOAD operator, else the binding pair is rejected.

[2] A CARRIER or OBJECT will only have a node if it is available for binding in a LOAD operator. For example, a CARRIER Carrier1 that is `IS-USED(Carrier1)` in the initial state will not have a representative node, and an OBJECT that is already at its goal location will not have a representative node.

When using LI control rules for this domain, the planner need not select its goals or operators in any particular order. The matter changes when using FOPC control rules, because FOPC cannot explicitly confirm that the consistency of the inequalities is being maintained. FOPC is still able to insure that no backtracking occurs in a non-pathological domain, but it must explicitly order its goal and operator selection to do so.

Table 1. LI Control Rule Set

1	**if** cand-match(STD_TRUCK – PRODUCE) & (*std* – 1 + *refrig* >= *produce* – 1) **then** select binding; *std*--; *produce*--; **else** reject binding
2	**if** cand-match(REFRIG_TRUCK – PRODUCE) & (*refrig* – 1 + *freezer* >= *dairy* + *meat*) **then** select binding; *refrig*--; *produce*--; **else** reject binding
3	**if** cand-match(REFRIG_TRUCK – DAIRY) & (*std* + *refrig* – 1 >= *produce*) **then** select binding; *refrig*--; *dairy*--; **else** reject binding
4	**if** cand-match(FREEZER_TRUCK – DAIRY) & (*freezer* – 1 >= *meat*) **then** select binding; *freezer*--; *dairy*--; **else** reject binding
5	**if** cand-match(FREEZER_TRUCK – MEAT) & (*refrig* + *freezer* – 1 >= *dairy* + *meat* – 1) **then** select binding; *freezer*--; *meat*--; **else** reject binding

In contrast, FOPC control rules as shown in Table 2 guide search by taking the given action(s) if a particular sentence in FOPC is true. FOPC cannot efficiently represent a linear inequality, and thus LI control rules have greater representational power.

The meta-predicate "candidate-goal" returns true if the given goal is queued, the meta-predicate "current-goal" returns true if the given goal has been selected, and the meta-predicate "candidate-op" returns true if the given operator is a candidate for achieving the current goal state.

By insuring that all `Is-Loaded(Meat)` goals are solved before all `Is-Loaded(Dairy)` and `Is-Loaded(Produce)` goals, FOPC is able to insure that the inequality *freezer* >= *meat* is not violated by an inauspicious FREEZER_TRUCK – DAIRY match. Similarly, by insuring that all `Is-Loaded(Dairy)` goals are solved before all `Is-Loaded(Produce)` goals, FOPC is able to insure that the inequality *refrig* + *freezer* >= *dairy* is not violated by an inauspicious REFRIG_TRUCK – PRODUCE match[3].

[3] Note that an alternative set of FOPC control rules could have been used instead. For example, we might have decided to solve all `Is-Loaded(PRODUCE)` goals first, followed by all `Is-`

Table 2. FOPC Control Rule Set

1	**if** candidate-goal(Is-Loaded(Meat)) & candidate-goal(Is-Loaded(Dairy)) **then** prefer goal Is-Loaded(Meat)
2	**if** candidate-goal(Is-Loaded(Meat)) & candidate-goal(Is-Loaded(Produce)) **then** prefer goal Is-Loaded(Meat))
3	**if** candidate-goal(Is-Loaded(Dairy)) & candidate-goal(Is-Loaded(Produce)) **then** prefer goal Is-Loaded(Dairy)
4	**if** current-goal(Is-Loaded(Meat)) & candidate-op(LOAD-FREEZER_TRUCK) **then** prefer op LOAD-FREEZER_TRUCK
5	**if** current-goal(Is-Loaded(Dairy)) & candidate-op(LOAD-FREEZER_TRUCK) **then** prefer op LOAD-FREEZER_TRUCK
6	**if** current-goal(Is-Loaded(Produce)) & candidate-op(LOAD-REFRIG_TRUCK) **then** prefer op LOAD-REFRIG_TRUCK

