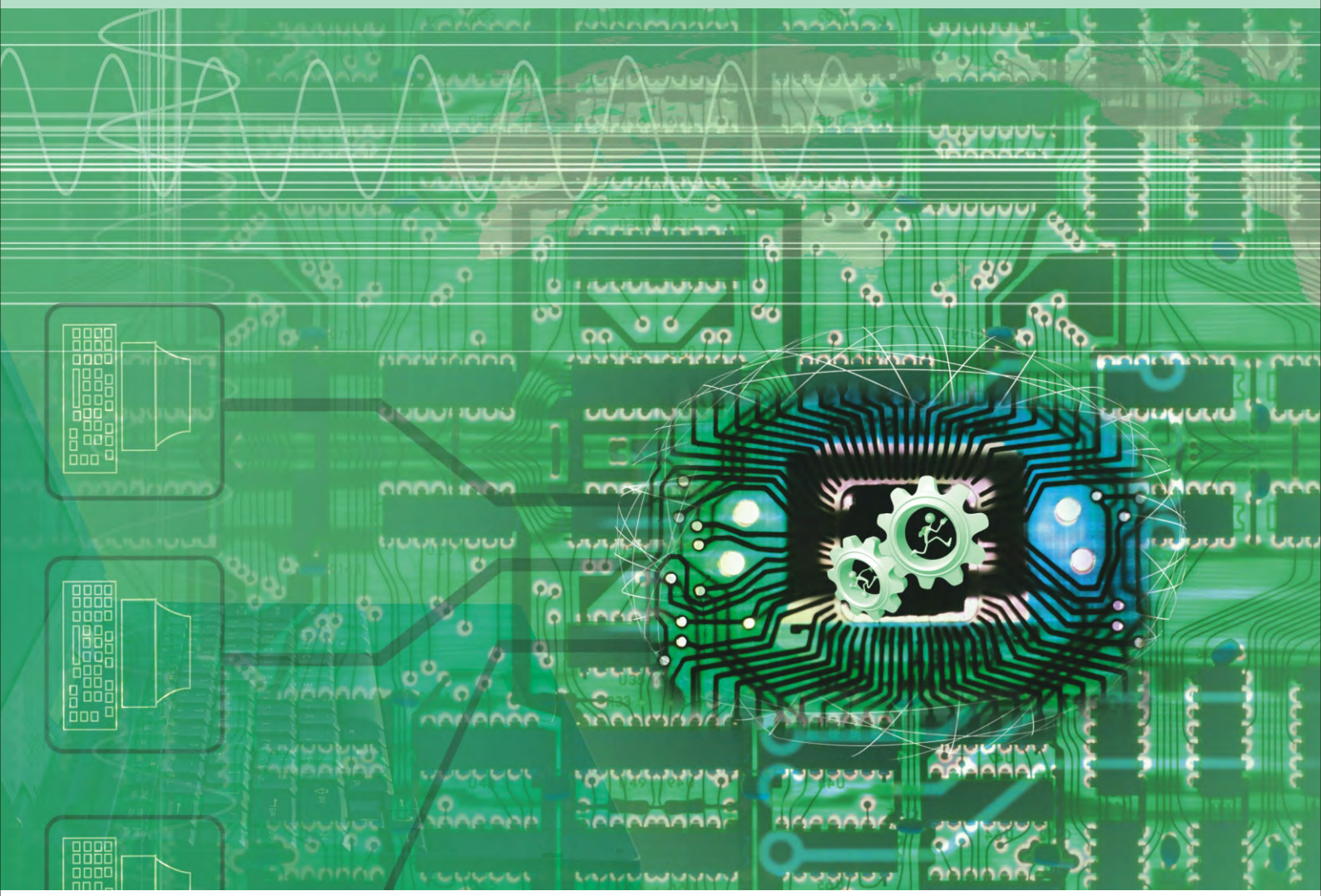
ISSN: 2150-8194 (Print), 2150-8208 (Online) Vol. 1, No. 2, November 2009

Intelligent Information Management

智能信息管理 國際期刊

Editor-in-Chief: Dr.Bin Wu



www.scirp.org/journal/iim

Scientific Research

Journal Editorial Board

ISSN: 2150-8194 (Print), 2150-8208 (Online)

http://www.scirp.org/journal/iim

Intelligent Information Management, **2009, 1, 65-138**
http://www.scirp.org/journal/iim

Scientific Research

Table of Contents

Intelligent Information Management (IIM)

Journal Information

SUBSCRIPTIONS

The *Intelligent Information Management* (Online at Scientific Research Publishing, www.SciRP.org) is published quarterly by Scientific Research Publishing, Inc., USA.

E-mail: service@scirp.org

Subscription rates: Volume 1 2009
Print: $50 per copy.
Electronic: free, available on www.SciRP.org.
To subscribe, please contact Journals Subscriptions Department, E-mail: service@scirp.org

Sample copies: If you are interested in subscribing, you may obtain a free sample copy by contacting Scientific Research Publishing, Inc. at the above address.

SERVICES

Advertisements
Advertisement Sales Department, E-mail: service@scirp.org

Reprints (minimum quantity 100 copies)
Reprints Co-ordinator, Scientific Research Publishing, Inc., USA.
E-mail: service@scirp.org

COPYRIGHT

PRODUCTION INFORMATION

For manuscripts that have been accepted for publication, please contact:
E-mail: iim@scirp.org

Intelligent Information Management, **2009, 1, 65-72**
doi:10.4236/iim.2009.12011 Published Online November 2009 (http://www.scirp.org/journal/iim)

Scientific Research

The Specification of Agent Interaction in Multi-Agent Systems

Dmitri CHEREMISINOV
United Institute of Informatics Problems, National Academy of Sciences of Belarus, Minsk, Belarus
Email: *cher@newman.bas-net.by*

Abstract: The problem of the description of interaction between agents in a multi-agent system (MAS) in the form of dialogues of negotiations is considered. For formalization of the description of interaction at a level of steps of the dialogue which is carried out in common by two spatially divided agents, the concept of synchronization of processes is analyzed. The approach to formalization of the description of conditions of synchronization when both the independent behaviour, and the communications of agents can be presented at a level of logic is offered. It is shown, that the collective behavior of agents can be described by the synthetic temporal logic that unite linear and branching time temporal logics.

Keywords: interaction protocol, temporal logic, parallel algorithm

1. Introduction

A multi-agent system [1] consists of set of agents that operate together in order to achieve some goals. Such system can be considered as the organization of agents (by analogy to the human organization) or, in other words, as some artificial society. Protocols play the central role in the organization of this society. The protocol of interaction can be stipulated by the designer of investigated system, or the agents who are taking part in information interchange agree about the application of a protocol before interaction took place.

The interaction protocol of agents defines the rules that govern the dialogue between agents in multi-agent system. The central problem of the interactions that took place in open systems and not being cooperative (dialogues of negotiations) is the problem of conformity check between agent behavior and interaction protocol. The implementation of conformity check encounters the trouble what is the identification of steps of dialogue. Recognition of the step which is carried out by two spatially divided agents jointly, require analyzing the concept of interaction of processes.

The basis of formal models of protocols is cooperating sequential processes. Fundamental features these models differ are the degree of synchronization of behaviour of participants of interaction. This paper is the attempt to analyze by logic the concept of synchronization in these frameworks.

2. Formalization of Concept of Interaction Event

Usually collective behaviour of system of agents is described as dialogue of agents which communicate by means of sending and receiving messages. On each step of activity the agent carries out some action depending on the internal state and the received message. As a result the action changes the internal state and sends messages to other agents. The collaboration of agents emphasise autonomy and cooperation (with other agents) in order to perform tasks for their owners. Collaboration behaviour of system of agents is unreachable when the agent doesn't know anything about individual behaviour of other agents. Particularly because the agents are autonomous and cannot be assumed to be benevolent, agents must know that others must do to act in certain ways without requiring that one examine the internal reasoning (or the source code) of the agents. Speaking informally, the protocol is formal representation of knowledge of the agent about individual behaviour of other agents into the scope of joint task. There is still a need for a proper formalism for protocols that is suitable for automated implementation. Suppose we are given a protocol specification, *PS*. One way of obtaining a concrete agent program from *PS* is to treat it as an *executable specification*, and *interpret* the specification directly in order to generate the agent's behavior. If models for the specification language can be given a computational interpretation, then model building can be viewed as executing the specification (Figure 1).

This knowledge is inexpedient to treat as parameter of an agent program from speed consideration. This knowledge must be realized by programmer as the agent program. In this scheme, we take our abstract representation of the knowledge, and transform it into a concrete program via some synthesis process.

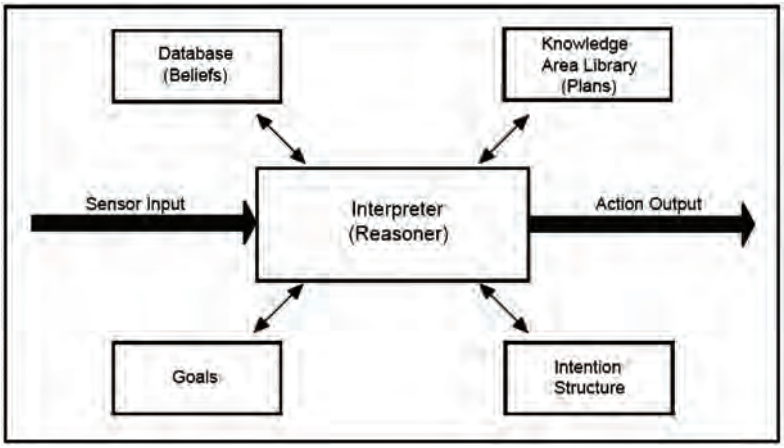


Figure 1. Parameters of universal agent

For the specification of agent independent behaviour are widely used formalisms of high level abstractness, for example, such as temporal logic. At the same time the communications between agents is specified by means of concepts of a realization level, such as mail boxes and messages. One of problems of such segregated approach to interactions consists that it is extremely difficult to model the interactions between agents though the independent behaviour of the agent is described completely. This problem arises due to absence of agent model unifying aspects of independent behaviour and the communications. The main reason of absence of the general model is the lack of conceptual basis unifying all abstraction, connected with collective behaviour of agents.

The independent behaviour of multi-agent system agents are characterized by processes. Process is defined by the full description of potential behaviour of the agent. The behaviour of process consists of events. Thus, suitable axiomatization of concept of event is required for the analysis of concept of interaction of processes.

The event serves as the concept to abstract from physical time at the description of behaviour of system. Widely widespread axiomatization of event is connected with the assumption, that events have no duration [2]. The behaviour multi-agent system consists of events – steps of dialogue between agents – and is consecutive in this sense. For recognition of the step which is carried out by two spatially divided agents jointly, it is impossible to bypass the concept of parallelism.

The models of parallelism known in the literature is possible roughly divided into two classes: 1) models, in which concurrent execution of two processes described by interleaving of (atomic) events of those processes; 2) models in which causal dependencies between events are set explicitly. Interleaving models are focused on systems with events are considered instant and indivisible. In this case the act of interaction is complete event which describe participation of all processes cooperating in this act [2]. This act as the step of dialogue are carried by two spatially divided agents represent event which should have duration and structure.

There is popular opinion the concept of the event having duration is reduced to concept of instant event. The following formulation of these assumptions is taken from Hoare [3] "The actual occurrence of each event in the life of an object should be regarded as an instantaneous or an atomic action without duration. Extended or time-consuming actions should be represented by a pair of events the first denoting its start and the second denoting its finish."

Now it is known that this opinion is erroneous, and behaviour in the events having duration is not reduced to the behaviour expressed through instant events. Mutual irreducibility the concept of the event having duration to concept of instant event is proved formally and constructively [4]. The formal proof is based on incomparability of formalism describing event systems [5]. Systems of the events having duration are described by the causal relation (branching-time temporal logic), systems of instant events – relation of consequence and parallelism (linear-time temporal logic).

3. Structure of Interaction Event

The elementary structure of durational event that is a step of dialogue is pair of durational events which constitute a step densely without a time interval between. First event can be interpreted as "pronouncing" of the message by one agent; the second event can be interpreted as "perception" of this message by other participant of dialogue. The basic feature of this structure is the assumption of density of an event composition and that constituting

events belong to behaviour of different agents (Figure 2). Absence of a time interval between members of pair durational events designates instant event *of synchronization* of processes.

3.1. The Event Model of Synchronization

Abstracting from function of agents, it is possible to consider synchronization of their behaviour as the goal of interaction. Thus, the step of dialogue is a composition of *three* events, *two* events are durational events, and *one* is instant event. However constituting event of interaction still remains event which occur simultaneously in *different* processes.

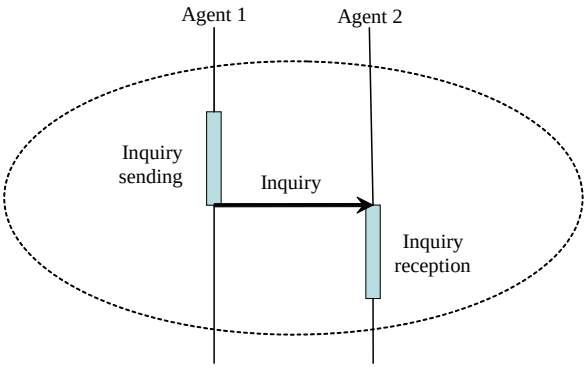


Figure 2. Structure of the communication event

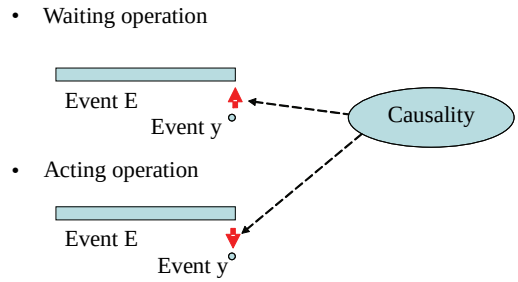


Figure 3. Structure of operations

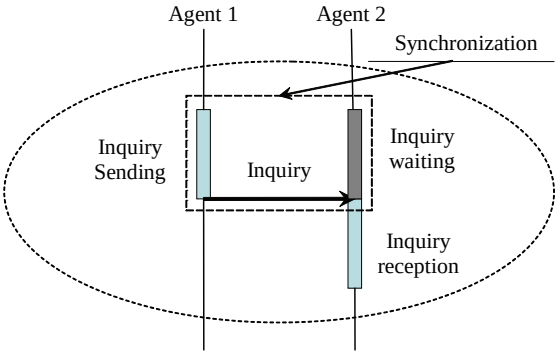


Figure 4. Synchronization of behaviour of agents

On the one hand synchronization of agent behaviour occurs during the rare moments. In the rest of the time communicating agents behave independently from each other. On the other hand, processes should interchange information about current states to ensure synchronization. Formally it can be reached by splitting of all events constituting agent behaviour on internal and external events. Only external events of the agent can be "visible" to other agents. In this case the specification of the agent behaviour is the causal relation on set of possible events, in particular, this relation describe the reasons of occurrence of external and internal events.

Let composition durational internal event E and instant external event y is *operation*. A composition $y \to E$ durational and instant events is waiting operation that wait external event y, and a composition $E \to y$ is acting operation which effect is realization of external event y (Figure 3). It is necessary to note, that order of events in both operations is the same: the first goes durational event, then goes instant event. These compositions allow to consider dependences between events of a composition as cause and effect because from physical reasons event-consequence occurs behind event-reason without overlapping on time. Waiting operation is durational event end reason of its termination is y. Acting operation is durational the event and the reason of y is its termination. The symbol "$\to$" we interpret as cause and effect dependence between events. Such treatment of waiting and acting operations is a basis of formal semantics PRALU language [6] in which conjunction of Boolean variables describe external event of operations.

PRALU language in this interpretation represents the synthetic temporal logic uniting linear and branching time temporal logics supplied by the assumption of density of time [7]. Temporal formulas of this logic are interpreted as the statement concerning order of events of two sorts: instant and durational.

The composition considered above allows to describe independent behaviour of agents. Parallel execution of waiting operation by one agent and acting operation by other agent leads to synchronization of behaviour of agents during the moment of instant event y (Figure 4). By definition the effect of waiting operation is its termination at moment of instant external event.

The line of "life" of the agent consists of pairs waiting and acting operations. The border between them serves as the synchronization event. Here the action consists from "perception" of the accepted message and "pronouncing" new message. Obviously, the occurrence of synchronization depends on duration of acting operations.

3.2. The Interaction as an Event of an Environment

The step of dialogue can be considered as the *other* composition of three events; *one of which* is durational,

and others *two* are instant (Figure 5). In this model of a dialogue step the durational event is directly interaction. This event, having duration, should have a physical basis. Without loss of a generality it is possible to consider that event of interaction occurs in *an environment* of agents. In this model event of interaction becomes not distributed, but this event is *local* in an environment.

4. Concept of an Environment

From consideration of physical realizations of the distributed systems follows that the synchronization achieve by the special organization of system of cooperating agents. Two general types of the system organization for achievement of synchronization are known: *synchronous* and *asynchronous* systems. The standard definition of distinction of these types of systems consists that synchronous systems have shared "clock", and it is said to be *asynchronous* if each agent has its own independent clock. It is obvious, that these shared "clock" are an accessory of an environment of agents.

In traditional interaction theories CCS [3] or CSP [8] concept of an environment is used implicitly, hence it is not formalized. CCS and CSP use on concept of an environment which consists in the following. The environment is considered simply other agent. Other words, the environment for the given agent includes all other agents of the system operating in parallel with this agent. In this case the agent and an environment are objects of identical nature. It is obvious, that this assumption of properties of an environment is not well from the point of view to specify the interaction for achievement of synchronization. Such approach is justified by following. It is considered that the concept of an environment concern with realization of system and is not representable at a level of behaviour.

Our purpose consists in offering formal model of interaction which is not concern with realization of system. We consider an environment essentially distinct from agents. The basis of this approach is the representation about interaction as the communication act consisting from sending and receiving of the messages. This formalization of interaction originates from Shannon's paper about the theory of communication [9] in which interaction is considered as a way of transfer of the message from sender to a receiver via a medium also called transfer environment. Physical realization of an environment can be the computer program, the device or the physical environment.

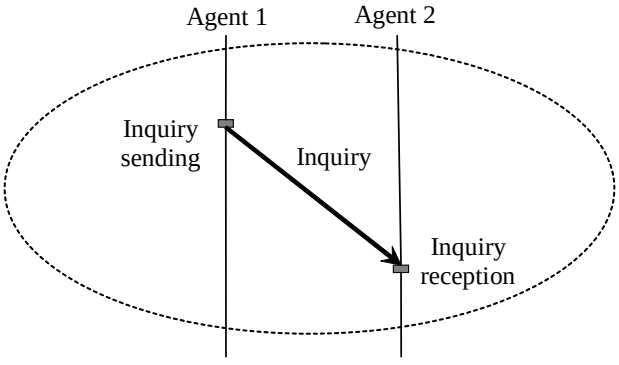


Figure 5. Synchronization event as environment event

Obviously, synchronization of agent behaviour is impossible without fixing data which are transferred during their interaction [10] by an environment. Thus, environment serves as model of transport system for delivery of messages. Other words environment represents the memory that is shared of all agents. This memory is known as a global state of multi-agent system. In its most simple form, the communications can be based on the fixed set of differing signals. In case of binary signals the representation of a global state is the set of the boolean variables which values are possible signals. In case of structural signals the environment of agents usually refers *as message passing system*. The concept of an environment is closely concern with *autonomy* of agents. Autonomy has its focus on freely choosing between actions and on acting independently. Autonomy means that the agents receive the information only through an environment.

System, in which the behaviour of an environment is deterministic, refers to *closed system*. In case of closed system it is supposed, that the reasons of all events are inside of system and its behaviour is self controlled completely. If the behaviour of an environment is nondeterministic, system refers to *opened system*. Unlike the memory considered in the theory of automatic devices, the behaviour of memory of an environment of the open system can depend on uncontrollable conditions.

The specification of interaction in the form of the description of message passing system does the description of autonomous behaviour of the separate agent not closed because this description is not enough for understanding of the complete behaviour of the agent. Obviously, most important property of the message passing system is restriction on length of durational events, imposed by this system.

5. Time as Logic Concept

In the previous part of this paper the time was considered as the logic concept expressed by relations between events through sequence and order of events. Time is discrete, because there is an observable time quantization by the events that fixed in behaviour of an environment. If we do not impose restrictions on duration of events in all components of multi-agent system, the time is not measured. The *measured* time means, that each event in history of system behaviour is accompanied with number that express either duration of event, or specifying the moment of time when it occur. Synchronization of be-

haviour of agents means the measured time.

Measured time can be realized, if we assume, that duration of all simultaneously executed acting operations in a multi-agent system is identical. This duration is natural for accepting for the unit of time. In this case in the closed systems duration of waiting operations are expressed by an integer $i \geq 1$. The assumption that duration of all simultaneously executed acting operations is identical holds in synchronous system. Obviously, this assumption defines the pairs of interacting waiting and acting operations by number of a step of time. Synchronous system keeps the assumption that time is discrete, dense and measured.

Other assumption to realize measured time is that any operation was carried out in parallel to itself is illegal. In this case realization of any operation in history of functioning of the agent is accompanied with counter number of this realization. The formal proof of this statement is in [7]. The function which calculate a counter number of operation realization (from the start of system) when this operation starts can be used for measurement of time. Interaction occurs only in pairs of waiting and acting operations which have the same count number. This is known as a *rendezvous* condition. Asynchronous system keeps the assumption that time is discrete and measured, but removes the assumption that duration of all simultaneously executed acting operations is identical. They measuring time principle is differ from synchronous type system.

6. Programming of Agents in the PRALU Language

The central problem in a multiagent system is that of coordinating the work of the agents, a process that may be understood as that of assuring that the sequences of actions performed by the agents is consistent and coordinated.

The specification of a conversation in a multiagent system is determined by two components, messages and protocols. The former are described in communication languages, such as Knowledge Query and Manipulation Language (KQML) [11], Foundation for Intelligent Physical Agents (FIPA) ACL [12], and others. A dictionary and messages which may be exchanged by agents are described in the communication languages.

A protocol is a set of interaction rules that serve to coordinate the work of several agents. We consider protocol as common knowledge by which agents achieve some goals. Formal models of protocols have been studied within the context of the theory of distributed computations. The fundamental feature that serves to distinguish these models is the degree of synchronization of behavior of the participants in an interaction. If we abstract from the function of the agents, the objective of interaction turns out to be synchronization of the behavior of the interacting agents, since excessive synchronization entirely undermines the feasibility of joint operation of the agents. Achievement of synchronization requires specialized organization of interacting processes.

The sequences consisting of action and waiting operations are linear algorithms in PRALU. For instance, the following expression means: wait for p and execute A, execute B, then wait for q and execute C: "$-p\rightarrow A\rightarrow B -q\rightarrow C$".

In general, a PRALU algorithm can be presented as an unordered set of chains α_j in the form $\mu_j{:}-p_jL_j\rightarrow \nu_j$, where L_j is a linear algorithm, μ_j and ν_j denote the initial and the terminal chain labels represented by some subsets of integers from the set $M= \{1,2,...,m\}$: $\mu_j, \nu_j \subset M$ and the expression "$\rightarrow\nu_i$" presents the transition operation: to the chains with labels from ν_j.

Chains can be fulfilled both serially and in parallel. The order in which they should be fulfilled is determined by the variable starting set $N_t \subseteq M$ (its initial value $N_0 = \{1\}$ as a rule): a chain $\alpha_j= \mu_j{:}-p_jL_j\rightarrow \nu_j$ (that was passive) is activated if $\mu_j \subseteq N_t$ and $p_j= 1$. After executing the operations of the algorithm L_j, N_t gets a new value $N_{t+1}= (N_t \setminus \mu_j) \cup \nu_j$.

6.1. Representing Agent Interaction Protocols in PRALU

For an example of an interaction protocol, consider an English auction [12]. The auctioneer seeks to find the market price of a good by initially proposing a price below that of the supposed market value and then gradually raising the price. Each time the price is announced, the auctioneer waits to see if any buyers will signal their willingness to pay the proposed price. As soon as one buyer indicates that it will accept the price, the auctioneer issues a new call for bids with an incremented price and continues until no buyers are prepared to pay the proposed price. If the last price that was accepted by a buyer exceeds the auctioneer's (privately known) reservation price, the good is sold to that buyer for the agreed price. If the last accepted price is less than the reservation price, the good is not sold.

In the case of the auction there are participants of two types: the Auctioneer and Buyers. So, we have two kinds of interaction protocols – those of Auctioneer and of Buyers. The last participants are peer and should be described with identical interaction protocols.

Interaction protocol as a whole can be represented in PRALU as three complex acting operations – blocks. Each block has some sets of inputs and outputs that are enumerated in brackets following the block name (the other variables of a block are its internal). Initialization of a complex acting operation is depicted by the fragment such as "$\rightarrow$*Buyer". The operation Buyer exists in as many copies as the number of participants of the auc-

tion, so the copies of the operation differ in their indexes only.

The modeling of the process of auction begins with the execution of "Main_process" triggering event that initiates the interaction protocol execution. Here the processes Auctioneer and $Buyer_n$s are executed concurrently. For the sake of simplicity we limit the number of buyers to two. The process Auctioneer starts with sending the first message (start_auction) that is waited by others participants to continue communication. Below PRALU description of the auction interaction protocol is shown.

Main_process ()
1: → 2.3.4
2: →∗Auctioneer →5
3: →∗$Buyer_1$ →6
4: →∗$Buyer_2$ →7
5.6.7: →**.**

$Buyer_n$ (start_auction, price_proposed, end_auction / $accept\_price_n$, not_understand)
1: – start_auction →2
2: – price_proposed →∗Decide(/ decision_accept, decision_reject) →3
– end_auction →**.**
3: –decision_accept → $accept\_price_n$ →4
– decision_reject →′$accept\_price_n$ →4
– not_understand →4
4: – timeout →2

Auctioneer ($accept\_price_1$, $accept\_price_2$, not_understand / start_auction, price_proposed, end_auction)
1: → start_auction →2
2:→′$accept\_price_1$.′$accept\_price_1$.′not_understand →∗ Price_propose(/price_proposed) →3
3:– not_understand →2
– accept_ $price_1$ →4
– accept_ $price_2$ →4
–′not_understand.′$accept\_price_1$.′$accept\_price_2$ → end_auction → ∗Is_reservation_price_exceeded (/ is_exceeded)→6
4: →′price_proposed.′win_1.′win_2 →5
5: – accept_ $price_1$ →win1 →2
– accept_ $price_2$ →win2 →2
6: – is_exceeded →good_sold →7
– ′is_exceeded →′good_sold →7
7: →.

It is assumed that all unformalized operations are referred to as acting operations that set values of logical variables assigned to them. For example, Buyer's operation "Decide" decides for accepting or rejecting the announced price. Depending on adopted decision, it outputs true value of logical variable "decision_accept" or "decision_reject". In a similar, Auctioneers operation "Price_propose" proposes an initial price or increments the charged price outputting true value of logical variable "price_proposed"; the operation "Is_reservation_price_exceeded" verifies if the price accepted by a buyer exceeds the auctioneer's reservation price outputting true or false value of logical variable "is_exceeded".

The operation "–timeout" (where timeout is integer number) means waiting for timeout unit times before doing something followed it. The operation "→**.**" is interpreted as the transition to an end of a process described by the block. When the processes of Auctioneer and all Buyers reach their end in the Main_process the transition to its end is executed.

Within the BDI architecture [13] agents are associated with beliefs (typically about the environment and other agents), desires or goals to achieve, and intentions or plans to act upon to achieve its desires. BDI agent consists of the sets of beliefs *B*, plans *P*, situations *E*, actions *A* and intentions *I*. When the agent notes a change in its environment, it decides, that there an event takes place representing some situation from *E*. Registration of the event consists in changing a state of the agents "mind": a choice of some belief from *B*. According to it and the desire (defined by some plan from *P*) the agent intends to execute some intention representing some sequence of actions from *A* to achieve the goal chosen from *I*. Thus, planned actions are defined by the chosen plan from *P*. After they are carried out, the current situation of the environment will be changed.

In traditional parallel programming languages the basic concepts are data and control of data calculation. Data are represented by values of variables, and the control is specified by a set of processes which transform local memory states defining variable values. The PRALU program of a BDI agent is a set of plans defining actions by means of which the agent should reach a goal of its functioning. The plan consists of head labels, a body and tail labels. The body of the plan is a sequence of actions, by means of which the agent should reach a goal of its functioning, and conditions which the agent should check up. The head and tail labels of the plan symbolize intentions. Critical concept for the behaviour of the agent is the concept of active intention. The plan will be carried out only when all intentions from its head are active. After executing the plan all intentions from the head become inactive and intentions from the tail quite the contrary become active. The current set of active intentions always is not empty, the body of a plan and a set of tail intentions can be empty.

6.2. The Programming Methodology

We suggest the natural methodology of designing agent program. It supposes splitting the program into two parts: a synchronization block and a functional block (Figure 6). The first block coordinates plan execution of the agent program, that is, it controls the agent behavior. The synchronization block should be described in PRALU

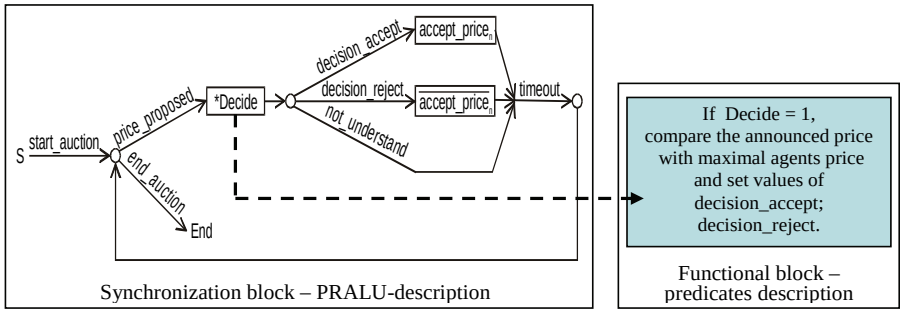


Figure 6. The program of the agent buyer in English auction

language. The functional block operates data and carries out calculations. This block is realized in procedural language.

The functional part is presented by predicates. The predicate is the conditions describing memory states of the agent program and the external environment or the order directing performance of some actions. The appropriate logic variable is introduced for each predicate in synchronization block.

Such an approach allows separating development of a synchronizing part of the PRALU-description from the functional one. When designing MAS, the implementation of the functional part can be delayed concentrating designer's attention on working out the synchronization block implementing interaction protocol on PRALU.

The PRALU-description compiled by PRALU compiler on the intermediate language applied in the simulation program [7] too. The program generated by the PRALU compiler consists of two interacting blocks– 1) calculation of reactions, 2) control of sensors and actuating mechanisms of the system. The heart of the control structure of the program of reactions calculation is an infinite loop; it consists of entering input signals into computer memory, calculating reactions and outputting signals to actuating mechanisms. The predicates of a functional part are considered as "the additional equipment" of MAS software in such model of the program organization. The program compiled from PRALU has semantics of measured time with of a rendezvous condition [5].

The majority of known systems for logic agent programming use model of calculations of Prolog. Prolog to find set of all decisions of a question, traverse a tree of search, tries many variants, which is not included into the decision, and comes back to earlier state in case of failure, trying other branch. This process is very expensive of space and time. Computing efficiency offered methodology is comparable to efficiency of the programs written in C.

7. Conclusions

The independent behaviour of agent in the majority of multi-agent system models is described by a formalism of high level abstractness, but the communications is specified by the concepts that close to realization. The difference of levels of the description does not allow to model the communications between agents at that level in which their independent behaviour is described. This problem arises because of absence of agent models that unify aspects of local behaviour and the communications.

In the present work we suggest to describe the synchronization conditions by specification of event properties which have been not concerned with the realization of these events. Our approach allows to specify both the independent behaviour and the communications at a level of logic. It is shown, that the collective behaviour of agents can be described by the synthetic temporal logic that unity the linear and branching time temporal logics. Such synthetic logic is an interpretation of PRALU language.

The suggested agent programming methodology ensures structuring the process of MAS designing by means of separating calculation part of MAS specification from control part. That allows the designer to concentrate on more complex stage of MAS designing – its interaction protocol. It is shown that language PRALU is very suitable for specifying interaction protocols of MAS and implementation of suggested methodology. Powerful theory and software has been developed that provides correctness verifying, simulation, hardware and software PRALU-description's implementation [6].

REFERENCES

[1] V. S. Subrahmanian, P. Bonatti, J. Dix, *et al.*, "Heterogeneous Agent Systems," MIT Press, 2000.

[2] S. D. Brookes, C. A. R. Hoare, and A. D. Roscoe, "A theory of communicating sequential processes," Journal of the ACM, Vol. 31, No. 3, pp. 560–599, 1984.

[3] S. D. Brookes, C. A. R. Hoare, and A. D. Roscoe, "A theory of communicating sequential processes," Journal of the ACM, Vol. 31, No. 3, pp. 560–599, 1984.

[4] C. A. R. Hoare, “Communicating sequential processes,” Prentice Hall International Series in Computer Science, 1985.

[5] R. Van Glabbeek and F. Vaandrager, “The difference between Splitting in *n* and *n+1*,” Report CS-R9553, Centre for Mathematics and Computer Science, Amsterdam 1995, Abstract in: Proceedings 3rd Workshop on Concurrency and Compositionality, Goslar, March 5-8, 1991 (E. Best & G. Rozenberg, eds.), GMD-Studien Nr. 191, Sankt Augustin, Germany 1991. Information and Computation, Vol. 136, No. 2, pp. 109–142, 1997.

[6] D. I. Cheremisinov, “The real difference between linear and branching temporal logics,” Workshop on Discrete-Event System Design DESDes’04, University of Zielona Gora Press, Poland, pp. 103–108, 2004.

[7] A. D. Zakrevski, “Parallel algorithms of logic control,” Minsk, Belarus, pp. 202, 1999. (in Russian)

[8] D. I. Cheremisinov and L. Cheremisinova, “Specifying agent interaction protocols with Parallel control algorithms,” Proceedings of 11th International Conference of Knowledge-Dialog-Solution (KDS’05), Varna, Bulgaria, pp. 496–503, June 20–30, 2005.

[9] R. Milner, “A calculus of communication systems,” LNCS’92, Springer Verlag, 1980.

[10] C. E. Shannon, “A mathematical theory of communication,” Bell System Technical Journal, Vol. 27, pp. 379–423, pp. 623–656, 1948.

[11] J. Odell, H. V. D. Parunak, M. Fleischer, and S. Brueckner, “Modeling agents and their environment,” Proceedings of the 3 International Workshop on Agent Oriented Software Engineering, Lecture Notes in Computer Science, Springer Verlag (Berlin, D), Vol. 2585, pp. 16–31, 2003.

[12] F. Finin, Y. Labrou and J. Mayfield, “KQML as an agent communication language,” edited by J. M. Bradshaw, Software Agents, MIT Press, pp. 291–316, 1997.

[13] Foundation for Intelligent Physical Agents (FIPA), http://www.fipa.org.

[14] V. Lesser, “Cooperative multiagent systems: A personal view of the state of the art,” in: IEEE Transactions of Knowledge and Data Engineering, Vol. 11, No. 1, pp. 133–142, 1999.

Intelligent Information Management, **2009, 1, 73-80**
doi:10.4236/iim.2009.12012 Published Online November 2009 (http://www.scirp.org/journal/iim)

Solving Linear Systems via Composition of their Clans

Dmitry A. ZAITSEV
Odessa National Telecommunication Academy, Kuznechnaya, Odessa, Ukraine
Email: *zaitsev_dmitry@mail.ru*

Abstract: The decomposition technique for linear system solution is proposed. This technique may be combined with any known method of linear system solution. An essential acceleration of computations is obtained. Methods of linear system solution depend on the set of numbers used. For integer and especially for natural numbers all the known methods are hard enough. In this case the decomposition technique described allows the exponential acceleration of computations.

Keywords: linear system, decomposition, clans, acceleration of computation

1. Introduction

The task of linear system solution is a classical task of linear algebra [10]. There are a variety of known methods for linear system solution. It should be noted that the choice of the method depends essentially on the set of numbers used. More precise it depends on the algebraic structure of the variables and coefficients. The solution of system in fields and bodies, for example, in rational or real numbers is the most studied. These algebraic structures allow the everywhere defined operation of division. It is convenient to develop the simple and powerful methods. Precise and approximate methods are distinguished. The most popular is Gauss method consisting in consequent elimination of variables and obtaining the triangular form of the matrix.

Ring algebraic structures such as integer numbers require special methods, as the division is not the everywhere-defined operation in a ring. There is known the universal method using unimodular transformations of matrix to obtain Smith normal form [10]. Result matrix has diagonal form and allow the simple representation of solutions. Unfortunately this method is exponential. Up to present time were suggested a few more complex but polynomial methods [3,7,8]. The most interesting is founded on consequent solution of system in the classes' fields of deductions modulo of prime numbers to construct a target general integer solution of the system [3].

The development of such areas of computer science as Petri net theory [6,13], logic programming [5], artificial intellect [1] require to solve integer systems over the set of nonnegative integer numbers. Notice that nonnegative integer numbers form algebraic structure of monoid. In monoid even the operation of subtraction is not everywhere defined. All the known methods of integer system solution in nonnegative integer numbers are exponential [2,4,9,12]. It makes significant difficulties in the application of these methods to real-life systems analysis.

For example, if the complexity of method is about 2^q, where q is the dimension of the system, then to solve the system with dimension 100 it is required about 10^{30} operations. A computer with the performance 10^9 operations per second will have executed this job in 10^{21} seconds or in about 10^{13} years.

Therefore, the task of the effective methods development for linear system solution, especially in nonnegative numbers, is enough significant. The goal of present paper is to introduce the decomposition technique of linear system solution. Application of this technique allows an essential acceleration of computations. In the case the exponential complexity of the source method of system solution the acceleration is exponential too.

2. Basic Concepts

Let us consider a homogeneous linear system

$$A \cdot \bar{x} = 0 \qquad (1)$$

where A is the matrix of coefficients with dimension of $m \times n$ and $\bar{x}$ is the vector-column of variables with dimension of n. So we have the system of m equations with n unknowns. We do not define now more precisely the sets of variables and coefficients. We suppose only that there is a method to solve a system (1) and to represent the general solution in the form

$$\bar{x} = \bar{y} \cdot G \qquad (2)$$

where matrix G is the matrix of basis solutions and $\bar{y}$

is the vector-row of independent variables. Each row of matrix G is a basis solution. To generate a particular solution of system the values of components of vector $\bar{y}$ may be chosen arbitrarily with respect to the set of values of variables $\bar{x}$. It should be noted that system (1) has always at least the trivial zero solution. For the brevity we shall name further homogeneous system an inconsistent in the case it has only the trivial solution.

Let us consider a nonhomogeneous system

$$A \cdot \bar{x} = \bar{b} \tag{3}$$

where $\bar{b}$ is the vector-column of free elements with dimension of m. We suppose also that there is a method to solve the system (3) and to represent the general solution in the form

$$\bar{x} = \bar{x}' + \bar{y} \cdot G \tag{4}$$

where $\bar{y} \cdot G$ is the general solution of corresponding homogenous system (1) and $\bar{x}'$ is the minimal particular solution of the nonhomogeneous system (3).

According to classical algebra [10], results represented above are true for arbitrary fields and rings. Moreover, according to [4], they are true also for monoid solutions with a ring structure of matrix A and vector $\bar{b}$. In this case the set of minimal solutions $\bar{x}'$ of nonhomogeneous system should be used.

3. Decomposition of System

Let us decompose the system (1) of homogeneous equations

$$A \cdot \bar{x} = 0$$

Above system may be considered as the predicate

$$L(\bar{x}) = L_1(\bar{x}) \wedge L_2(\bar{x}) \wedge ... \wedge L_m(\bar{x}) \tag{5}$$

where $L_i(\bar{x})$ are the equations of the system:

$$L_i(\bar{x}) = (\bar{a}^i \cdot \bar{x} = 0)$$

and $\bar{a}^i$ is the i-th row of the matrix A. We imply that $\bar{a}^i$ is a nonzero vector so at least one component of $\bar{a}^i$ is nonzero. Moreover, we shall denote as X the set of variables of system.

Let us consider the set of equations $\Im = \{L_i\}$ of system L. We shall introduce relations over the set $\Im$.

Definition 3.1. *Near relation:* two equations $L_i, L_j \in \Im$ are *near* and denoted as $L_i \mathbf{0} L_j$ if and only if $\exists x_k \in X$: $a_{i,k}, a_{j,k} \neq 0$, $sign(a_{i,k}) = sign(a_{j,k})$.

Lemma 3.1. Near relation is reflexive and symmetric.

Proof.

a) Reflexivity: $L_i \mathbf{0} L_i$. As $\bar{a}^i$ is a nonzero vector so there exists $x_k \in X$: $a_{i,k} \neq 0$. Then it is trivial that $sign(a_{i,k}) = sign(a_{i,k})$.

b) Symmetry: $L_i \mathbf{0} L_j \Rightarrow L_j \mathbf{0} L_i$. It is symmetric according to symmetry of equality sign: $sign(a_{i,k}) = sign(a_{j,k}) \Rightarrow sign(a_{j,k}) = sign(a_{i,k})$.

For each equation chosen we may construct the set of near equations. It is useful to consider the transitive closure of near relation. We introduce the special notation for this transitive closure.