3 Pathological Domains

While LI and FOPC control rules have similar expressive powers on some domains, there are domains containing pathological dependency cycles for which no set of FOPC control rules can insure that backtracking will not occur. If one or more pathological dependency cycles occur within a domain, then that is a pathological domain. For example, if in our previously modified logistics domain we had specified that STD_TRUCK objects could move either PRODUCE or MEAT:

LOAD-STD_TRUCK (obj1 isa(OR(PRODUCE, MEAT)), carrier1 isa(STD_TRUCK))

Then the domain's graphical representation will change as follows. Note the new STD_TRUCK – MEAT edge.

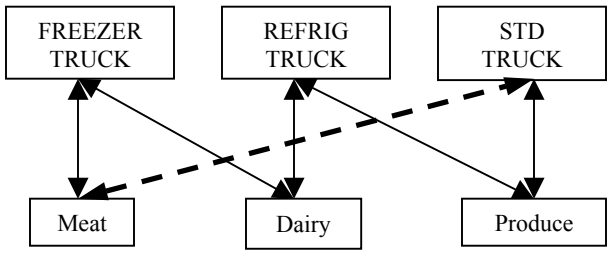


Figure 2. Pathological Domain.

`Loaded(Dairy)` goals. We could not, however, have solved all `Is-Loaded(Dairy)` goals first, for as we will explain in Section 4 the *refrig* + *freezer* >= *dairy* equation creates a pathological sub-domain.

We call this relationship a pathological dependency cycle, because the addition of a cycle to the graph structure effectively disables FOPC control rules. A problem in the pathological domain will possess a solution iff

1. *std* + *refrig* >= *produce*
2. *refrig* + *freezer* >= *dairy*
3. *freezer* + *std* >= *meat*
4. *std* + *refrig* + *freezer* >= *produce* + *dairy* + *meat*

Note the change in inequality 3 from the non-pathological domain. Our previous set of FOPC control rules will no longer work in maintaining the consistency of these inequalities, for *any* possible match has the chance of causing the left-hand side of an inequality to possess a lesser value than the right-hand side. For example, given a simple problem with *freezer* = 1, *std* = 1, *meat* = 1, and *dairy* = 1, it is clear that a FREEZER_TRUCK – MEAT binding will violate equation 2, yet our FOPC control rules have no means of determining that this is the case and will thus have to backtrack to reverse this binding decision. If we alter our control rules so that they favor STD_TRUCK – MEAT bindings over FREEZER_TRUCK – MEAT bindings, then the simple problem with *freezer* = 1, *std* = 1, *meat* = 1, and *produce* = 1 will require backtracking to free up the STD_TRUCK object to load the PRODUCE object. In contrast, our LI control rules require little modification, and are still capable of insuring that no backtracking need occur.

Table 3. Pathological LI Control Rule Set

1	**if** cand-match(STD_TRUCK – PRODUCE) & (*freezer* + *std* – 1 >= *meat*) **then** select binding; *std*--; *produce*--; **else** reject binding
2	**if** cand-match(STD_TRUCK – MEAT) & (*std* – 1 + *refrig* >= *produce*) **then** select binding; *std*--; *meat*--; **else** reject binding
3	**if** cand-match(REFRIG_TRUCK – PRODUCE) & (*refrig* – 1 + *freezer* >= *dairy*) **then** select binding; *refrig*--; *produce*--; **else** reject binding
4	**if** cand-match(REFRIG_TRUCK – DAIRY) & (*std* + *refrig* – 1 >= *produce*) **then** select binding; *refrig*--; *dairy*--; **else** reject binding
5	**if** cand-match (FREEZER_TRUCK – DAIRY) & (*std* + *freezer* – 1 >= *meat*) **then** select binding; *freezer*--; *dairy*--; **else** reject binding
6	**if** cand-match(FREEZER_TRUCK – MEAT) & (*refrig* + *freezer* – 1 >= *dairy*) **then** select binding; *freezer*--; *meat*--; **else** reject binding