Definition 3.2. *Clan relation:* two equations $L_i, L_j \in \Im$ *belong to the same clan* and are denoted as $L_i \mathbf{0} L_j$ if and only if there exists a sequence (may be empty) of equations $L_{l_1}, L_{l_2}, ..., L_{l_k}$ such that: $L_i \mathbf{0} L_{l_1} \mathbf{0} ... \mathbf{0} L_{l_k} \mathbf{0} L_j$.

Theorem 3.1. Clan relation is the equivalence relation.

Proof. We have to prove that relation above is reflexive, symmetric and transitive. We shall implement this in a constructive way.

a) Reflexivity: $L_i \mathbf{0} L_i$. As Definition 3.2 allows the empty sequence and near relation is reflexive according to Lemma 3.1, so clan relation is reflexive.

b) Symmetry: $L_i \mathbf{0} L_j \Rightarrow L_j \mathbf{0} L_i$. As $L_i \mathbf{0} L_j$, so, according to Definition 3.2, we may construct the sequence $L_{l_1}, L_{l_2}, ..., L_{l_k}$ such as $L_i \mathbf{0} L_{l_1} \mathbf{0} ... \mathbf{0} L_{l_k} \mathbf{0} L_j$. Since near relation is symmetry according to Lemma 3.1, we may construct the reversed sequence $L_{l_k}, L_{l_{k-1}}, ..., L_{l_1}$ such as $L_j \mathbf{0} L_{l_k} \mathbf{0} ... \mathbf{0} L_{l_1} \mathbf{0} L_i$, then $L_j \mathbf{0} L_i$.

c) Transitivity: $L_i \mathbf{0} L_j, L_j \mathbf{0} L_l \Rightarrow L_i \mathbf{0} L_l$. As $L_i \mathbf{0} L_j$, $L_j \mathbf{0} L_l$, so, according to Definition 3.2, we may construct two sequences $s = L_{i_1}, L_{i_2}, ..., L_{i_k}$ and $V = L_{l_1}, L_{l_2}, ..., L_{l_r}$ such that $L_i \mathbf{0} L_{i_1} \mathbf{0} ... \mathbf{0} L_{i_k} \mathbf{0} L_j$ and $L_j \mathbf{0} L_{l_1} \mathbf{0} ... \mathbf{0} L_{l_r} \mathbf{0} L_l$. Let us consider the concatenation $s\, L_j V$. This sequence connects the elements L_i and L_l with the chain of elements satisfying the near relation. Thus $L_i \mathbf{0} L_l$.

Corollary. Clan relation defines the partition of the set $\Im$: $\Im = \mathbf{U}_j C^j$, $C^i \mathbf{I} C^j = \varnothing$, $i \neq j$.

Definition 3.3. *Clan:* element of the partition $\{\Im, \mathbf{0}\}$ will be named a *clan* and denoted C^j. The variables $X^j = X(C^j) = \{x_i \,|\, x_i \in X, \exists L_k \in C^j : a_{k,i} \neq 0\}$ will be named *variables of clan* C^j.

Definition 3.4. *Internal variable:* variable $x_i \in X(C^j)$ is an *internal variable of the clan* C^j if and only if for

all the other clans C^l, $l \neq j$ it is true $x_i \notin X^l$. The set of internal variables of clan C^j we shall denote as $\overline{X}^j$.

Definition 3.5. *Contact variable:* variable $x_i \in X$ is a *contact variable* if and only if clans C^j and C^l exist such as $x_i \in X^j$, $x_i \in X^l$. The set of all the contact variables we shall denote as X^0. We shall also denote the set of contact variables of clan C^j as $\underline{X}^j$ so $X^j = \overline{X}^j \cup \underline{X}^j$ and $\overline{X}^j \cap \underline{X}^j = \varnothing$.

Lemma 3.3. An arbitrary contact variable $x_i \in X^0$ cannot appear in different clans with the same sign.

Proof. We shall assume the contrary. Let variable $x_i^* \in X$ appears in different clans $C^{j_1}, C^{j_2}, ..., C^{j_k}$ with the same sign, where $k > 1$. Thus, according to Definition 3.3, equations $L_{l_1} \in C^{j_1}$, $L_{l_2} \in C^{j_2}$, … , $L_{l_k} \in C^{j_k}$ exist such as $a_{l_1,i} \neq 0$, $a_{l_2,i} \neq 0$,…, $a_{l_k,i} \neq 0$ and $sign(a_{l_1,i}) = sign(a_{l_2,i}) = ... = sign(a_{l_k,i})$. Then, according to Definitions 3.2 and 3.3, equations $L_{l_1}, L_{l_2}, ..., L_{l_k}$ belong to the same clan. So we have obtained the contradiction.

Lemma 3.4. An arbitrary contact variable $x_i \in X^0$ belongs to two clans exactly.

Proof. We shall implement the proof from the contrary. Let us assume that a contact variable $x_i \in X$ belongs to q different clans and $q \neq 2$. We shall consider separately two cases:

a) $q < 2$. Than, according to Definition 3.5, variable x_i is not a contact variable.

b) $q > 2$ We have contradiction with Lemma 3.3 so there are only two signs: plus and minus.

Corollary. An arbitrary contact variable $x_i \in X^0$ is contained in different clans with opposite signs.

Definition 3.6. Clan C^j will be named an *input clan of the contact variable* $x_i \in X^0$ and denoted as $I(x_i)$ if and only if it contains this variable with the sign plus.

Definition 3.7. Clan C^j will be named an *output clan of the contact variable* $x_i \in X^0$ and denoted as $O(x_i)$ if and only if it contains this variable with the sign minus.

It should be noted that analogous classification into input and output might be introduced for the contact variables of the clan.

Thus we have obtained on the one hand the partition of equations into clans and on the other hand the partition of variables into internal and contact. Let us make a new enumeration of equations and variables. The enumeration of equations we start from the equations of the first clan and so on up to the last clan. The enumeration of variables we start from the contact variables and than continue with internal variables of clans in ascending order. Then we order sets X and $\Im$ according to the new enumeration.

As a result we obtain a block form of matrix A:

$$A = \begin{Vmatrix} A^{0,1} & \overline{A}^1 & 0 & 0 & 0 \\ A^{0,2} & 0 & \overline{A}^2 & 0 & 0 \\ \mathbb{M} & \mathbb{M} & \mathbb{M} & \mathbb{M} & \mathbb{M} \\ A^{0,k} & 0 & 0 & 0 & \overline{A}^k \end{Vmatrix}$$

With respect to clans and subsets of variables it may be more convenient to represent the matrix in Table 1.

Let us consider more detailed the structure of matrixes for the contact variables. According to Definition 3.5, $\underline{X}^j$ denotes the contact variables of the clan C^j. Then we have $X^0 = \bigcup_j \underline{X}^j$ but this is not a partition of the set X^0 as each contact variable $x_i \in X^0$, according to Lemma 3.4, belongs to two clans exactly. Further we shall use matrixes $\overline{A}^j$ also. Notice that unlike of $A^{0,j}$ that contains values for all contact places X^0 the matrix $\overline{A}^j$ contains values only for contact variables $\underline{X}^j$ of the clan C^j. In other words the matrix $\overline{A}^j$ contains only nonzero columns of the matrix $A^{0,j}$.

As a result of Lemma 3.4 application to matrix A we conclude that each contact variable $x_i \in X^0$ appears in

Table 1. The block structure of the system matrix

Clan/Variables	X^0	$\overline{X}^1$	$\overline{X}^2$	…	$\overline{X}^k$
C^1	$A^{0,1}$	$\overline{A}^1$	0	…	0
C^2	$A^{0,2}$	0	$\overline{A}^2$	…	0
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
C^k	$A^{0,k}$	0	0	…	$\overline{A}^k$

two of matrixes $A^{0,j}$, $j=\overline{1,k}$ exactly and besides, it appears in one matrix with positive coefficients and in another matrix with negative coefficients.

4. Solution of Decomposed System

At first we shall solve the system separately for each clan. If we consider only variables of the clan we have the system

$$A^j \cdot \bar{x}^j = 0 \tag{6}$$

where

$$A^j = \left\| \begin{matrix} & & \\ A^j & & A^j \\ & & \end{matrix} \right\|, \quad \bar{x}^j = \left\| \begin{matrix} \\ \bar{x}^j \\ \\ \bar{x}^j \\ \\ \end{matrix} \right\|.$$

System (6) will be denoted also as $L^{C^j}(\bar{x})$. Notice that values of variables $X \setminus X^j$ may be chosen arbitrarily. More detailed

$$L^{C_j}(\bar{x}) = \mathop{\&}_{L_l \in C^j} L_l(\bar{x})$$

Theorem 4.1. If system (1) has a nontrivial solution then each system (6) has a nontrivial solution.

Proof. Let us consider the representation (5) of the system (1). As, according to Theorem 3.1, clan relation defines a partition of the set of equations, so representation (5) may be written in the form

$$L(\bar{x}) = L^{C_1}(\bar{x}) \wedge L^{C_2}(\bar{x}) \wedge ... \wedge L^{C_k}(\bar{x}) \tag{7}$$

Thus, an arbitrary solution of (1) is the solution of each system (6).

Corollary. If at least one of systems (6) is inconsistent then the entire system (1) is inconsistent.

Let us the general solution of the system (6), according to (2), has the form

$$\bar{x}^j = \bar{y}^j \cdot G^j$$

As each $x_i \in X^j$ is contained exactly in one system (6), so for all the internal variables of clans we have

$$\left\| \begin{matrix} \\ \bar{x}^j \\ \\ \end{matrix} \right\| = \bar{y}^j \cdot \left\| \begin{matrix} \\ G^j \\ \\ \end{matrix} \right\|$$

Each contact variable, according to Lemma 3.4, belongs to two systems exactly. Thus, its values have to be equal

$$\bar{x}_i^j = \bar{x}_i^l \text{ or } \bar{y}^j \cdot G_i^j = \bar{y}^l \cdot G_i^l,$$

where G_i^j denotes the column of matrix G^j corresponding to variable x_i.

Therefore, we obtain the system

$$\begin{cases} \bar{x}^j = \bar{y}^j \cdot G^j, \quad j=\overline{1,k}, \\ \bar{y}^j \cdot G_i^j = \bar{y}^l \cdot G_i^l, \quad x_i \in X^0, \quad C^j = O(x_i), \quad C^l = I(x_i). \end{cases} \tag{8}$$

Theorem 4.2. System (8) is equivalent to the system (1).

Proof. As Expression (7) is the equivalent representation of the source system (1) and there is a partition of the set of variables X into contact and internal, so above reasoning proves the theorem.

Let us consider more detailed the equations of system (8) for the contact variables. Equation

$$\bar{y}^j \cdot G_i^j = \bar{y}^l \cdot G_i^l$$

may be represented in a block form as

$$\left\| \bar{y}^j \quad \bar{y}^l \right\| \cdot \left\| \begin{matrix} G_i^j \\ -G_i^l \end{matrix} \right\| = 0$$

We enumerate all the variables $\bar{y}^j$ in such a manner to obtain the joint vector

$$\bar{y} = \left\| \bar{y}^1 \quad \bar{y}^2 \quad ... \quad \bar{y}^k \right\|$$

and assemble G_i^j, G_i^l into the joint matrix F. Thus, we obtain the system

$$\bar{y} \cdot F = 0 \quad \text{or} \quad F^T \cdot \bar{y}^T = 0 \tag{9}$$

System obtained has the form (1) so the general solution of system has the form (2):

$$\bar{y} = \bar{z} \cdot R \tag{10}$$

Let us compose the joint matrix G of systems' (6) solutions for all the clans in such a manner that

$$\bar{x} = \bar{y} \cdot G \tag{11}$$

This matrix has a block structure as follows

$$G = \left\| \begin{matrix} S^1 & \left\| \begin{matrix} \\ G^1 \\ \\ \end{matrix} \right\| & 0 & 0 & 0 \\ S^2 & 0 & \left\| \begin{matrix} \\ G^2 \\ \\ \end{matrix} \right\| & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S^k & 0 & 0 & 0 & \left\| \begin{matrix} \\ G^k \\ \\ \end{matrix} \right\| \end{matrix} \right\|$$

The structure of the first column of above block representation that corresponds to contact variables is complex enough. For each contact variable $x_i \in X^0$ we create the column so that either block S^j or S^l contain nonzero elements according to $\left\| \begin{matrix} \\ G^j \\ \\ \end{matrix} \right\|$ or $\left\| \begin{matrix} \\ G^l \\ \\ \end{matrix} \right\|$ where $C^j = O(x_i)$, $C^l = I(x_i)$. Really, as a contact variable belong to two clans, so its value may be calculated either according to general solution of input clan or according to general solution of output clan.

Let us substitute (10) into (11):

$$\bar{x} = \bar{z} \cdot R \cdot G$$

Therefore

$$\bar{x} = \bar{z} \cdot H, \quad H = R \cdot G, \tag{12}$$

Theorem 4.3. Expressions (12) represent the general

solution of homogeneous system (1).

Proof. As only equivalent transformations were used, so above reasoning proves the theorem.

Let us implement analogous transformations for a nonhomogeneous system (3). The general solution for each clan, according to (4), has the form

$$\overline{x}^j = \overline{x}'^j + \overline{y}^j \cdot G^j$$

Equations for contact variables may be represented as follows

$$\overline{x}_i'^j + \overline{y}^j \cdot G_i^j = \overline{x}_i'^l + \overline{y}^l \cdot G_i^l$$

and further

$$\overline{y}^j \cdot G_i^j - \overline{y}^l \cdot G_i^l = \overline{b}_i', \quad \overline{b}_i' = \overline{x}_i'^l - \overline{x}_i'^j$$

or in the joint matrix form

$$\overline{y} \cdot F = \overline{b}' \quad \text{or} \quad F^T \cdot \overline{y}^T = \overline{b}'^T .$$

The general solution of above system, according to (4), may be represented as

$$\overline{y} = \overline{y}' + \overline{z} \cdot R .$$

Using the joint matrix G we may write

$$\overline{x} = \overline{x}' + \overline{y} \cdot G$$

or

$$\overline{x} = \overline{x}' + \left(\overline{y}' + \overline{z} \cdot R\right) \cdot G = \overline{x}' + \overline{y}' \cdot G + \overline{z} \cdot R \cdot G$$

and finally

$$\overline{x} = \overline{y}'' + \overline{z} \cdot H , \quad \overline{y}'' = \overline{x}' + \overline{y}' \cdot G , \quad H = R \cdot G . \qquad (13)$$

Theorem 4.4. Expressions (13) represent the general solution of nonhomogeneous system (2).

Proof. As only equivalent transformations were used, so above reasoning proves the theorem.

5. Description of Algorithm

Solution of linear homogeneous system of Equations (1) with decomposition technique consists in:

Stage 1. Decompose system (1) into set of clans: $\{C^j\}$.

Stage 2. Solve the system (6): $\overline{x}^j = \overline{y}^j \cdot G^j$ for each clan C^j. If at least one of systems (6) is inconsistent, then the entire system (1) is inconsistent. Stop.

Stage 3. Construct and solve system (9) for contact variables: $\overline{y} = \overline{z} \cdot R$. If this system is inconsistent, then the entire system (1) is inconsistent. Stop.

Stage 4. Compose matrix G and calculate matrix H of basis solutions: $H = R \cdot G$. Stop.

It should be noted that Stages 2, 3 use a known method of linear system solution. The choice of this method is defined by the sets of numbers used. For example, this is Gauss method for rational numbers, unimodular transformations for integer numbers and Toudic method for nonnegative integer solutions. Moreover, analogous technique, according to Theorem 4.4, may be implemented for nonhomogeneous system.

It has to describe more precisely now the decomposition algorithm used at Stage 1. Let us q is a maximum dimension of matrix A: $q = \max(m, n)$. Clan relation may be calculated in a standard way, as transitive closure of near relation but this way is hard enough from the computational point of view.

We propose the following algorithm of decomposition (Figure 1) with a total complexity about q^3:

```
j := 1;
while ℑ ≠ ∅
do
    C^j := ∅;
    curC := L, (L ∈ ℑ);
    ℑ := ℑ \ L;
    do
    newC := ∅;
    for L′ in curC
    for L″ in ℑ
    if L′ o L″ then do ℑ := ℑ \ L″; newC := newC U L″; od;
    C^j := C^j U curC;
    curC := newC;
    od until newC = ∅
    j := j + 1;
od;
```

Figure 1. Algorithm of system decomposition

It creates clans C^j out of the source set of equations $\Im$ of the system. To create the transitive closure the algorithm compares each equation of the set *curC* with each equation of $\Im$. Equations included into current clan are extracted from the set $\Im$. The usage of two auxiliary subsets *curC* and *newC* allow the once comparison of each pair of equations. As the calculation of near relation is linear, so the total complexity of the algorithm is about q^3.

Let us $M(q)$ is the time complexity of solution for linear system of size q. We shall estimate the total complexity of linear system solution with decomposition technique. Let us the source system (1) is decomposed in k clans and p is the maximal number of either contact or clans' internal variables. Thus $q \le (k+1)\cdot p$. The following expression estimates the complexity of system solution with decomposition technique:

$$V(q) \le ((k+1)\cdot p)^3 + k\cdot M(p) + M(p) + ((k+1)\cdot p)^3.$$

Each of four addends of this expression estimates the complexity of the corresponding stage. In more simplified form it is represented as

$$V(q) \approx 2\cdot(k+1)^3\cdot p^3 + (k+1)\cdot M(p) \approx k^3\cdot p^3 + k\cdot M(p).$$

Let us estimate the acceleration of computations under the decomposition technique usage. The target expression is

$$Acc(q) = \frac{M(q)}{k^3\cdot p^3 + k\cdot M(p)}$$

It is enough clear that even for polynomial methods of system solution with a degree higher than 3 we obtain acceleration greater than one. Now we estimate the acceleration for exponential methods with complexity $M(q) = 2^q$ such, for example, as Toudic method [9,12].

$$AccE(q) = \frac{2^q}{k^3\cdot p^3 + k\cdot 2^p} \approx \frac{2^q}{2^p} = 2^{q-p}.$$

The acceleration obtained is exponential. It is good enough result.

6. Example

Let us solve the homogeneous system (1) of 9 equations with 10 variables. The matrix of the system is as follows

$$A = \begin{Vmatrix} 1 & 0 & 2 & 0 & 0 & -1 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 2 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \end{Vmatrix}$$

Stage 1. *Decomposition.* Above system is decomposed into two clans

$$C^1 = \{L_1, L_2, L_5, L_6\},\ C^2 = \{L_3, L_4, L_7, L_8, L_9\}.$$

It has the following subsets of clans' variables

$$X^1 = \{x_3, x_6, x_8, x_{10}, x_1, x_2, x_7\},$$

$$X^2 = \{x_3, x_6, x_8, x_{10}, x_4, x_5, x_9\},$$

contact variables

$$X^0 = X^1 = X^2 = \{x_3, x_6, x_8, x_{10}\}$$

and internal variables

$$\overline{X}^1 = \{x_1, x_2, x_7\},\ \overline{X}^2 = \{x_4, x_5, x_9\}.$$

According to new enumeration of variables

$$nx = (3\ \ 6\ \ 8\ \ 10\ \ 1\ \ 2\ \ 7\ \ 4\ \ 5\ \ 9)$$

and new enumeration of equations

$$nL = (1\ \ 2\ \ 5\ \ 6\ \ 3\ \ 4\ \ 7\ \ 8\ \ 9)$$

the matrix A has the form

$$A = \begin{Vmatrix} 2 & -1 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & 0 & -1 & 1 & 0 & 0 & 0 \\ -1 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 2 & -1 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \end{Vmatrix}$$

Stage 2. *Solution of systems for clans.* Application of Toudic method gives us the following matrixes of basis solutions for clans:

$$G^1 = \begin{Vmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{Vmatrix},$$

$$G^2 = \begin{Vmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{Vmatrix}$$

with respect to vectors of free variables $\bar{y}^1 = (y_1^1, y_2^1, y_3^1)$ and $\bar{y}^2 = (y_1^2, y_2^2)$ correspondingly.

Stage 3. *Solution of system for contact variables.* The system for contact variables has the form:

$$\begin{cases} y_1^1 - y_1^2 = 0, \\ y_1^1 - y_1^2 = 0, \\ y_2^1 - y_1^2 = 0, \\ y_2^1 - y_1^2 = 0. \end{cases}$$

It should be noted that equations correspond to contact variables x_3, x_6, x_8, x_{10} and first and second equations coincide as well as third and fourth. The general solution of this system is

$$R = \begin{Vmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{Vmatrix}$$

Stage 4. *Composition of target solution.* As it was mentioned in section 4, matrix G may be composed in different ways. Two variants of this matrix are represented as follows

$$G = \begin{Vmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{Vmatrix} \text{ or}$$

$$G = \begin{Vmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{Vmatrix}.$$

And finally the result matrix of basis solutions of the system (1) has the form

$$H = R \cdot G = \begin{Vmatrix} 1 & 1 & 1 & 1 & 0 & 2 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{Vmatrix}.$$

The result obtained coincides with the basis solutions calculated in usual manner with Toudic method.

7. Conclusions

Present paper introduced new technique of linear system solution with decomposition into clans. It is efficient in the combination with methods of system solution more hard than a polynomial of third degree and in the case the system may be decomposed in more than one clan.

As a result of this technique application to various matrixes it may be concluded that a good opportunity to decomposition have sparse matrixes. This is realistic enough situation as large-scale real-life models contain well-localized interconnections of elements.

It should be noted also the relationship of technique proposed with the decomposition of Petri nets [11]. We may construct the matrix of directions D for an arbitrary matrix A of linear system as $D = sign(A)$. Matrix D may be considered further as incidence matrix of Petri net.

The most significant acceleration of computations was obtained for Diophantine (integer) systems especially solving over the set of nonnegative numbers. There are only exponential methods for solution of such systems. In this case the acceleration of computation obtained is exponential.

REFERENCES

[1] F. Baader and J. Ziekmann, "Unification theory," Handbook of logic in artificial intelligence and logic programming, Oxford: Univ. Press, pp. 1–85, 1994.

[2] E. Contejan and F. Ajili, "Avoiding slack variables in solving linear Diophantine equations and inequations," Theoretical Computer Science, Vol. 173, pp. 183–208, 1997.

[3] M. A. Frumkin, "Algorithm of system of linear equations solution in integer numbers," Research on Discrete Optimisation, Moskow, Nauka, pp. 97–127, 1976.

[4] S. L. Kryviy, "Methods of solution and criteria of compatibility of systems of linear Diophantine equations over the set of natural numbers," Cybernetics and Systems Analysis, Vol. 4, pp. 12–36, 1999.

[5] J. Lloyd, "Foundation of logic programming," Berlin, Springer-Verlag, 1987.

[6] T. Murata, "Petri nets: Properties, analysis and applications," Proceedings of IEEE, Vol. 77, No. 4, pp. 541–580 1989.

[7] L. Pottier, "Minimal solutions of linear diophantine systems: bounds and algorithms," Proceedings of the Fourth Intern. Conference on Rewriting Technique and Application, Como, Italy, pp. 162–173, 1991.

[8] J. F. Romeuf, "A polynomial algorithm for solving systems of two linear Diophantine equations," Theoretical Computer Science, Vol. 74, No. 3, pp. 329–340, 1990.

[9] J. M. Toudic, "Linear algebra algorithms for the structural analysis of petri nets," Rev. Tech. Thomson CSF, Vol. 14, No. 1, pp. 136–156, 1982.

[10] B. L. Van Der Warden, Algebra, Berlin, Heidelberg,

Springer-Verlag, 1971.

[11] D. A. Zaitsev, "Subnets with input and output places," Petri Net Newsletter, Vol. 64, pp. 3–6, 2003.

[12] D. A. Zaitsev, "Formal grounding of toudic mmethod," Proceedings of the 10th Workshop "Algorithms and Tools for Petri Nets", Eichstaett, Germany, pp. 184–190, September 26-27, 2003.

[13] D. A. Zaitsev and A. I. Sleptsov, "State equations and equivalent transformations of timed petri nets," Cybernetics and System Analysis, Vol. 33, No. 5, pp. 659–672, 1997.

Intelligent Information Management, **2009, 1, 81-88**
doi:10.4236/iim.2009.12013 Published Online November 2009 (http://www.scirp.org/journal/iim)

Scientific Research

A Quantity Model for Controlling and Measuring Software Quality Based on the Expert Decision-Making Algorithm

Che-Wei CHANG[1], Der-Juinn HORNG[2], Hung-Lung LIN[2]
[1]*Department of Information Management, Yuanpei University, Hsin Chu, Taiwan, China*
[2]*Department of Business Administration, National Central University, Jhongli City, Taiwan, China*
Email: *chewei@mail.ypu.edu.tw, horng@cc.ncu.edu.tw, hsa*8936.*hsa*8936*@msa.hinet.net*

Abstract: Researchers have been active in the field of software engineering measurement over more than 30 years. The software quality product is becoming increasingly important in the computerized society. Target setting in software quality function and usability deployment are essential since they are directly related to development of high quality products with high customer satisfaction. Software quality can be measured as the degree to which a particular software program complies with consumer demand regarding function and characteristics. Target setting is usually subjective in practice, which is unscientific. Therefore, this study proposes a quantity model for controlling and measuring software quality via the expert decision-making algorithm-based method for constructing an evaluation method can provide software in relation to users and purchasers, thus enabling administrators or decision makers to identify the most appropriate software quality. Importantly, the proposed model can provide s users and purchasers a reference material, making it highly applicable for academic and government purposes.

Keywords: software quality characteristics, software quality model, multiple criteria decision making (MCDM), analytic hierarchy process (AHP)

1. Introduction

The numerous challenges involved in software development include devising quality software, accurately controlling overhead costs, complying with a progress schedule, maintaining the software system, coping with unstable software systems and satisfying consumer demand with respect to software quality [1–3]. These challenges may incur a software development crisis if performed inefficiently. Problems within the software sector can be summed up as follows: 1) Inability to accurately forecast or control software development costs, 2) substandard quality, poor reliability and ambiguous requests on how to enhance requests by management directives regarding, 3) unnecessary risks while offering and maintaining quality assurance, and 4) high personnel turnover rate, leading to lack of continuity and increased incidence of defects in software development [4].

Researchers have been active in the field of software engineering measurement over more than 30 years. The software quality product is becoming increasingly important in the computerized society. Developing software quality is complex and difficult, and so a firm must maintain the software quality to gain a competitive advantage. However, when software quality is being developed, developers must simultaneously consider developmental budget, the schedule, the ease of maintenance of the system and the user's requirements. Users' requirements and the management of software firms together determine the implementation of software systems.

Target setting in software quality function and usability deployment are essential since they are directly related to development of high quality products with high customer satisfaction. Software quality can be measured as the degree to which a particular software program complies with consumer demand regarding function and characteristics (degree of conformity to requirements). However, target setting is usually subjective in practice, which is unscientific [5,6]. Therefore, this study proposes a quantity model for controlling and measuring software quality via the expert decision-making algorithm-based method for constructing an evaluation method can provide software in relation to users and purchasers, thus enabling administrators or decision makers to identify the most appropriate software quality. Importantly, the proposed model can provide s users and purchasers a reference material, making it highly applicable for academic and government purposes.

2. Model for the Software Quality

Conversely, capability assessment frameworks usually assess the process that is followed on a project in practice in the context of a process reference model, defined separately and independently of any particular methodology [7]. Software architecture is a key asset in any organization that builds complex software-intensive systems. Because of its central role as a project blueprint, organizations should first analyze the architecture before committing resources to a software development program [8]. To resolve the above problems, multi-attribute characteristics or factors of software quality must be considered. Effective management strategies in a technology setting are thus essential for resolving crises in software development. Technological aspects are software technology adopted and design expertise of a software developer [4].

Multiple criteria decision making (MCDM) is a methodology that helps decision makers make preference decisions (e.g. assessment, ranking, selection) regarding a finite set of available alternatives (courses of action) characterized by multiple, potentially conflicting attributes [9,10]. MCDM provides a formal framework for modeling multi-attribute decision problems, particularly problems whose nature demands systematic analysis, including analysis of decision complexity, regularity, significant consequences, and the need for accountability [9]. Among those well-known methods, MCDM has only relatively recently been employed to evaluate software quality performance. MCDM-based decision-making is a wide method for the measuring the software quality [4,11–13]. Among those well-known evaluation methods, MCDM has been employed relatively recently to evaluate organizational performance and it uses a set of attributes to resolve decision-making issues. Currently, one of the most popular existing evaluation techniques was performed by adopting the analytic hierarchy process (AHP), which was utilized by setting up hierarchical or skeleton within which multi-attribute decision problems can be structured [13–18].

2.1. Analytic Hierarchic Process (AHP)

Assume that we have n different and independent criteria (C_1, C_2, …, C_n) and they have the weights (W_1, W_2, …, W_n), respectively. The decision-maker does not know in advance the values of W_i, i = 1, 2, …, n, but he is capable of making pair-wise comparison between the different criteria. Also, assume that the quantified judgments provided by the decision-maker on pairs of criteria (C_i, C_j) are represented in an $n \times n$ matrix as in the following:

$$A = [a_{ij}] = \begin{array}{c} \\ C_1 \\ C_2 \\ \vdots \\ C_n \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & C_2 & \cdots & C_n \end{array} \\ \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \end{array}, \tag{1}$$

If for example the decision-maker compares C_1 with C_2, he provides a numerical value judgment a_{12} which should represent the importance intensity of C_1 over C_2. The a_{12} value is supposed to be an approximation of the relative importance of C_1 to C_2; i.e., $a_{12} \approx (W_1 / W_2)$. This can be generalized and the following can be concluded:

1) $a_{12} \approx (W_1 / W_2)$ $i, j = 1, 2, ..., n.$
2) $a_{ii} = 1,\ i = 1,\ 2,\ ...,\ n.$
3) If $a_{ij} = \delta,\ \delta \neq 0,$ than $a_{ji} = 1/\delta,\ i = 1,\ 2,\ ...,\ n.$
4) If C_i is more important than C_j, than $a_{12} \approx (W_1 / W_2) > 1$.

This implies that matrix *A* should be a positive and reciprocal matrix with 1's in the main diagonal and hence the decision-maker needs only to provide value judgments in the upper triangle of the matrix. The values assigned to a_{ij} according to Saaty scale are usually in the interval of 1 - 9 or their reciprocals. Table 1 presents Saaty's scale of preferences in the pair-wise comparison process. It can be shown that the number of judgments (L) needed in the upper triangle of the matrix are:

$$L = n(n-1)/2, \tag{2}$$

where n is the size of the matrix A:

Having recorded the numerical judgments aij in the matrix *A*, the problem now is to recover the numerical weights (W_1, W_2, …, W_n) of the criteria from this matrix. In order to do so, consider the following equation:

$$\begin{bmatrix} \tilde{a}_{11} & \tilde{a}_{12} & \cdots & \tilde{a}_{1n} \\ \tilde{a}_{21} & \tilde{a}_{22} & \cdots & \tilde{a}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{a}_{n1} & \tilde{a}_{n2} & \cdots & \tilde{a}_{nn} \end{bmatrix}$$

$$\approx \begin{bmatrix} W_1/W_1 & W_1/W_2 & \cdots & W_1/W_n \\ W_2/W_1 & W_2/W_2 & \cdots & W_2/W_n \\ \vdots & \vdots & \ddots & \vdots \\ W_n/W_1 & W_n/W_2 & \cdots & W_n/W_n \end{bmatrix}. \tag{3}$$

Moreover, by multiplying both matrices in Equation (3) on the right with the weights vector $W = (W_1, W_2, ..., W_n)$, where *W* is a column vector. The result of the multiplication of the matrix of pair-wise ratios with *W* is *nW*, hence it follows:

$$A \times W = n \times W. \tag{4}$$

This is a system of homogenous linear equations. It has a non-trivial solution if and only if the determinant of $A - nI$ vanishes, that is, n is an eigenvalue of *A*. *I* is an $n \times n$ identity matrix. Saaty's method computes *W* as the principal right eigenvector of the matrix *A*, that is,

Table 1. Saaty's scale of preferences in the pair-wise comparison process

Numerical	Verbal judgments of preferences between C_i and C_j
1	*i* is equally important to *j*
3	*i* is slightly more important than *j*
5	*i* is strongly more important than *j*
7	*i* is very strongly more important than *j*
9	*i* is extremely more important than *j*
2, 4, 6, 8	Intermediate values

Table 2. Average random index for corresponding matrix size

(*n*)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
(*R.I.*)	0	0	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51	1.48	1.56	1.57	1.59

$$A \times W = \lambda_{max}\ W \tag{5}$$

where λ_{max} is the principal eigenvalue of the matrix *A*.

If matrix *A* is a positive reciprocal one then $\lambda_{max} \geq n$, [19]. The judgments of the decision-maker are perfectly consistent as long as

$$a_{ij}a_{jk} = a_{ik},\ i,\ j,\ k = 1,\ 2,\ ...,\ n, \tag{6}$$

which is equivalent to

$$(W_i / W_j)(W_j / W_k) = (W_i / W_k), \tag{7}$$

The eigenvector method yields a natural measure of consistency. Saaty defined the consistency index (*C.I.*) as

$$C.I. = (\lambda_{max} - n) / (n - 1). \tag{8}$$

For each size of matrix *n*; random matrices were generated and their mean *C.I.* value, called the random index (*R.I.*), was computed and tabulated as shown in Table 2. Accordingly, Saaty defined the consistency ratio as

$$C.R. = C.I / R.I.. \tag{9}$$

The consistency ratio *C.R.* is a measure of how a given matrix compares to a purely random matrix in terms of their consistency indices. A value of the consistency ratio $C.R. \leq 0.1$ is considered acceptable. Larger values of *C.R.* require the decision-maker to revise his judgments.

3. Case Implementation

The city government of Hsinchu, in northern Taiwan, intends to implement a system for monitoring public spaces. Hence, the Hsinchu City Government is installing digital video recorder systems (DVRs). in place of traditional surveillance camera systems. The DVRs circumvents restrictions corrects the limitations of traditional surveillance camera systems, which include: a) lapses in recording due to operator neglect or machine error; b) difficulty in locating a desired time sequence following completion of a recording; c) poor video quality and d) difficulty in maintaining and preserving tapes due to a lack storage space and natural degradation of film quality.

According to government procurement regulations, regional governments must select at least five evaluators to review more than three firms before evaluating the best DVRs software quality. This study considers four candidate DVRs software packages common in the surveillance market manufactured by Firms A, B, C and D.

As Figure 1 shows, the ISO 9126-1 standard was developed in 2001 not only to identify major quality attributes of computer software, but also as a measure of six major quality attributes. The proposed method adopts the ISO 9126-1 model to evaluate the DVRs software quality. The applicability of the proposed model is demonstrated in a case study. This model for evaluating the DVRs software quality comprises the following steps.

3.1. Step 1: Establish an Evaluation Model and Define the Criteria

Evaluate the ideal model, as six evaluation criteria, twenty-one sub-criteria and, finally, comparison with four alternatives (Figure 1). The evaluation criteria and sub-criteria used to evaluate the DVRs software quality are defined as follows:

■ Functionality (C_1): The degree to which the software satisfies stated requirements, including the four sub-criteria of suitability, accuracy, interoperability and security.

- Suitability: capability of software to provide an appropriate set of functions for specified tasks and user objectives.
- Maturity: capability of software to avert failure caused by software defects.

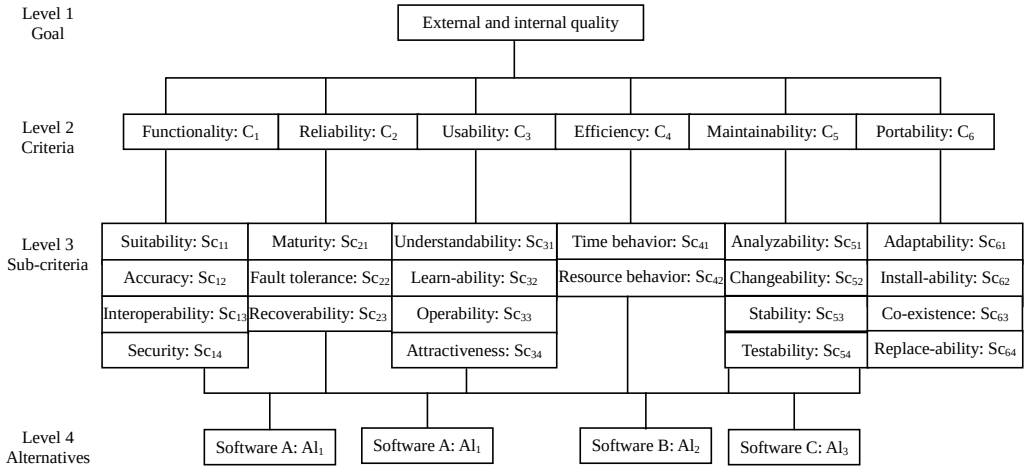


Figure 1. The ISO 9126-1 evaluate model

- Fault tolerance: capability of software to maintain a specified performance level in case of software errors or infringement of its specified interface.
- Accuracy: capability of software to provide correct or anticipated results or effects.
- Interoperability: capability of software to interact with one or more specified systems.
- Security: capability of software to prevent prohibited access and withstand deliberate attacks intended to gain unauthorized access to confidential information, or to make unauthorized access.

■ Reliability (C_2): How long the software is available for use; this includes the three sub-criteria of maturity, fault tolerance, and recoverability.

- Recoverability: capability of software to re-establish its level of performance and recover the data directly affected if a failure occurs.

■ Usability (C_3): Ease of implementation, including the four sub-criteria of understandability, learn-ability, operability and attractiveness.

- Understandability: capability of software to enable users to understand the appropriateness of a software and its use for particular tasks and conditions of use.
- Learning ability: capability of software to enable users to learn its application.
- Operability: capability of software to enable users to operate and control it.
- Attractiveness: capability of software to gain user acceptance.

■ Efficiency (C_4): Optimal use of system resources, including the two sub-criteria of time behavior and resource behavior.

- Time behavior: capability of software to provide appropriate responses, processing times and throughput rates when performing its function under stated conditions.
- Resource behavior: capability of software to use appropriate resources in time when the software implements its function under stated conditions.

■ Maintainability (C_5): The ease of which repairs can be made to the software, including the four sub-criteria of analyzability, changeability, stability and testability.

- Analyzability: capability of software to be diagnosed for deficiencies or causes of failures in the software or for identification of parts requiring modification.
- Changeability: capability of software to enable a specified modification to be implemented.
- Stability: ability of software to minimize unexpected effects from software modifications.
- Testability: ability of software to validate modified software.

■ Portability (C_6): How easily the software can be transposed from one environment to another; including four sub-criteria of adaptability install ability, co existence and replace ability.

- Adaptability: capability of software to be modified for specified environments without applying actions or means other than those provided for the software considered.
- Install-ability: capability of software to be installed in a specified environment.
- Co-existence: capability of software to co-exist with other independent software in a common environment sharing common resources.
- Replace-ability: capability of software to replace other specified software in the environment of that software.

3.2. Step 2: Establish the Pair-Wise Comparison Matrix and Determine Consistency

Twenty one experts are assigned and rated on a nine-point scale against each criterion to assess criteria and sub-criteria. The experts were proficient in PC modules,

software modules and network communication modules. The program adopted of the respondents' data to cross-compare all criteria and alternatives to determine the weights and inconsistency ratios. The inconsistency ratio is a measure of the percentage of time when decision makers are inconsistent in making judgment. The "acceptable" inconsistency ratio was approximately 0.1 or less, but "particular circumstance" may warrant the acceptance of a higher value. However, an inconsistency ratio of 1 is unacceptable because the ratings are as good as random judgments. Four of the twenty one experts had inconsistency ratios above 0.15. This was too high and their responses were discarded. Of the remaining seventeen, ten experts had low inconsistency ratios (<0.05), and seven had ratios between 0.12 and 0.14. These seven respondents were each given another chance to recheck at their ratings and determine whether they would like to modify their decisions, and eventually modified their rating by making their own adjustments to the data. Chang *et al.* proposal determining consistency procedure as shows Figure 2.

After discarding ht responses of the four inconsistent experts, the weights were then determined for a sample group of seventeen individuals matching the above characteristics with each respondent, making a pair-wise comparison of the decision elements and assigning them relative scores.