Note that when insufficient CARRIER resources exist, it will be necessary for goal transformations (Cox, 2003; Cox and Veloso, 1998) to eliminate nodes from the right-hand (OBJECT) side of one or more inequalities, else the LI control rules will prevent a partial solution from being achieved. For example, given a simple problem with *freezer* = 1, *meat* = 1, and *dairy* = 1, the single FREEZER_TRUCK object will be prevented from being bound to the MEAT object because this would make the goal of loading the DAIRY object unattainable, but the FREEZER_TRUCK object will also be prevented from being bound to the DAIRY object because this would make the goal of loading the MEAT object unattainable. When a goal transformation erodes or eliminates one of the `Is-Loaded(Object)` goals then the LI control rules will achieve their desired effect.

4 Multi-Agent Systems

The advantages of using LI control rules in lieu of or in addition to FOPC control rules are most apparent when a multi-agent system splits a domain into sub-domains to delineate agent responsibility for planning problems (Cox, Elahi, & Cleereman, 2003; Elahi, 2003). For example, we may wish to split the non-pathological logistics domain containing 3 Load operators (given in section 2) into a 3-agent system so that each agent will possess a single Load operator. Specifically, Agent-1 possesses the Load-Std_Truck operator, Agent-2 possesses the Load-Refrig_Truck operator, and Agent-3 possesses the Load-Freezer_Truck operator. We will assume that all three agents have access to all OBJECT objects in a given problem, meaning that they can bind their CARRIER objects to any OBJECT they wish.

We can treat this domain as we did the single-agent domain by stipulating that only one agent may act on the problem space during any given clock cycle, effectively solving problems sequentially. For example, if we stipulate that Agent-2 cannot act until Agent-3 has either bound all of its FREEZER_TRUCK objects or has bound all of the MEAT and DAIRY objects then we are in effect applying control rules {1, 2, 4, 5} given in Table 2. If we further stipulate that Agent-1 cannot act until Agent-2 has bound all of its REFRIG_TRUCK objects then our multi-agent system will be essentially equivalent to the single-agent system in Section 2. However, it is often desirable for agents in a multi-agent system to act in parallel and/or asynchronously to maximize processor throughput and minimize the time needed to generate a plan. Because one or more sub-domains of a non-pathological domain may be pathological (as it is in our example), LI control rules will often be needed to maximize planning efficiency.

Agent-1:
1. *std* + *refrig* >= *produce*

Agent-2:
1. *std* + *refrig* >= *produce*
2. *refrig* + *freezer* >= *dairy*

Agent-3:
3. *refrig* + *freezer* >= *dairy*
4. *freezer* >= *meat*

Agent-1 and Agent-3 both possess non-pathological sub-domains, because any reduction in *std* will by necessity produce a reduction in *produce*, and likewise for Agent-3 any reduction in *freezer* will produce a reduction in *meat* when the Table 2 control rules are being applied. However, Agent-2 possesses a pathological sub-domain: any reduction in *refrig* that is not accompanied by a reduction in *produce* (i.e., any REFRIG_TRUCK – DAIRY match) may violate equation 1, and any reduction in *refrig* that is not accompanied by a reduction in *dairy* (i.e., any REFRIG_TRUCK – PRODUCE match) may violate equation 2. It is thus necessary for Agent-2 to explicitly confirm one of these equations via LI control rules before it commits to a match.

The problem with a multi-agent pathological domain is essentially equivalent to the problem with a single-agent pathological domain. If we divide up the Load operators of the pathological logistics domain given in section 3 as we divided up the Load operators of the non-pathological logistics domain earlier in this section, then we are left with the following system of inequalities to form the basis of multi-agent negotiation and coordination.

Agent-1:
1. *std* + *refrig* >= *produce*
2. *freezer* + *std* >= *meat*

Agent-2:
1. *std* + *refrig* >= *produce*
2. *refrig* + *freezer* >= *dairy*

Agent-3:
1. *refrig* + *freezer* >= *dairy*
2. *freezer* + *std* >= *meat*

This pathological domain has split into three pathological sub-domains, leaving all agents unable to commit to any bindings when using FOPC control rules. However, LI control rules behave no differently for the multi-agent domain than for the single-agent domain, allowing even pathological domains to benefit from the advantages of being represented as a multi-agent system.

5 Conclusions

The problem of efficiently allocating resources is ubiquitous in planning, but is often inefficiently

represented by STRIPS and ADL like languages. Linear Inequality control rules are capable of producing vast increases in planning efficiency, yet they require that only a domain's control rule representation be modified, *not* the domain itself.

More importantly, LI control rules define a means by which agents in a multi-agent system can communicate with each other and efficiently allocate planning resources between one another. Agents are not bound to a sequential order of operations as they may be under the direction of FOPC control rules, but are instead given the freedom to act in parallel and/or asynchronously with other agents.

We are currently in the process of investigating the use of LI control rules in automatic goal transformations, as well as the use of LI control rules in domains represented as a graph of 3 or more colors. We have already determined that LI control rules allow some 3-colored domains to produce efficient solutions, but we are still unsure of whether this is the exception or the rule. The general 3-colored matching problem is NP-Complete, so we would expect that either some domains cannot be solved with LI control rules, or that the task of formally generating LI control rules for a general 3-colored domain is NP-Complete, or both.

Acknowledgements

This research is funded by a contract from the Air Force Research Laboratory and by the Ohio Board of Regents. We especially thank Dr. Mateen Rizki of the Wright State University CECS Department for recognizing the parallel between our binding selection problem and the bipartite matching problem. We also thank Mr. Langdon and Mr. Somanchi of the Wright State University Collaboration and Cognition laboratory for their valuable input.

References

Asratian, A. S., Denley, T., and Häggkvist, R. (1998). *Bipartite Graphs and Their Applications.* Cambridge University Press: Cambridge.

Bylander, T. (1991). Complexity results for planning. In *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 274-279)**.**

Chapman, D. (1987). *Planning for conjunctive goals*. Artificial Intelligence 32:333-377.

Cleereman, K., and Cox, M. T. (2004). Pathological Dependency Cycles in State-Space Planning: When Control Rules Fail. In *Proceedings of the Florida Artificial Intelligence Research Society (FLAIRS-04) Conference.* Menlo Park, CA: AAAI Press.

Cox, M. T. (2003). Planning as mixed-initiative goal manipulation . In *Proceedings of the Workshop on Mixed-Initiative Intelligent Systems at the 18th International Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press.

Cox, M. T., Elahi, M., and Cleereman, K. (2003). A distributed planning approach using multiagent goal transformations. In Ralescu (Ed.), *Proceedings of the 14th Midwest Artificial Intelligence and Cognitive Science Conference* (pp 18-23). Cincinnati: Omnipress.

Cox, M. T., and Veloso, M. M. (1998). Goal transformations in continuous planning. In M. desJardins (Ed.), *Proceedings of the 1998 AAAI Fall Symposium on Distributed Continual Planning* (pp. 23-30). Menlo Park, CA: AAAI Press / The MIT Press.

Elahi, M. M. (2003). *A distributed planning approach using multiagent goal transformations.* Masters Thesis, Department of Computer Science and Engineering, Wright State University.

Nering, E. D. (1993). *Linear Programs and Related Problems* (pp 293-298). Boston: Academic Press.

Selman, B. (1994). Near-Optimal Plans, Tractability, and Reactivity. In J. Doyle, E. Sandewall, & P. Torasso, (Eds.), *Proc. 4th Conference in Knowledge Representation* (pp. 521—529), San Francisco: Morgan Kaufmann.

Veloso, M. M. (1994). *Planning and Learning by Analogical Reasoning*. Berlin: Springer.

Index of Authors