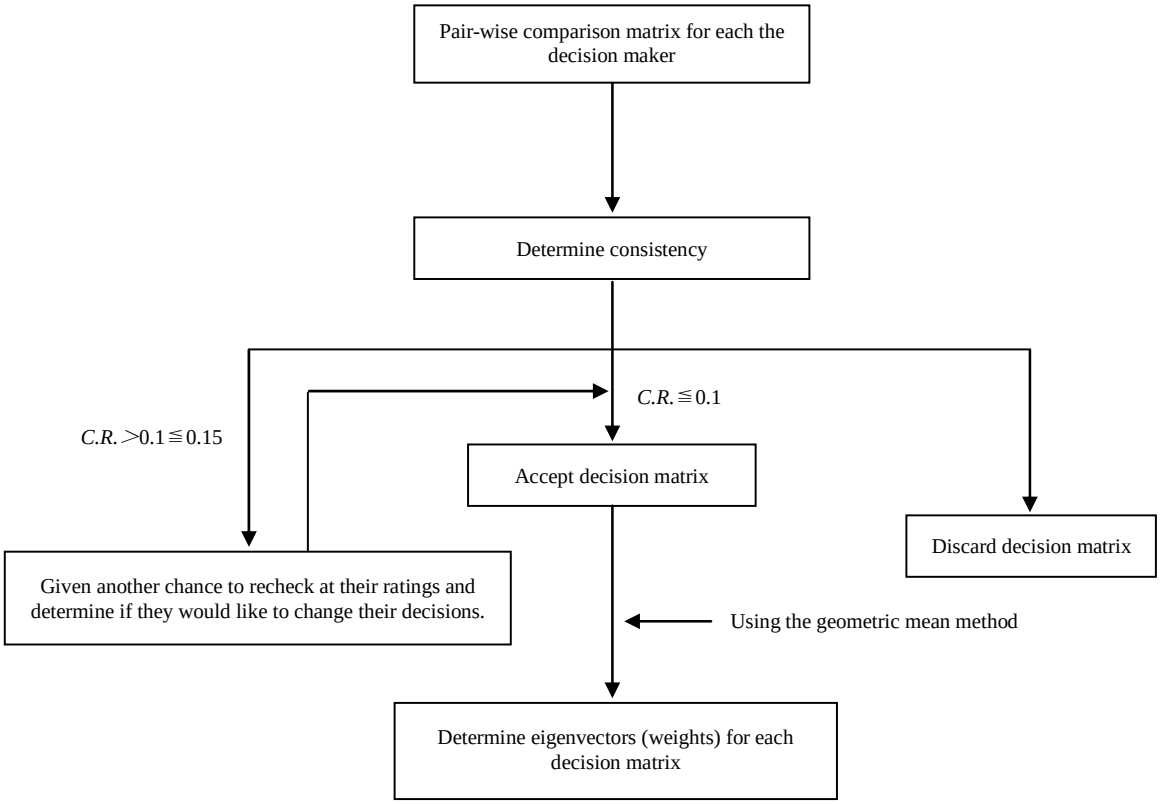


Figure 2. The procedure for determining consistency

Table 3. Aggregate pair-wise comparison matrix with eigenvectors

	C_1	C_2	C_3	C_4	C_5	C_6	Eigenvectors
C_1	1.000	0.812	0.474	0.321	0.722	1.182	0.108
C_2	1.231	1.000	0.762	0.695	0.433	2.150	0.147
C_3	2.110	1.313	1.000	0.913	1.167	1.909	0.207
C_4	3.120	1.438	1.095	1.000	1.278	2.110	0.241
C_5	1.385	2.310	0.857	0.782	1.000	1.636	0.198
C_6	0.846	0.465	0.524	0.474	0.611	1.000	0.098
$\lambda_{max} = 6.118$, *C.I.* = 0.024, *R I.* = 1.24, *C.R.* = 0.019							

Table 4. Summarizes eigenvectors (weights) results for levels 2 to 4

Criteria	Weights for level 2	Sub-Criteria	Weights for level 3	Weights of the overall	Weights for level 4			
					Al_1	Al_2	Al_3	Al_4
C_1	0.108	SC_1	0.207	0.041	0.290	0.300	0.152	0.258
		SC_2	0.157	0.035	0.316	0.281	0.140	0.263
		SC_3	0.258	0.052	0.278	0.260	0.180	0.282
		SC_4	0.378	0.060	0.296	0.309	0.126	0.269
C_2	0.147	SC_5	0.236	0.067	0.266	0.465	0.105	0.164
		SC_6	0.354	0.057	0.240	0.348	0.190	0.222
		SC_7	0.410	0.037	0.343	0.280	0.162	0.216
C_3	0.207	SC_8	0.323	0.046	0.320	0.347	0.112	0.221
		SC_9	0.275	0.114	0.344	0.221	0.112	0.323
		SC_{10}	0.179	0.127	0.222	0.207	0.236	0.335
C_4	0.241	SC_{11}	0.223	0.058	0.217	0.140	0.372	0.271
		SC_{12}	0.475	0.035	0.213	0.180	0.338	0.269
		SC_{13}	0.525	0.043	0.329	0.289	0.111	0.271
C_5	0.198	SC_{14}	0.295	0.062	0.287	0.265	0.114	0.334
		SC_{15}	0.177	0.017	0.289	0.330	0.126	0.254
		SC_{16}	0.216	0.033	0.246	0.358	0.121	0.276
		SC_{17}	0.312	0.024	0.330	0.306	0.100	0.264
C_6	0.098	SC_{18}	0.172	0.024	0.314	0.387	0.138	0.162
		SC_{19}	0.336	0.041	0.406	0.246	0.103	0.245
		SC_{20}	0.244	0.035	0.207	0.526	0.112	0.155
		SC_{21}	0.248	0.052	0.271	0.478	0.089	0.161

3.3. Step 3: Determine Eigenvectors

The relative scores provided by 11 experts were then aggregated by the geometric mean method. Table 3 presents the aggregate pair-wise comparison matrix and the consistency test for level 2. The eigenvectors (weighs) for level 2 can be determined by the procedure described in the previous section, and are as follows

$$\begin{matrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \end{matrix}\begin{bmatrix} 0.108 \\ 0.147 \\ 0.207 \\ 0.241 \\ 0.198 \\ 0.098 \end{bmatrix}. \tag{11}$$

The respective weights of the six evaluative criteria are functionality (0.305), reliability (0.255), usability (0.160), efficiency (0.092), maintainability (0.135) and portability (0.053).

The eigenvectors (weighs) for level 3 can be determined by the procedure described in the previous section, and are as follows

The twenty-one evaluative sub-criteria are weighted as follows: suitability (0.207), accuracy (0.157), interoperability (0.258), security (0.378), maturity (0.236), fault tolerance (0.354), recoverability 0.410), understandability (0.323), learn-ability (0.275), operability (0.179), attractiveness (0.223), time behavior (0.475), resource behavior (0.525), analyzability (0.295), changeability (0.177), stability (0.216), testability (0.312), adaptability (0.172), install-ability (0.336), co-existence (0.244) and replace-ability (0.248). Table 4 summarizes eigenvectors (weights) results for levels 2 to 4.

3.4. Step 4: Determine DVRs' Software Quality

According to Table 4, the quality of the four DVRs' software programs are then determined by Equation (12). Equation (12) indicates that the quality of the four DVRs' software programs are as follows: DVR A = 0.300, DVR B = 0.314, DVR C = 0.170 and DVR D= 0.277.

$$\begin{bmatrix} 0.290 & 0.316 & 0.278 & 0.296 & 0.266 & 0.240 & 0.343 & 0.320 & 0.344 & 0.222 & 0.217 & 0.213 & 0.329 & 0.287 & 0.289 & 0.246 & 0.330 & 0.314 & 0.406 & 0.207 & 0.271 \\ 0.300 & 0.281 & 0.260 & 0.309 & 0.465 & 0.348 & 0.280 & 0.347 & 0.221 & 0.207 & 0.140 & 0.180 & 0.289 & 0.265 & 0.330 & 0.358 & 0.306 & 0.387 & 0.246 & 0.526 & 0.478 \\ 0.152 & 0.140 & 0.180 & 0.126 & 0.105 & 0.190 & 0.162 & 0.112 & 0.112 & 0.236 & 0.372 & 0.338 & 0.111 & 0.114 & 0.126 & 0.121 & 0.100 & 0.138 & 0.103 & 0.112 & 0.089 \\ 0.258 & 0.263 & 0.282 & 0.269 & 0.164 & 0.222 & 0.216 & 0.221 & 0.323 & 0.335 & 0.271 & 0.269 & 0.271 & 0.334 & 0.254 & 0.276 & 0.264 & 0.162 & 0.245 & 0.155 & 0.161 \end{bmatrix}$$

$$\times \begin{bmatrix} 0.041 \\ 0.035 \\ 0.052 \\ 0.060 \\ 0.067 \\ 0.057 \\ 0.037 \\ 0.046 \\ 0.114 \\ 0.127 \\ 0.058 \\ 0.035 \\ 0.043 \\ 0.062 \\ 0.017 \\ 0.033 \\ 0.024 \\ 0.024 \\ 0.041 \\ 0.035 \\ 0.052 \end{bmatrix} = \begin{matrix} Al_1 \\ Al_2 \\ Al_3 \\ Al_4 \end{matrix} \begin{bmatrix} 0.300 \\ 0.314 \\ 0.170 \\ 0.277 \end{bmatrix} \tag{12}$$

The DVR B performed the best; end users must test the stability of the system. The software products were tested on an Intel Pentium 4 3.2GB, 1GB DDR400 RAM PC and sixteen visual channels running Windows XP Professional. Therefore, the mean CPU efficiency was 32%, and maximum efficiency was 43%. The mean MEM loading was 10586 K, and the top loading was 11200 K. During a week of testing, the system never crashed and never required automatic shutdown or restart. Clearly, DVR B had the best software quality.

4. Conclusions

This study proposes a multi-criteria evaluation model and algorithm capable of effectively evaluating software quality from the perspective of users or purchasers, thus enabling administrators or decision makers to identify optimum software quality. Significantly, this study provides procurement personnel with an easily applied and objective method of assessing the appropriateness of software quality. Therefore, this study proposes a method for identifying the best software quality from among those offered by four firms, by considering multiple assessment characteristics, improving upon the popular MCDM approach to alternative prioritization. This study presents an optimal operating model and algorithm for monitoring software in relation to users and purchasers, thus enabling administrators or decision makers to identify the most appropriate software quality. Based on the measurement results in this study, software users and developers can not only more thoroughly understand the merits and limitations of software products, but also ultimately enhance its overall quality. Administrators or decision makers adopt the measurement results of this study to evaluate software quality.

REFERENCES

[1] H. Aras, S. Erdogmus, and E. Koc, "Multi-criteria selection for a wind observation station location using analytic

hierarchy process," Renewable Energy, Vol. 29, 2004, pp. 1383–1392.

[2] V. Belton and T. J. Stewart, "Multiple criteria decision analysis: An integrated approach," Kluwer Academic Publishers, Boston, 2002.

[3] G. P. Cesar, M. Tom, and H. S. Brian, "A Metamodel for assessable software development methodologies," Software Quality Journal, Vol. 13, No. 2, pp. 195–214, 2005.

[4] C. W. Chang, C. R. Wu, and H. L. Lin, "Evaluating the digital video recorder systems using analytic hierarchy and analytic network processes," Information Sciences, Vol. 177, No. 16, pp. 3383–3396, 2007.

[5] C. W. Chang, C. R. Wu, and H. L. Lin, (2007b), "Integrating fuzzy theory and hierarchy concepts to evaluate software quality," Software Quality Journal, Published online, Vol. 11, No. 27, 2007.

[6] C. W. Chang, C. R. Wu, and H. L. Lin, "Group decision-making in a multiple criteria environment—A case using the AHPGR model to assess digital video recorder systems," Journal of Testing and Evaluation, Vol. 36, No. 2, pp. 583–589, 2008.

[7] P. F. Hsu and B.-Y. Chen, "Developing and implementing a selection model for bedding chain retail store franchisee using Delphi and fuzzy AHP," Quality and Quantity, Vol. 41, No. 2, pp. 275–290, 2007.

[8] ISO/IEC9126-1, "Software engineering-product quality-Part1: Quality model," 2001.

[9] G. Issac, C. Rajendran, and R. N. Anantharaman, "An instrument for the measurement of customer perceptions of quality management in the software industry: An empirical study in India," Software Quality Journal, Vol. 14, No. 4, pp. 291–308, 2005.

[10] L. S. Jose and H. Ines, "An AHP-based methodology to rank critical success factors of executive information systems," Computer Standards and Interfaces, Vol. 28, pp. 1–12, 2005.

[11] R. Kazman, L. Bass, M. Klein, T. Lattanze, and L. Northrop, "A Basis for Analyzing Software Architecture Analysis Methods," Software Quality Journal, Vol. 13, No. 4, pp. 329–355, 2005.

[12] T. M. Khoshgoftaar, A. Herzberg, and N. Seliya, "Resource oriented selection of rule-based classification models: An empirical case study," Software Quality Journal, Vol. 14, No. 4, pp. 309–338, 2006.

[13] T. M. Khoshgoftaar, N. Seliya, and N. Sundaresh, "An empirical study of predicting software faults with case-based reasoning," Software Quality Journal, Vol. 14, No. 2, pp. 85–111, 2006.

[14] L. Z. Lin, and T. H. Hsu, "The qualitative and quantitative models for performance measurement systems: The agile service development," Quality & Quantity, Vol. 42, No. 4, pp. 445–476, 2008.

[15] F. Liu, K. Noguchi, A. Dhungana, A. V. V. N. S. N. Srirangam, and P. Inuganti, "A quantitative approach for setting technical targets based on impact analysis in software quality function deployment," Software Quality Journal, Vol. 14, No. 2, pp. 113–134, 2005.

[16] M. Mollaghasemi and J. Pet-Edwards, "Making multiple-objective decisions," Los Alamitos, IEEE Computer Society Press, CA, 1997.

[17] T. Rafla, P. N. Robillard, and M. C. Desmarais, (2007), "A method to elicit architecturally sensitive usability requirements: Its integration into a software development process," Software Quality Journal, Vol. 15, No. 2, pp. 117–133.

[18] T. L. Saaty, (1980), "The analytic hierarchy process," McGraw Hill, New York, NY.

[19] E. Tolgaa, M. L. Demircana, and C. Kahraman, "Operating system selection using fuzzy replacement analysis and analytic hierarchy process," International Journal of Production Economics, Vol. 97, pp. 89–117, 2005.

[20] C. R. Wu, C. W. Chang, and H. L. Lin, "FAHP sensitivity analysis for measurement nonprofit organizational performance," Quality & Quantity, Vol. 42, No. 3, pp. 283–302, 2008.

Intelligent Information Management, **2009, 1, 89-97**
doi:10.4236/iim.2009.12014 Published Online November 2009 (http://www.scirp.org/journal/iim)

Scientific Research

Optimization of Fused Deposition Modelling (FDM) Process Parameters Using Bacterial Foraging Technique

Samir Kumar PANDA[1], Saumyakant PADHEE[2], Anoop Kumar SOOD[3], S. S. MAHAPATRA[4]
[1]*Department of Mechanical Engineering, National Institute of Technology, Rourkela, India*
[2]*Department of Manufacturing Science and Technology,*
Veer Surendra Sai University of Technology, Sambalpur, India
[3]*Department of Manufacturing Science, National Institute of Foundry and Forge Technology, Ranchi, India*
[4]*Department of Mechanical Engineering, National Institute of Technology, Rourkela, India*
Email: {*Samirpanda.nitrkl, soumyakantpadhee*2011,*anoopkumarsood*}*@gmail.com,*
*mahapatrass*2003*@yahoo.com*

Abstract: Fused deposition modelling (FDM) is a fast growing rapid prototyping (RP) technology due to its ability to build functional parts having complex geometrical shapes in reasonable build time. The dimensional accuracy, surface roughness, mechanical strength and above all functionality of built parts are dependent on many process variables and their settings. In this study, five important process parameters such as layer thickness, orientation, raster angle, raster width and air gap have been considered to study their effects on three responses viz., tensile, flexural and impact strength of test specimen. Experiments have been conducted using central composite design (CCD) and empirical models relating each response and process parameters have been developed. The models are validated using analysis of variance (ANOVA). Finally, bacterial foraging technique is used to suggest theoretical combination of parameter settings to achieve good strength simultaneously for all responses.

Keywords: fused deposition modelling (FDM), strength, distortion, bacterial foraging, ANOVA, central composite design (CCD)

1. Introduction

Reduction of product development cycle time is a major concern in industries to remain competitive in the marketplace and hence, focus has shifted from traditional product development methodology to rapid fabrication techniques like rapid prototyping (RP) [1–5]. Although RP is an efficient technology, full scale application has not gained much attention because of compatibility of presently available materials with RP technologies [6,7]. To overcome this limitation, one approach may be development of new materials having superior characteristics than conventional materials and its compatibility with technology. Another convenient approach may be suitably adjusting the process parameters during fabrication stage so that properties may improve [8,9]. A critical review of literature suggests that properties of RP parts are function of various process related parameters and can be significantly improved with proper adjustment. Since mechanical properties are important for functional parts, it is absolutely essential to study influence of various process parameters on mechanical properties so that improvement can be made through selection of best settings. The present study focus on assessment of mechanical properties viz. tensile, flexural and impact strength of part fabricated using fused deposition modelling (FDM) technology. Since the relation between a particular mechanical property and process parameters related to it is difficult to establish, attempt has been made to derive the empirical model between the processing parameters and mechanical properties using response surface methodology (RSM). In addition, effect of each process parameter on mechanical property is analysed. Residual analysis has been carried out to establish validity of the model. Development of valid models helps to search the landscape to find out best possible parametric combination resulting in theoretical maximum strength which has not been explored during experimentation. In order to follow search procedure in an efficient manner, latest evolutionary technique such as bacteria foraging has been adopted due to its superior performance over other similar random search techniques [10]. First, bacteria foraging is easier to implement because only few parameters need to be ad-

justed. Every bacterium remembers its own previous best value as well as the neighbourhood best and hence, it has a more effective memory capability than other techniques. Bacteria foraging is more efficient in maintaining the diversity of the swarm as all the particles use the information related to the most successful particle in order to improve themselves. In contrast, the worse solutions are discarded and only the good ones are saved in genetic algorithm (GA) [11]. Therefore, the population revolves around a subset of the best individuals in GA. Particle swarm optimization (PSO) can be considered as a good option because of its fast convergence but it has the tendency to get trapped at local optimum unless some procedure is adopted to escape [12].

2. Literature Review

Fused deposition modelling (FDM) is one of the RP processes that build part of any geometry by sequential deposition of material on a layer by layer basis. The process uses heated thermoplastic filaments which are extruded from the tip of nozzle in a prescribed manner in a semi molten state and solidify at chamber temperature. The properties of built parts depend on settings of various process parameters fixed at the time of fabrication. Recently, research interest has been devoted to study the effect of various process parameters on responses expressed in terms of properties of built parts. Studies have concluded through design of experiment (DOE) approach that process parameters like layer thickness, raster angle and air gap significantly influence the responses of FDM ABS prototype [13,14]. Lee *et al*. [15] performed experiments on cylindrical parts made from three RP processes such as FDM, 3D printer and nano-composite deposition (NCDS) to study the effect of build direction on the compressive strength. Out of three RP technologies, parts built by NCDS are severely affected by the build direction. Wang *et al*. [16] have recommended that material used for part fabrication must have lower glass transition temperature and linear shrinkage rate because the extruded material is cooled from glass transition temperature to chamber temperature resulting in development of inner stresses responsible for appearance of inter- and intra-layer deformation in the form of cracking, de-lamination or even part fabrication failure. Bellehumeur *et al*. [17] have experimentally demonstrated that bond quality between adjacent filaments depends on envelope temperature and variations in the convective conditions within the building part while testing flexural strength specimen. Temperature profiles reveal that temperature at bottom layers rises above the glass transition temperature and rapidly decreases in the direction of movement of extrusion head. Microphotographs indicate that diffusion phenomenon is more prominent for adjacent filaments in bottom layers as compared to upper layers. Simulation of FDM process using finite element analysis (FEA) shows that distortion of parts is mainly caused due to accumulation of residual stresses at the bottom surface of the part during fabrication [18]. The literature reveals that properties are sensitive to the processing parameters because parameters affect meso-structure and fibre-to-fibre bond strength. Also uneven heating and cooling cycles due to inherent nature of FDM build methodology results in stress accumulation in the built part resulting in distortion which is primarily responsible for week bonding and thus affect the strength. It is also noticed that good number of works in FDM strength modelling is devoted to study the effect of processing conditions on the part strength but no significant effort is made to develop the strength model in terms of FDM process parameters for prediction purpose. The present study uses the second order response surface model to derive the required relationship among respective process parameters and tensile, flexural and impact strength. The predictive equations once validated can be used to envisage theoretical possible best parameter settings to attain maximum in response characteristic. Hence, the problem becomes constrained optimization problem. In this study, bacteria foraging approach has been adopted to deal with the optimization problem.

The use of evolutionary algorithms to solve complex optimization problems is very common these days because they provide very competitive results when solving engineering design problems [19,20]. Furthermore, swarm intelligence approaches have been also used to solve this kind of problems [21,22]. However, most of the work is centered on some algorithms such as Particle Swarm Optimization [23], Ant Colony Optimization [24] and Artificial Bee Colony [25]. Recently, another swarm-intelligence-based model known as Bacterial Foraging Optimization Algorithm (BFOA), inspired in the behavior of bacteria E. Coli in its search for food, has been proposed. Three behaviors were modeled by Passino in his original proposal [26]: 1) Chemotaxis, 2) reproduction and 3) elimination-dispersal. BFOA has been successfully applied to solve different type of problems like forecasting [27], transmission loss reduction [28] and identification of nonlinear dynamic systems [29].

3. Experimental Procedure

The tensile test and three-point bending tests were performed using Instron 1195 series IX automated material testing system with crosshead speeds of 1mm/s and 2mm/s respectively in accordance with ISO R527:1966 and ISO R178:1975 respectively. Charpy impact test performed in Instron Wolpert pendulum impact test machine is used to determine the impact strength of specimen in accordance with ISO 179:1982. During impact testing, specimen is subjected to quick and intense blow by hammer pendulum striking the specimen with a speed of 3.8m/s. The impact energy absorbed is measure of the

toughness of material and it is calculated by taking the difference in potential energy of initial and final position of hammer. Impact energy is converted into impact strength using the procedure mentioned in the standard.

Three specimens per experimental run are fabricated using FDM Vantage SE machine for respective strength measurement. The 3D models of specimen are modelled in CATIA V5 and exported as STL file. STL file is imported to FDM software (Insight). Here, factors as shown Table 1 are set as per experiment plan (Table 2). All tests are carried out at the temperature 23±2ºC and relative humidity 50±5% as per ISO R291:1977. Mean of each experiment trial is taken as represented value of respective strength and shown in Table 2. The material used for test specimen fabrication is acrylonitrile butadiene styrene (ABS P400).

4. Experimental Plan

It is evident from the literature that strength of FDM processed component primarily depend process parameters. Therefore, five important control factors such as layer thickness (A), part build orientation (B), raster angle (C), raster width (D) and raster to raster gap (air gap) (E) are considered in this study. Other factors are kept at their fixed level.

In order to build empirical model for each tensile strength, flexural strength and impact strength, experiments were conducted based on central composite design (CCD) [30]. The CCD is capable of fitting second order polynomial and is preferable if curvature is assumed to be present in the system. To reduce the experiment run, half factorial 2^K design (K factors each at two levels) is considered. Maximum and minimum value of each factor is coded into +1 and -1 respectively using Equation 1 so that all input factors are represented in same range.

$$\xi_{ij} = \left(\frac{x_{ij} - \overline{x_i}}{\Delta x_i}\right) \times 2 \qquad (1)$$

$$\overline{x_i} = \frac{\sum_{j=1}^{2} x_{ij}}{2} \quad \text{and} \quad \Delta x_i = x_{i2} - x_{i1}$$

$$1 \le i \le K; \quad 1 \le j \le 2$$

where ξ_{ij} and x_{ij} are coded and actual value of j^{th} level of i^{th} factor respectively.

Apart from high and low levels of each factor, zero level (center point) and ±α level (axial points) of each factor is also included. To reduce the number of levels due to machine constraints, face centred central composite design (FCCCD) in which α=1 is considered. This design locates the axial points on the centres of the faces of cube and requires only three levels for each factor. Moreover, it does not require as many centre points as spherical CCD. In practice, two or three centre points are sufficient. In order to get a reasonable estimate of experimental error, six centre points are chosen in the present work. Table 1 shows the factors and their levels in terms of uncoded units as per FCCCD. For change in layer thickness, change of nozzle is needed. Due to unavailability of nozzle corresponding to layer thickness value at centre point as indicated by Equation 1, modified centre point value for layer thickness is taken. Half factorial 2^5 unblocked design having 16 experimental run, 10 (2K, where K=5) axial run and 6 centre run is shown in Table 2 together with the response value for tensile, flexural and impact strength for each experimental run.

5. Bacteria Foraging Optimization Algorithm (BFOA)

As other swarm intelligence algorithms, BFOA is based on social and cooperative behaviors found in nature. In fact, the way bacteria look for regions of high levels of nutrients can be seen as an optimization process. Each bacterium tries to maximize its obtained energy per each unit of time expended on the foraging process and avoiding noxious substances. Besides, swarm search assumes communication among individuals. The swarm of bacteria, S, behaves as follows:

1) Bacteria are randomly distributed in the map of nutrients.
2) Bacteria move towards high-nutrient regions in the map. Those located in regions with noxious substances or low-nutrient regions will die and disperse, respectively. Bacteria in convenient regions will reproduce (split).

Table 1. Factors and their levels (*modified)

Factor	Symbol	Unit	Low Level (-1)	Centre point (0)	High Level (+1)
Layer thickness	A	mm	0.1270	0.1780*	0.2540
Orientation	B	degree	0.0000	15.000	30.000
Raster angle	C	degree	0.0000	30.000	60.000
Raster width	D	mm	0.4064	0.4564	0.5064
Air gap	E	mm	0.0000	0.0040	0.0080

Table 2. Experimental data obtained from the FCCCD runs

RunOrder	Factor (Coded units)					Tensile Strength (MPa)	Flexural Strength (MPa)	Impact Strength (MJ/m^2)
	A	B	C	D	E			
1	-1	-1	-1	-1	+1	15.6659	34.2989	0.367013
2	+1	-1	-1	-1	-1	16.1392	35.3593	0.429862
3	-1	+1	-1	-1	-1	9.1229	18.8296	0.363542
4	+1	+1	-1	-1	+1	13.2081	24.5193	0.426042
5	-1	-1	+1	-1	-1	16.7010	36.5796	0.375695
6	+1	-1	+1	-1	+1	17.9122	38.0993	0.462153
7	-1	+1	+1	-1	+1	18.0913	39.2423	0.395833
8	+1	+1	+1	-1	-1	14.0295	22.2167	0.466667
9	-1	-1	-1	+1	-1	14.4981	27.6040	0.342708
10	+1	-1	-1	+1	+1	14.8892	34.5569	0.429167
11	-1	+1	-1	+1	+1	11.0262	20.0259	0.379167
12	+1	+1	-1	+1	-1	14.7661	25.2563	0.450001
13	-1	-1	+1	+1	+1	15.4510	36.2904	0.375000
14	+1	-1	+1	+1	-1	15.9244	37.3507	0.437785
15	-1	+1	+1	+1	-1	11.8476	22.9759	0.419792
16	+1	+1	+1	+1	+1	15.9328	28.8362	0.482292
17	-1	0	0	0	0	13.4096	27.7241	0.397222
18	+1	0	0	0	0	15.8933	33.0710	0.44757
19	0	-1	0	0	0	14.4153	34.7748	0.402082
20	0	+1	0	0	0	9.9505	25.2774	0.388539
21	0	0	-1	0	0	13.7283	27.5715	0.382986
22	0	0	+1	0	0	14.7224	30.0818	0.401388
23	0	0	0	-1	0	13.5607	28.9856	0.401041
24	0	0	0	+1	0	13.8388	28.8622	0.395833
25	0	0	0	0	-1	13.6996	28.8063	0.405555
26	0	0	0	0	+1	13.8807	29.0359	0.409028
27	0	0	0	0	0	14.4088	29.7678	0.407292
28	0	0	0	0	0	13.0630	31.6717	0.396373
29	0	0	0	0	0	13.8460	30.1584	0.406558
30	0	0	0	0	0	13.8727	31.0388	0.397712
31	0	0	0	0	0	13.5914	29.1475	0.401156
32	0	0	0	0	0	13.2189	31.9426	0.410686

3) Bacteria are located in promising regions of the map of nutrients as they try to attract other bacteria by generating chemical attractants.

4) Bacteria are now located in the highest-nutrient region.

5) Bacteria now disperse as to look for new nutrient regions in the map.

Three main steps comprise bacterial foraging behavior: 1) Chemotaxis (tumble and swimming), 2) reproduction and 3) elimination-dispersal. Based on these steps, Passino proposed the Bacterial Foraging Optimization Algorithm which is summarized in pseudo-code as follows:

Pseudo-Code for BFOA

Begin

Initialize input parameters: number of bacteria (S_b), chemotactic loop limit (N_c), swim loop limit (N_s), reproduction loop limit (N_{re}), number of bacteria for reproduction (S_r), elimination-dispersal loop limit (N_{ed}),

step sizes (C_i) depending on dimensionality of the problem and probability of elimination dispersal (P_{ed}).

Create a random initial swarm of bacteria $\theta^i(j, k, l)\ \forall_i$, $i = 1, \ldots, S_b$

Evaluate $f(\theta^i(j, k, l))\ \forall_i$, $i = 1, \ldots, S_b$

For l=1 to N_{ed} Do

For k=1 to N_{re} Do

For j=1 to N_c Do

For i=1 to S_b Do

Perform the chemotaxis step (tumble-swim or tumble-tumble) for bacteria θi(j, k, l)

End For

End For

Perform the reproduction step by eliminating the S_r (half) worst bacteria and duplicating the other half

End For

Perform the elimination-dispersal step for all bacteria $\theta^i(j, k, l)\ \forall_i$, $i = 1, \ldots, N_b$ with probability $0 \le P_{ed} \le 1$

End For

End

The chemotactic step was modeled by Passino with the generation of a random direction search (Equation 2)

$$\varphi(i) = \frac{\Delta(i)}{\sqrt{\Delta(i)^T \Delta(i)}} \tag{2}$$

where $\Delta\ (i)^n$ is a randomly generated vector with elements within the interval [−1, 1]. After that, each bacteria $\theta^i(j, k, l)$ modifies its positions as indicated in Equation 3.

$$\theta^i(j+1,\ k,\ l) = \theta^i(j,\ k,\ l) + C(i)\,\Phi\ (i) \tag{3}$$

Equation 2 represents a tumble (search direction) and Equation 3 represents a swim. The swim will be repeated N_s times if the new position is better than the previous one: $f(\theta^i(j + 1, k, l)) < f(\theta^i(j, k, l))$. The reproduction step consists on sorting bacteria in the population $\theta^i(j, k, l)\ \forall_i$, $i = 1, \ldots, N_b$ based on their objective function value $f(\theta^i(j, k, l))$ and to eliminate half of them with the worst value. The remaining half will be duplicated as to maintain a fixed population size. The elimination-dispersal step consists on eliminate each bacteria $\theta^i(j, k, l)\ \forall_i$, $i = 1, \ldots, N_b$ with a probability $0 \le P_{ed} \le 1$. It should be noted that BFOA requires 7 parameters and the n step sizes depending of the number of variables of the problem to be fine-tuned by the user.

5. Results and Discussions

Analysis of the experimental data obtained from FCCCD design runs is carried out using MINITAB R 14 software for full quadratic response surface model as given by Equation 4.

$$y = \beta_0 + \sum_{i=1}^{k} \beta_i x_i + \sum_{i=1}^{k} \beta_{ii} x_i x_i + \sum_{i<j} \sum \beta_{ij} x_i x_j \tag{4}$$

where y is the response, x_i is i^{th} factor

For significance check, F value given in ANOVA table is used. Probability of F value greater than calculated F value due to noise is indicated by p value. If p value is less than 0.05, significance of corresponding term is established. For lack of fit, p value must be greater the 0.05. An insignificant lack of fit is desirable because it indicates any term left out of model is not significant and developed model fits well. Based on analysis of variance (ANOVA), full quadratic model is found to be suitable for tensile strength, flexural strength and impact strength (Table 3) with regression p-value less than 0.05 and lack of fit more then 0.05. For tensile strength, all the terms are significant where as square terms are insignificant for

Table 3. Analysis of variance (ANOVA) table

Source	DOF	Tensile strength				Flexural strength				Impact strength			
		SS	MS	F	p	SS	MS	F	p	SS	MS	F	p
Regression	20	112.482	5.6241	11.65	0.000	799.058	39.953	14.96	0.000	0.0293	0.00146	16.72	0.000
Linear	5	64.373	12.8750	26.66	0.000	611.818	122.36	45.81	0.000	0.0258	0.00515	58.88	0.000
Square	5	14.966	2.9932	6.20	0.006	4.470	0.894	0.330	0.882	0.0019	0.00038	4.30	0.021
Interaction	10	33.143	3.3143	6.86	0.002	182.771	18.277	6.840	0.002	0.0016	0.00016	1.85	0.164
Residual	11	5.312	0.4829			29.383	2.671			0.0010	8.8E-05		
Lack of fit	6	4.116	0.6861	2.870	0.134	23.245	3.874	3.160	0.114	0.0008	0.00013	4.03	0.074
Pure error	5	1.196	0.2392			6.138	1.228			0.0002	3.3E-05		
Total	31	117.794				828.442				0.0302			

DOF= degree of freedom; SS= sum of square; MS= mean sum of square

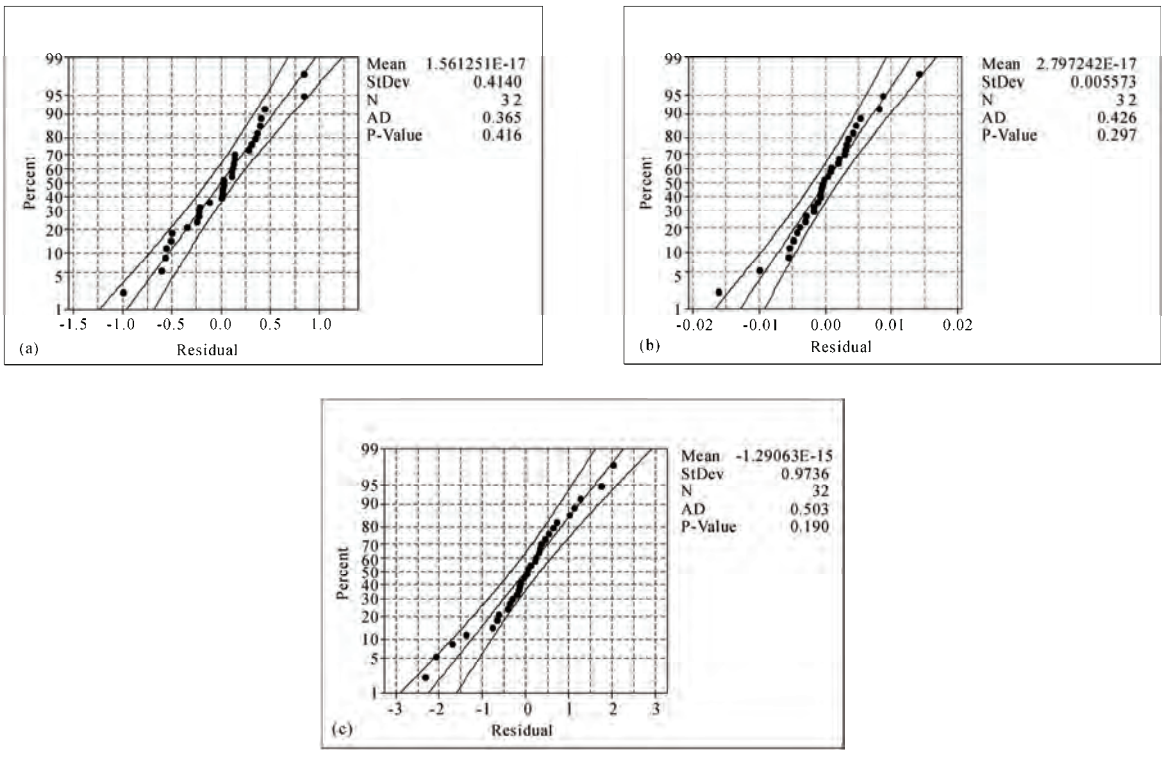


Figure 1. Normal probability plot of residual at 95% of confidence interval (a) Tensile strength (b) Impact strength (c) Flexural strength

flexural strength and interaction terms do not impart significant effect on impact strength.

The t-test was performed to determine the individual significant term at 95% of confidence level and final response surface equations in uncoded form for tensile strength (T_S), flexural strength (F_S) and impact strength (I_S) are given from Equation 5 to Equation 7 respectively in terms of coded units. The coefficient of determination (R^2) which indicates the percentage of total variation in the response explained by the terms in the model is 95.5%, 96.5% and 96.8% for tensile, flexural and impact strength respectively.

$$T_S = 13.5625 + 0.7156\,A - 1.3123\,B + 0.9760\,C + 0.5183\,E + 1.1671\,A^2 - 1.3014\,B^2 - 0.4363\,(A\times C) + 0.4364\,(A\times D) - 0.4364\,(A\times E) + 0.4364\,(B\times C) + 0.4898\,(B\times E) - 0.5389\,(C\times D) + 0.5389\,(C\times E) - 0.5389\,(D\times E) \quad (5)$$

$$F_S = 29.9178 + 0.8719\,A - 4.8741\,B + 2.4251\,C - 0.9096\,D + 1.6626\,E - 1.7199\,(A\times C) + 1.7412\,(A\times D) - 1.1275\,(A\times E) + 1.0621\,(B\times E) + 1.0621\,(C\times E) + 1.0408\,(D\times E) \quad (6)$$

$$I_S = 0.401992 + 0.034198\,A + 0.008356\,B + 0.013673\,C + 0.021383\,A^2 + 0.008077\,(B\times D) \quad (7)$$

Anderson-Darling (AD) normality test results are shown in Figure 1 for respective strength residue. P-value of the normality plot is more than 0.05 and signifies that residue follows the normal distribution.

Once the empirical models are validated for tensile, flexural, and impact strength of FDM built parts, next step is to search the optimization region for finding out suitable parameter settings that maximize responses beyond the experimental domain. Here, the objective function to be maximized is given as:

$$\text{Maximize} \quad (T_S \text{ or } F_S \text{ or } I_S). \quad (8)$$

Subjected to constraints:

$$A_{min} \le A \le A_{max} \quad (9)$$

$$B_{min} \le B \le B_{max} \quad (10)$$

$$C_{min} \le C \le C_{max} \quad (11)$$

$$D_{min} \le D \le D_{max} \quad (12)$$

$$E_{min} \le E \le E_{max} \quad (13)$$

The min and max in constraints 9-13 show the lowest and highest control factors settings (control factors) used in this study (Table 1). The constraints to be selected must be relevant to the response as shown in Equations 5-7. In order to apply BFOA, initial parameters of the are set as: number of bacteria, S_b=50, chemotactic loop limit, N_c=30, swim loop limit, N_s=100, reproduction loop limit, N_{re}=20, number of bacteria for reproduction, S_r=50%, elimination-dispersal loop limit, N_{ed}=50, step sizes, C_i

=1% of the difference of the maximum and minimum value of the variable, and probability of elimination dispersal, P_{ed} =0.05. The algorithm is coded in MATLAB and run on IBM Pentium IV desktop computer. The convergence curve for different responses is shown in Figure 2. The response values and optimal parameter settings are shown in Table 4. It can be observed from it that the response values significantly (in the tune of three to four times of experimental values) improve only setting the parameters within available range. Layer thickness (A) should be maintained at lower limit for improving tensile and flexural strength whereas it is to be set at higher range for the improvement of impact strength. Same trend of setting has also been observed for part orientation (B). Raster angle (C) and raster width (D) must be set at higher ranges for improving all responses. However, air gap (E) should be set at higher and medium range for improvement of tensile and flexural strength respectively whereas it does not play significant role for improvement of impact strength Lower layer thickness and air gap helps to occur diffusion in an effective manner causing better bonding among layers and better heat dissipation.

6. Conclusions

In this work, functional relationship between process parameters and strength (tensile, flexural and impact) forFDM process has been developed using response surface methodology for prediction purpose. The process pa rameters considered are layer thickness, orientation, raster angle, raster width and air gap. The empirical models

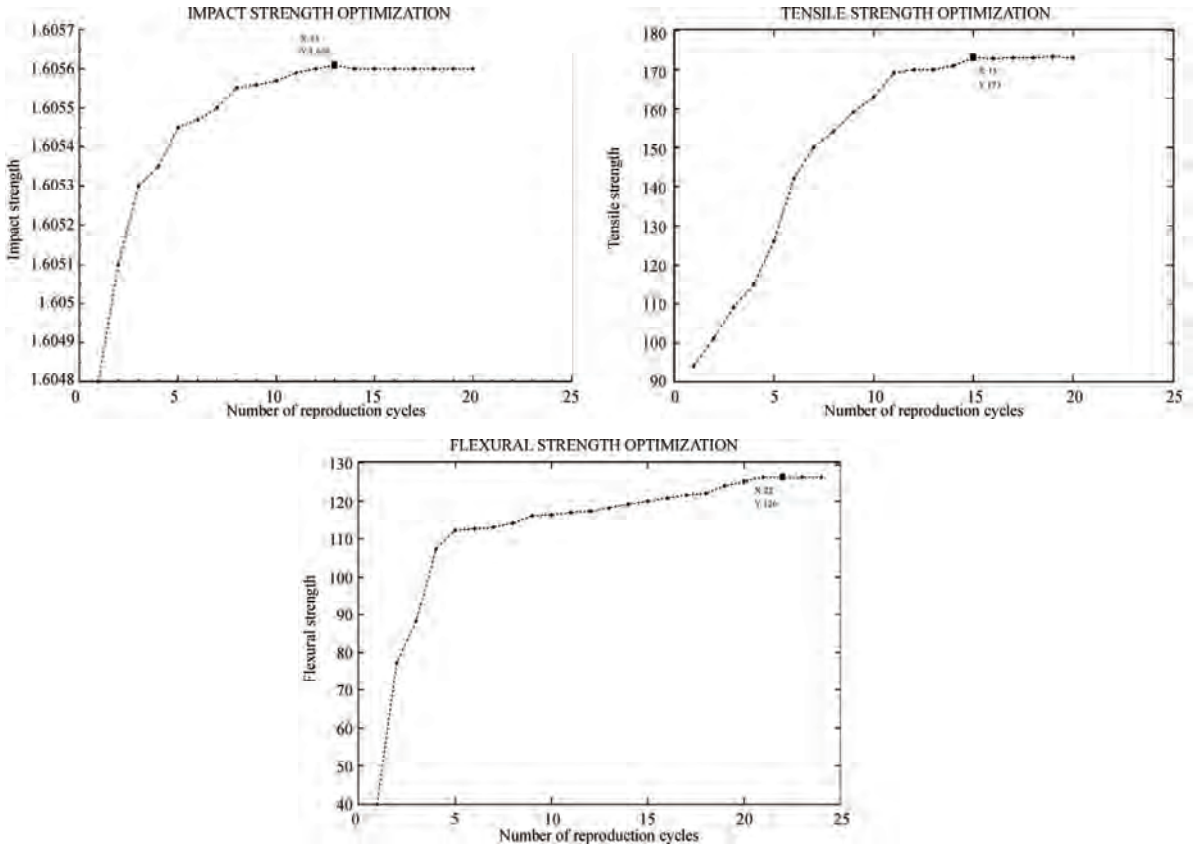


Figure 2. Convergence curves

Table 4. Optimal parameter settings with response values

Responses	**Value**	Layer thickness (A)	Orientation (B)	Raster angle (C)	Raster width (D)	Air gap (E)
Tensile strength	174.3177	0.1318	9.6100	59.9937	0.4196	0.0074
Flexural strength	126.4818	0.1278	4.9504	54.7311	0.4960	0.0034
Impact strength	1.6056	0.2531	29.9963	59.9951	0.5063	Not applicable

developed here have passed all the known statistical tests and can be used in practice. Since FDM process is a complex one, it is really difficult to obtain good functional relationship between responses and process parameters. A latest evolutionary approach such as bacterial foraging has been used to predict optimal parameter settings. It predicts parameters in universal range of values that may not exist in the experimental system set up. Therefore, this technique suggests alternative system settings so as to produce optimum results in the process. The parameter settings seem to be logical from the practical point of view. Number of layers in a part depends upon the layer thickness and part orientation. If number of layers is more, it will result in high temperature gradient towards the bottom of part. This will increase diffusion between adjacent rasters and strength will improve. Small raster angles are not preferable as they will results in long rasters which will increase the stress accumulation along the direction of deposition resulting in more distortion and hence weak bonding. Thick rasters results in high temperature near the boding surfaces which may improve the diffusion and may result in strong bond formation. The study can also extended in the direction of studying compressive strength, fatigue strength and vibration analysis.

REFERENCES

[1] B. Wiedemann and H. A. Jantzen, "Strategies and applications for rapid product and process development in Daimler-Benz AG. Computers in Industries," Vol. 39, No. 1, pp. 11–25, 1999.

[2] S. Upcraft and R. Fletcher, "The rapid prototyping technologies", Rapid Prototyping Journal, Vol. 23, No. 4, pp. 318–330, 2003.

[3] S. Mansour and R. Hauge, "Impact of rapid manufacturing on design for manufacturing for injection moulding," Journal of Engineering Manufacture, Part B, Vol. 217, No. 4, pp. 453–461, 2003.

[4] N. Hopkinson, R. J. M. Hagur, and P. H.Dickens, "Rapid manufacturing: An industrial revolution for the digital age," John Wiley and Sons Ltd., England, 2006.

[5] A. Bernarand and A. Fischer, "New trends in rapid product development," CIRP Annals-Manufacturing Technology, Vol. 51, No. 2, pp. 635–652, 2002.

[6] G. N. Levy, R. Schindel, J. P. Kruth, and K. U. Leuven, "Rapid manufacturing and rapid tooling with layer manufacturing (LM) technologies-State of the art and future perspectives," CIRP Annals-Manufacturing Technologies, Vol. 52, No. 2, pp. 589–609, 2003.

[7] A. Pilipović, P. Raos, M. Šercer, "Experimental analysis of properties of materials for rapid prototyping," International Journal of Advanced Manufacturing Technology, Vol. 40, No. 11–12, pp. 105–115, 2009.

[8] P. M. Pandey, P. K. Jain, and P. V. M.Rao, "Effect of delay time on part strength in selective laser Sintering," International Journal of Advanced Manufacturing Technology, Vol. 43, No. 1–2, pp. 117–126, 2009.

[9] K. Chockalingama, N. Jawahara, and U. Chandrasekhar, "Influence of layer thickness on mechanical properties in stereolithography," Rapid Protyping Journal, Vol. 12, No. 6, pp. 106–113, 2006.

[10] Y. Liu, K. M. Passino, and M. A. Simaan, "Biomimicry of social foraging bacteria for distributed optimization: Models, principles, and emergent behaviors," Journal of Optimization Theory and Applications, Vol. 115, No. 3, pp. 603–628, 2002.

[11] D. E. Goldberg, "Genetic algorithm in search, optimization and machine learning," Addison-Wesley Longman Publishing Co., Inc. Boston, MA, 1989.

[12] S. Biswas and S. S. Mahapatra, "An improved metaheuristic approach for solving the machine loading problem in flexible manufacturing systems," International Journal of Services and Operations Management, Vol. 5, No. 1, pp. 76–93, 2009.

[13] S. H. Ahn, M. Montero, D. Odell, S. Roundy, and P. K. Wright, "Anisotropic material properties of fused deposition modelling ABS," Rapid Prototyping Journal, Vol. 8, No. 4, pp. 248–257, 2002.

[14] Khan, Z.A., Lee, B.H., and J. Abdullah, "Optimization of rapid prototyping parameters for production of flexible ABS object," Journal of Material Processing Technology, Vol. 169, pp. 54–61, 2005.

[15] C. S. Lee, S. G. Kim, H. J. Kim, and S. H. Ahn, "Measurement of anisotropic compressive strength of rapid prototyping parts," Journal of Material Processing Technology, Vol. 187–188, pp. 637–630, 2007.

[16] T. M. Wang, J. T. Xi, and Y. Jin, "A model research for prototype warp deformation in the FDM process," International Journal of Advanced Manufacturing Technology, Vol. 33, No. 11–12, pp. 1087–1096, 2007.

[17] C. T. Bellehumeur, P. Gu, Q. Sun, and G. M. Rizvi, "Effect of processing conditions on the bonding quality of FDM polymer filaments," Rapid Prototyping Journal, Vol. 14, No. 2, pp. 72–80, 2008.

[18] K. Chou and Y. Zhang, "A parametric study of part distortion in fused deposition modeling using three dimensional element analysis," Journal of Engineering Manufacture, Part B, Vol. 222, pp. 959–967, 2008.

[19] E. Mezura-Montes and C. A. Coello Coello, "Constrained optimization via Multiobjective Evolutionary Algorithms," In J. Knowles, D. Corne, and K. Deb, editors, Multiobjective Problem Solving from Nature, Springer, Heidelberg, pp. 53–75, 2008.

[20] Z. Michalewicz and M. Schoenauer, "Evolutionary algorithms for constrained parameter optimization problems," Evolutionary Computation, Vol. 4, No. 1, pp. 1–32, 1996.

[21] S. He, E. Prempain, and Q. H. Wu, "An improved par-

ticle swarm optimizer for mechanical design optimization problems," Engineering Optimization, Vol. 36, No. 5, pp. 585–605, 2004.

[22] T. Ray and K. Liew, "Society and civilization: an optimization algorithm based on the simulation of social behavior," IEEE Transactions on Evolutionary Computation, Vol. 7, No. 4, pp. 386–396, 2003.

[23] J. Kennedy and R. C. Eberhart, "Swarm intelligence," Morgan Kaufmann, UK, 2001.

[24] G. Leguizamón and C. Coello-Coello, "A boundary search based ACO algorithm coupled with stochastic ranking," In 2007 IEEE Congress on Evolutionary Computation (CEC'07), Singapore, September 2007. IEEE Press, pp. 165–172, 2007.

[25] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," Applied Soft Computing, Vol. 8, No. 1, pp. 687–697, 2008.

[26] K. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," IEEE Control Systems Magazine, Vol. 22, No. 3, pp. 52–67, 2002.

[27] R. Majhi, G. Panda, G. Sahoo, P. Dash, and D. Das, "Stock market prediction of s&p 500 and DJIA using bacterial foraging optimization technique," In Proceedings of the IEEE Congress on Evolutionary Computation (CEC'07), Singapore, IEEE Service Center, pp. 2569-2575, September 2007.

[28] S. Mishra, G. D. Reddy, P. E. Rao, and K. Santosh, "Implementation of new evolutionary techniques for transmission loss reduction," In Proceedings of the IEEE Congress on Evolutionary Computation (CEC' 07), Singapore, IEEE Service Center, pp. 2331–2336, September 2007.

[29] B. Majhi and G. Panda, "Bacteria foraging based identificacion of nonlinear dynamic system," In Proceedings of the IEEE Congress on Evolutionary Computation (CEC'07), Singapore, IEEE Service Center, pp. 1636–1641, September 2007.

[30] D. C. Montgomery, "Design and Analysis of Experiments", Fifth Edition, John Wiley and Sons Pvt. Ltd., Singapore, 2003.

Intelligent Information Management, **2009, 1, 98-107**
doi:10.4236/iim.2009.12015 Published Online November 2009 (http://www.scirp.org/journal/iim)

Scientific Research

What can Software Engineers Learn from Manufacturing to Improve Software Process and Product?

Norman SCHNEIDEWIND
Information Sciences, Naval Postgraduate School, Monterey, US
Email: *ieeelife@yahoo.com*

Abstract: The purpose of this paper is to provide the software engineer with tools from the field of manufacturing as an aid to improving software process and product quality. Process involves classical manufacturing methods, such as statistical quality control applied to product testing, which is designed to monitor and correct the process when the process yields product quality that fails to meet specifications. Product quality is measured by metrics, such as failure count occurring on software during testing. When the process and product quality are out of control, we show what remedial action to take to bring both the process and product under control. NASA Space Shuttle failure data are used to illustrate the process methods.

Keywords: manufacturing methods, software process, software product, product and process quality, Taguchi methods

1. Introduction

Based on "old" ideas from the field of manufacturing, we propose "new" ideas for software practitioners to consider for controlling the quality of software. Although there are obviously significant difference between hardware and software, there are design quality control processes that can be adapted from hardware and applied advantageously to software. Among these are the Taguchi manufacturing methods, statistical quality control charts (statistical quality control is the use of statistical methods to control quality [1]), and design of experiments. These methods assess whether a product manufacturing process is within statistical control. Statistical control means that product attributes like software failure occurrence is within specified control limits.

Every process has inherent variation that hampers accurate process prediction. In addition, interactions of people, machines, environment, and methods introduce noise into the system. You can control the inherent variation by identifying and controlling its cause, thereby bringing the process under statistical control [2]. While this is important, we should not get carried away by focusing exclusively on process. Most customers do not care about process. They are interested in *product* quality! Thus, when evaluating process improvement, it is crucial to measure it in terms of the product quality that is achieved. Thus, we explore methods, like Taguchi methods, that tie process to product. We apply these methods to NASA Space Shuttle software using actual failure data. The use of the Shuttle as an example is appropriate because this is a CMMI Level 5 process that uses product quality measurements to improve the software development process [3]. It is a characteristic of processes at this level that the focus is on continually improving process performance through both incremental and innovative technological changes and improvements. At maturity level 5, processes are concerned with addressing statistical common causes of process variation and changing the process (for example, shifting the mean of the process performance) to improve process performance. This would be done at the same time as maintaining the likelihood of achieving the established quantitative process-improvement objectives. While some software engineers argue that a CMMI Level 5 process is not applicable to their software, we suggest that it is applicable to *any* software as an *objective*, with the benefit of evolving to a higher CMMI level.

Typical metrics that software engineers use for assessing software quality are defect, fault, and failure counts, complexity measures that measure the intricacies of the program code, and software reliability prediction models. The data to drive these methods are collected from defect reports, code inspections, and failure reports. The data are collected either by software engineers or automatically by using a variety of software measurement tools. These data are used in design and code inspections to remove defects and in testing to correct faults.

Since software engineers are typically not schooled in

manufacturing systems, they are unaware of how manufacturing methods can be applied to improving the quality of software. Our objective is to show software researchers and practitioners how manufacturing methods can be applied to improving the quality of software.

2. Relationship of Manufacturing to Software Development

There is a greater intersection between manufacturing methods and software development than software practitioners realize. For example, software researchers and industry visionaries have long dreamed about a Leg0 block style of software development: building systems by assembling prefabricated, ready-to-use parts. Over the last several decades, we have seen the focus of software development shift from the generation of individual programs to the assembly of large number of components into systems or families of systems. The two central themes that emerged from this shift are components, the basic system building blocks, and architectures, the descriptions of how components are assembled into systems. Interest in components and architectures has gone beyond the research community in recent years. Their importance has increasingly been recognized in industry. Today, a business system can be so complex that major components must be developed separately. Nevertheless, these individually developed components must work together when the system is integrated. The need for a component-based strategy also arises from the increased flexibility that businesses demand of themselves and, by extension, of their software systems. To address the business needs, the software industry has been moving very rapidly in defining architecture standards and developing component technologies. An emerging trend in this movement has been the use of middleware, a layer of software that insulates application software from system software and other technical or proprietary aspects of underlying run-time environments. [4] Component-based software construction enhances quality by confining faults and failures to specific components so that the cause of the problem can be isolated and the faults removed.

Manufacturing is an important part of any software development process. In the simplest case, a single source code file must be compiled and linked with system libraries. More usually, a program will consist of multiple separately written components. A program of this kind can be manufactured by compiling each component in turn and then linking the resulting object modules. Software generators introduce more complexity into the manufacturing process. When using a generator, the programmer specifies the desired effect of a component and the generator produces the component implementation. Generators enable effective reuse of high-level designs and software architectures. They can embody state of-the-art implementation methods that can evolve without affecting what the programmer has to write. While the programmer can manually invoke compilers and linkers, it is much more desirable to automate the software manufacturing process. The main advantage of automation is that it ensures manufacturing steps are performed when required and that only necessary steps are performed. These benefits are particularly noticeable during maintenance of large programs, perhaps involving multiple developers. Automation also enables other parties to manufacture the software without any detailed knowledge of the process. [5] Component reuse is a boon to software reliability because once a component is debugged, it can be used repeatedly with assurance that it is reliable.

Another tool borrowed from manufacturing is configuration control [6]. At first blush, one may wonder how configuration control fits with software development. Actually, it is extremely important because software is subject to change throughout the life cycle in requirements, design, coding, testing, operation and maintenance. A dramatic illustration of the validity of this assertion is the software maintenance function. For example, no sooner is the software delivered to the customer than requests for changes are received by the developer. This is in addition to the changes caused by bug fixes after delivery. If configuration management is not employed, the process will become chaotic with neither customer nor developer knowing the state of the software and the status of the process.

3. Arguments against Applying Manufacturing Methods to Software

We note that not all software engineers subscribe to the idea that manufacturing methods can be applied to software. They claim that it is infeasible to measure software because unlike high volume manufacturing of hardware, with close tolerances, many software projects are low volume produced with a fuzzy process, where product tolerance has no meaning. [7] Actually, none of this is true. While component-based software development is not a high volume process, it produces products for use in a large number of applications (e.g., edit module in a word processor). As for process, we have cited the CMMI Level 5 process that can control the variation in product quality by feeding back deviations in product quality to the development process for the purpose of correction the process that led to undesired variance in product quality. While it is true that software cannot be manufactured to the precision of software, statistical quality control of defect count, confidence intervals of reliability predictions, numerical reliability specifications, and quality prediction accuracy measurement, are just a few of the quantitative measurements from the physical world that have been applied to software, for example, in

the Space Shuttle [3]. Thus, we are convinced that it is appropriate to apply manufacturing methods to software development in the sections to follow.

4. Taguchi Methods

Taguchi [1] revolutionized the manufacturing process in Japan through cost savings, for example, at Toyota. He understood that all manufacturing processes are affected by outside influences, like noise. Taguchi developed methods of identifying those noise sources that have the greatest effects on product variability. His ideas have been adopted by successful manufacturers around the globe because they created superior production processes at much lower costs.

Taguchi speeded up product and process design by separating controllable from uncontrollable variables. By concentrating on the controllable variables, fewer experiments are needed to arrive at the best product design. For example, in software, one of the key variables that is controllable is quality. On the other hand, user expectations are largely uncontrollable from the developer's perspective. However, controlling quality to achieve positive results, would contribute to meeting users' expectations.

Since a good manufacturing process will be faithful to a product design, robustness must be designed into a product before manufacturing begins. According to Taguchi, if a product is designed to avoid failure in the field, then factory defects will be simultaneously reduced. This is one aspect of Taguchi Methods that is often misunderstood. There is no attempt to reduce variation, which is assumed to be inevitable, but there is a definite focus on reducing the effect of variation. "Noise" in processes will exist, but the effect can be minimized by designing a strong "signal" into a product [8].

To relate software to Taguchi methods, we use the following definitions:

4.1. Definitions

Observation i: the recording of cumulative failures during test or operational time t.

Y_i: value of observation i: i^{th} value of actual or predicted cumulative software failures

T_i: specified target value of cumulative software failures for observation i (i.e., desire $Y_i \leq T_i$)

Values of T_i are listed in Table 1. In order to not bias the analysis, the individual failure counts d_i were assigned a uniformly distributed number between 0 and 4. Then these numbers were summed obtain the cumulative failure target values T. Software engineers could choose values appropriate for their software.

Table 1. Shuttle OI3 Loss Functions and Signal to Noise Ratios

								Forecasted	Forecasted
		Actual	Actual	Random	Cumulative	Actual	Predicted	Actual	Predicted
	Failure	Failure	Cumulative	Failure Count	Failure	Loss	Loss	Loss	Loss
Test	Time	Count	Failures	Target	Target	Function	Function	Function	Function
I	t	d_i	Y_{ia}	T_i:	T_i	LF_a	LF_p	LFia	LFip
1	0.97	2	2	2	2	0	8.55	3.54	-9.75
2	1.20	0	2	0	2	0	4.62	3.73	-8.51
3	3.03	2	4	0	2	4	3.62	5.18	1.17
4	3.07	2	6	2	4	4	0.00	5.21	1.34
5	4.00	0	6	1	5	1	0.01	5.95	6.21
6	4.23	0	6	3	8	4	7.31	6.14	7.42
7	8.20	0	6	1	9	9	3.86	9.29	27.69
8	9.63	1	7	0	9	4	3.16	10.44	34.85
9	9.70	0	7	2	11	16	14.23	10.49	35.18
10	9.77	4	11	1	12	1	22.71	10.54	35.51
11	10.37	0	11	3	15	16	59.57	11.02	38.48
12	12.13	1	12	0	15	9	58.23	12.42	47.15
13	27.67	1	13	4	19	36	133.41	24.78	117.83
14	35.93	1	14	2	21	49	183.61	31.36	151.43
15	87.53	1	15	3	24	81	273.91	72.41	298.15
16	136.53	1	16	2	26	100	344.11	111.40	336.92
	140.00							114.16	335.96
	145.00							118.13	333.70
	150.00							122.11	330.42
	155.00							126.09	326.12
	160.00							130.07	320.80
	days					SN_T	SN_T	SN_T	SN_T
						1.5773	1.9060	1.8595	2.1514

LF: loss function (variability of difference between actual cumulative failures and target values during testing and operation)

SN_T: signal to noise ratio (proportional to the mean of the loss function)

n: number of observations (e.g., number of observations of Y_i and T_i)

Taguchi devised an equation -- called the loss function (LF) -- to quantify the decline of a customer's *perceived value* of a product as its *quality declines* [9,10] in Equation (1). This equation is a squared error function that is used in the analysis of errors resulting from deviations from desired quality. LF tells managers how much product quality they are losing because of variability in their production process. He also computed his version of the signal to noise ratio that essentially produces the mean of the loss function [9,10] in Equation (2). Note that in most applications, a high signal to noise ratio is desirable. However, in the Taguchi method this is not the case because Equation (2) only measures signal, and no noise. Thus, since Equation (2) represents the mean value of the difference between observed and target values, a high value of SN_T is *undesirable*.

$$LF = (Y_i - T_i)^2, \quad (1)$$

$$SN_T = 10\log_{10}\left(\frac{\sum_{i=1}^{n}(Y_i - T_i)^2}{n}\right) \quad (2)$$

The development of the loss function and signal to noise ratio, using the Shuttle failure data, involves the following steps:

1) First, we note that cumulative failure data are appropriate for comparing actual failure of a software release against the predicted values [11].

2) Thus, for release OI3 of the Shuttle flight software, we used *actual* cumulative failure data Y_{ia} to compute the *actual* loss function LF_a in Equation (1).

3) Then we used this loss function to compute the *actual* signal to noise ratio SN_T in Equation (2).

4) Then we used the *predicted* cumulative failure data Y_{ip} to obtain the *predicted* loss function LF_p and used this loss function to compute the *predicted* signal to noise ratio. The predicted cumulative failure counts were obtained from another research project using the Schneidewind Software Reliability Model (SSRM) [12]. Note in Figure 1 that there is a good fit with the actual data with $R^2 = .9359$. Also note that the failure times are long because the Shuttle flight software is subjected to continuous testing over many years of operation.

5) Next, the *actual* failure data Y_{ia} were fitted, as a function of failure time *t*, with the regression Equation (3).

6) Then *predicted* failure data Y_{ip} were fitted, as a function of failure time *t*, in the regression Equation (4), which also has a good fit to the data in Figure 1 with $R^2 = .9701$.

7) Finally, signal to noise ratios were computed using Equation (2) for each of the four cases.

$$LF_{ia}=0.7956t+2.7771 \quad (3)$$

$$LF_{ip}=-0.0204t^2+5.3622t-14.912 \quad (4)$$

The four loss functions -- two actual and two predicted – are shown in Table 1 and Figure 1, along with the four corresponding signal to noise ratios. Figure 1 and Equations (3) and (4) can be used by a software engineer to forecast the loss function *beyond* the range of the actual and predicted loss functions.

4.2. Results of Applying Taguchi Methods to Shuttle Software

In Figure 1 and Table 1, we see that the loss functions

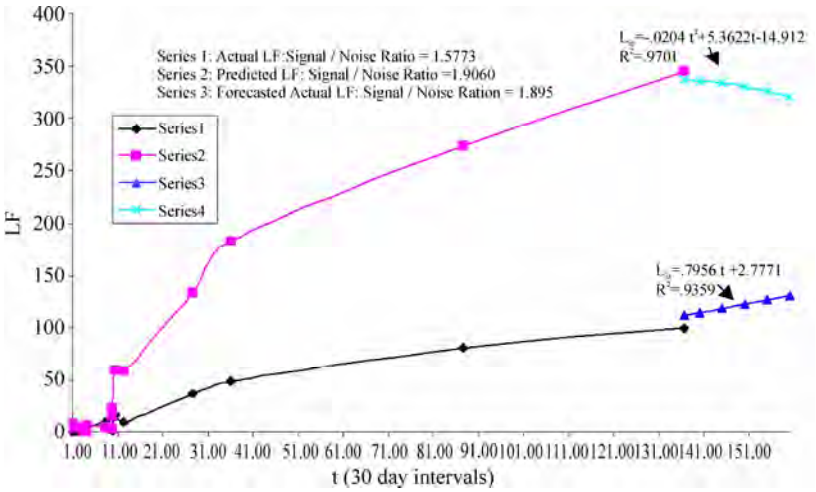


Figure 1. NASA space shuttle OI3: Loss function LF vs. failure time t

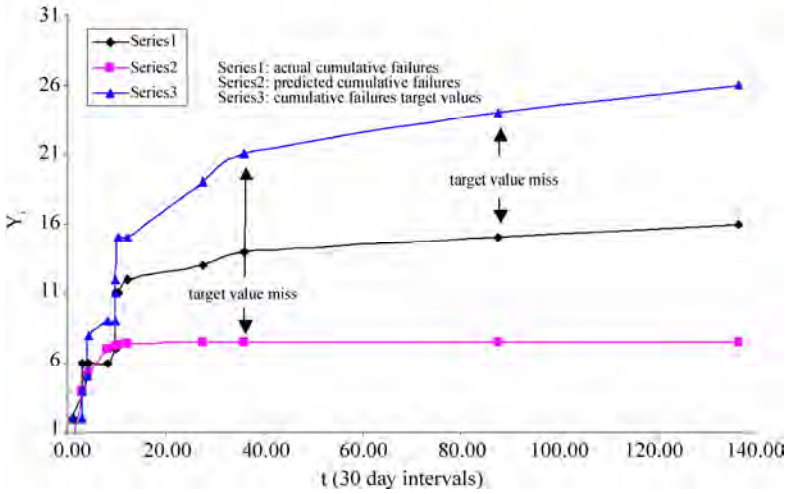


Figure 2. NASA shuttle OI3 cumulative failures Y_i vs. failure time t

show that there is excessive variation between the desired and target values, based on both actual and predicted cumulative failures. Secondly, the mean variabilities, or signal to noise ratios, are large. This result suggests that the software production process requires tightened quality control.

Figure 1 and Table 1 reveal additional information about variability in loss functions, as given by the signal to noise ratio S/N_T. We see that predicted values have greater variability than actual values and that forecasted values have the greatest variability of all LFs. The latter result is to be expected because forecasting into the unknown future is fraught with uncertainty. If the loss function could be reduced by exerting greater control over the software development process, the signal to noise ratio would also be reduced. By greater control we mean, for example, strengthening software inspection procedures and subjecting the software to stress tests to uncover and remove defects earlier in the development process.

Another perspective of applying Taguchi methods is shown in Figure 2, where we have plotted actual, predicted, and target cumulative failures against the time of failure for Shuttle release OI3. The message of this figure is that there is a large miss between the target values and the actual and predicted cumulative failures. Therefore, again, the remedy would be to strengthen the software development process by, for example, employing stringent inspection procedures to reduce the incidence of failures.

5. Statistical Quality Control

Now we illustrate another common manufacturing process control method that we show has applicability to software. Process control is based on two key assumptions, one of which is that random variability is basic to any production process. No matter how perfectly a process is designed, there will be some random variability, also called *common causes*, in quality characteristics from one unit to the next [13]. For example, a software development process cannot achieve perfection in specifying requirements, producing error-free code, inspecting and finding every defect, and finding and correcting every fault during testing. When undesirable variation in these activities becomes excessive, the process is out of control, and the search is on for *assignable causes* [13] For example, requirements analysts may be *systematically* misinterpreting customer requirements or software designers may make an incorrect mapping from requirements to code. In these situations, the process must be corrected.

Statistical quality control is based on the idea of monitoring quality and providing an alert when the process is out of control. Usually, variation is monitored and controlled by deviations from the mean, as computed by the standard deviation. Control is exercised by accepting the product if quality is within limits and rejected it, otherwise. For software, product metrics like the count of failures in excess of limits, as failures occur over a series of tests, can be used as a *surrogate* for identifying an out of control process.

As pointed out by [14], in hardware manufacturing, the number of observed failures is close to the actual number of failures that occur over time. In software, this is not the case. For example, a tester may observe two failures on a particular test, but the actual number of failures that would have occurred with a better test is ten. Thus, if the control limit were five, the tester would record this software as being under control on the control chart. To mitigate against this possibility, you should observe the trend of failure count, with increasing testing, to see whether the trend is increasing.

Statistical quality control for software uses the fol-

lowing definitions and quality control relationships:

5.1. Definitions

n: Sample size
d_i: failure count for test i
$\overline{d_i}$: Mean value of d_i
Sample i: i^{th} observation of d_i during test i
s: Sample standard deviation of d_i
d_i (min): minimum value of d_i
d_i: (max) maximum value of d_i

5.2. Quality Control Relationships

For the case where the standard deviation is used to compute limits, Equations (5) and (6) are used.

Upper Control Limit (UCL): $\overline{d_i} + Z\ s$ (5)

Lower Control Limit (LCL): $\overline{d_i} - Z\ s$ (6)

where Z is the number of standard deviations from the mean, which can be 1, 2, or 3, depending on the degree of control exercised (Z = 1: tight control, Z =3: loose control).

5.3. Results of Applying Statistical Quality Control Methods to Shuttle Software

Since the Shuttle is a safety critical system, we chose Z = 1 to provide tight control. Figure 3 reveals, as the Taguchi methods did, that there is an out of control situation when failure count become excessive -- in this case at test # 10. In combining the results from Taguchi methods and statistical quality control, corrective action, like increased inspection, is necessary.

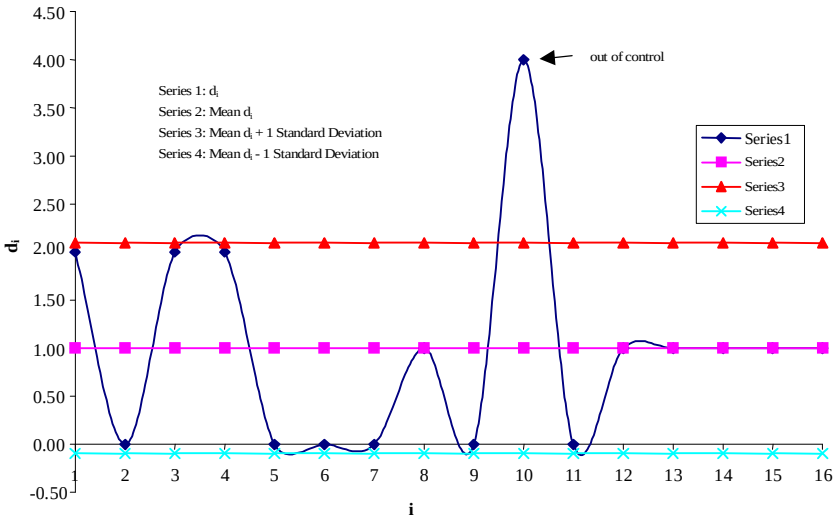


Figure 3. NASA space shuttle OI3: Failure count d_i vs. test i

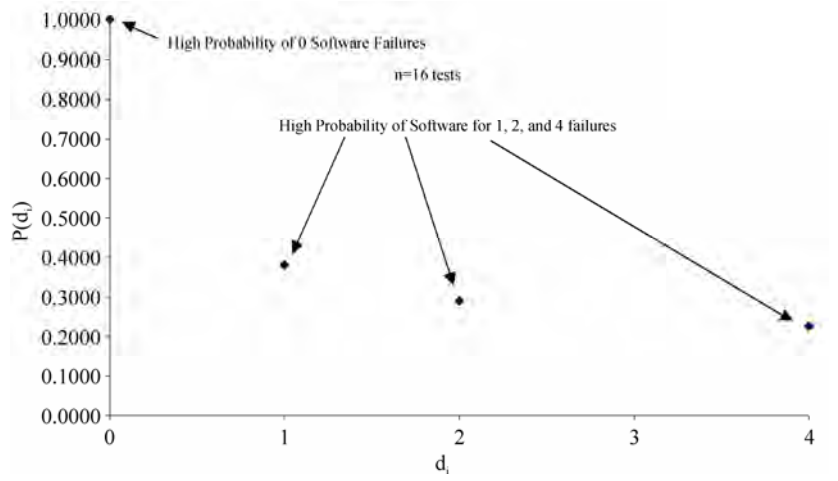


Figure 4. NASA space shuttle OI3: Probability of d_i failures P (d_i) on n tests vs. d_i

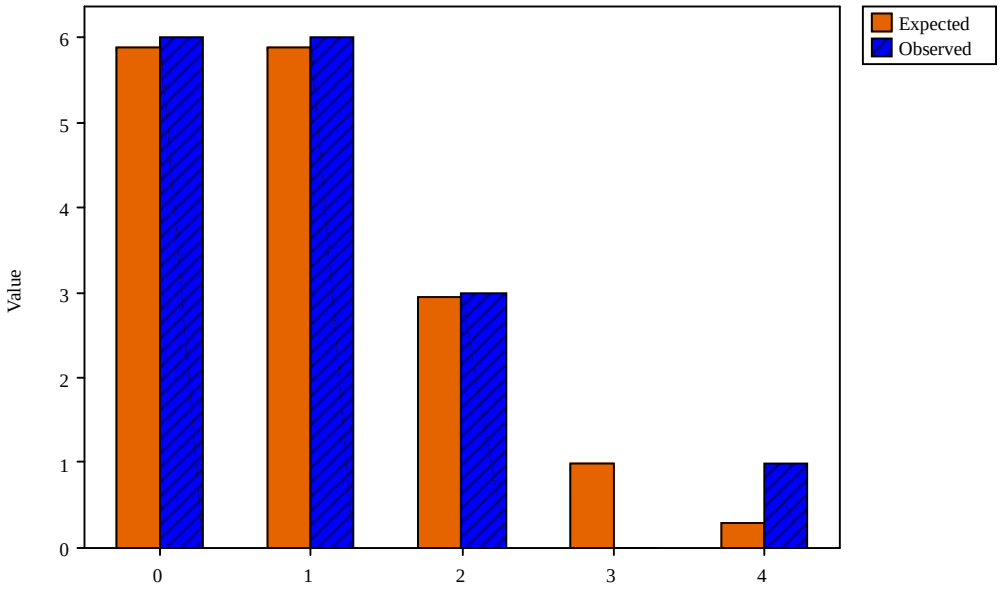


Figure 5. Test of poisson distribution of di number of failures during test i

6. Statistical Process Control Using Failure Probability and Failure Counts [15]

Next we show other methods of statistical process control that focuses on the probability of failures and number of failures occurring in a sample on a test.

6.1. Probability of Failures during Tests

We use probability of failure P (d_i) of d_i failures during *n* tests to control quality. This approach is based on the binomial distribution with probability p_i of failures on test *i* [16]. In our example, using the Shuttle data, the probability of failures P (d_i) is computed from the failure counts d_i and p_i in Equation (8). In order to compute P (d_i), we first compute probability p_i in Equation (7).

$$p_i = di / n \tag{7}$$

$$P(d_i) = \left[\frac{n!}{d_i!(n-d_i)!}\right][p_i^{d_i}][(1-p_i)^{n-d_i}] \tag{8}$$

Applying Equation (8) to Figure 4, we note that there would be considerable risk in deploying this software in view of the high probability of failure for 2, 3, and 4 failures, given that 16 tests were used. An obvious remedy for this problem would be to remove the faults causing the failures before the software is deployed.

6.2. Poisson Control Charts

The Poisson control chart is based on the assumption that failure counts are distributed during tests according to a Poisson distribution. As can be seen in Figure 5, the observed failure counts d_i on the x axis are a good match with the expected counts for a Poisson distribution. Therefore, the Poisson distribution of failure counts, given in Equation (9), is used. There is sometimes a concern that the Poison distribution requires the assumption of independence of failures. Actually, dependence is not a frequent occurrence. For example, Musa [17] found that in 15 projects there was little association among failures.

$$p_i = \frac{\bar{d}^{\,d_i} e^{-\bar{d}}}{d_i!} \tag{9}$$

Next, having estimated p_i in Equation (9), estimate the *weighted mean number* of failures across n samples, using Equation (10):

$$\bar{P} = \sum_{i=1}^{n} d_i p_i \tag{10}$$

Since the standard deviation s of the Poisson distribution is equal to square root of the mean, we have:

$$s = \sqrt{\bar{P}} \tag{11}$$

Now, compute the lower control limit for the *number* of failures using Equations (10) and (11) and producing Equation (12):

$$LCL = \bar{P} - 3\ s \tag{12}$$

It is necessary to compute the lower limit in order to identify quality that may be too *high*, resulting in waste of resources on software that does not require high quality.

Then, compute the upper control limit for the *number* of failures, again using Equations (10) and (11) and pro-

ducing Equation (13):

$$UCL = \overline{P} + 3\ s \qquad (13)$$

Figure 6 confirms the result obtained in Figure 3 in that there is loss of control at test # 10 where four failures occur. The faults causing these failures must be uncovered and removed.

7. Design of Experiments [18]

The last method in manufacturing process control we consider is the design of experiments that employs a hypothesis about the difference between desired (target) and actual values of a product attribute, such as cumulative failures, using software tests as the experiments. The idea is to use samples (software tests) and statistical tests to estimate whether there is a significant difference between desired and actual values of a product attribute. If the difference is significant, it may be attributed to process factors, such as deficient quality control. We apply this method to Shuttle software.

First determine whether the data are approximately normally distributed as required for using the t test [16]. In Figure 7 it is shown that the cumulative failure data Y_i are approximately normally distributed by virtue of the data (red) dots being close to the normal (blue) line.

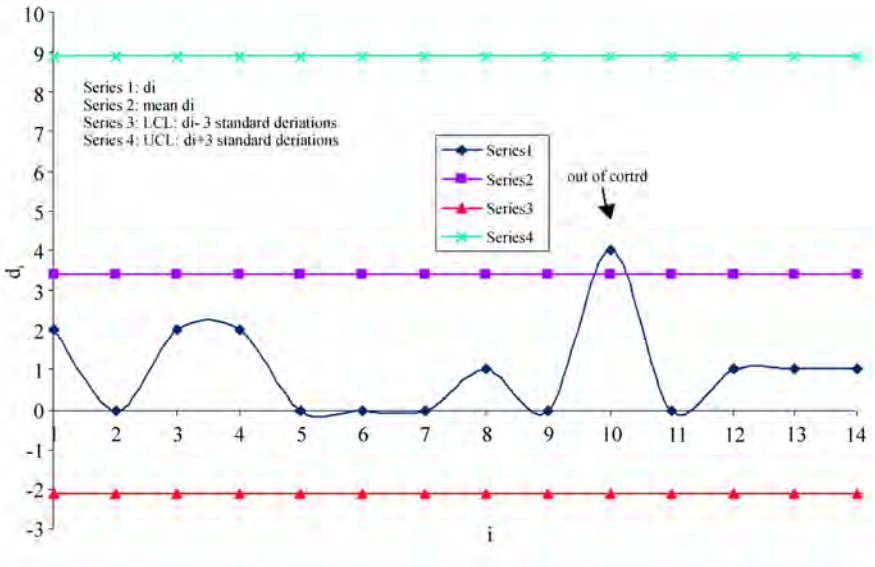


Figure 6. NASA space shuttle OI3; process control using poisson distribution: Number of failures d_i vs. test number i

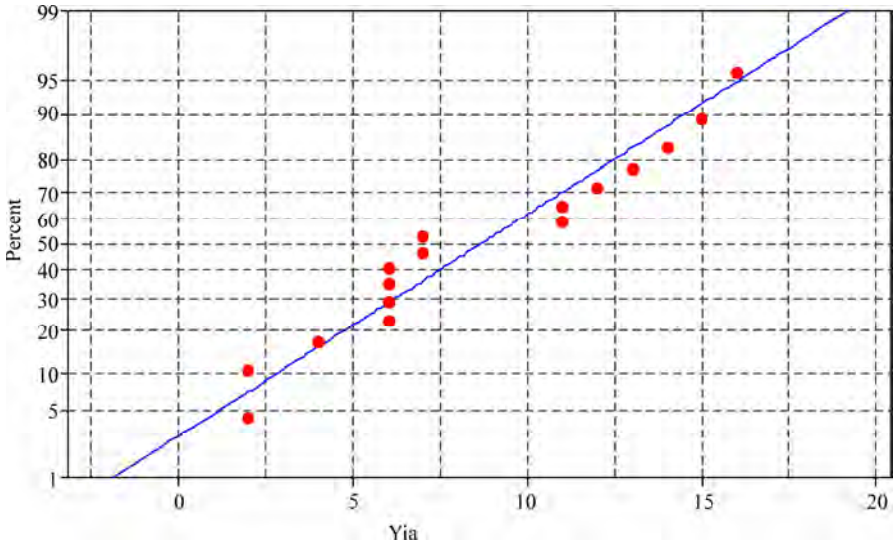


Figure 7. Normality test of cumulative number of failures yia

Table 2. Shuttle OI3 design of experiments

T_i	Y_{ia}		
2	2	s	t
2	4		1.7531
4	6		t table
5	6	**α**	0.05
8	6	df	15
9	6	No statistically	
9	7	Significant difference	
11	7		
12	11		
15	11		
19	13		
21	14		
24	15		
26	16		
11.5000	8.6250		
$\bar{T}_i$	$\bar{Y}_i$		
7.8825	4.5295		
S_{T_i}	S_{Y_i}		

Second, state the null H_0 and alternate H_1 hypotheses:

H_0: there *is no* statistically significant difference between T_i and Y_i

H_1: there *is a* statistically significant difference between T_i and Y_i

Third, compute the means of the cumulative failures and the target cumulative failures in Equations (14) and (15), respectively, and the standard deviation of the sum of the variance of Y_i, s_Y^2, and variance of T_i, s_T^2, in Equation (16)

$$\bar{Y}_i = \frac{\sum_{i=1}^{n} Y_i}{n} \tag{14}$$

$$\bar{T}_i = \frac{\sum_{i=1}^{n} T_i}{n} \tag{15}$$

$$s = \sqrt{s_{Y_i}{}^2 + s_{T_i}{}^2} \tag{16}$$

Fourth, based on the outcome of these computations, the t statistic is computed in Equation (17).

$$t = \frac{(\bar{Y}_i - \bar{T}_i)}{s} \tag{17}$$

Fifth, a comparison is made between the value computed from Equation (17) and the value obtained from the t statistic table. If (t < table value), accept H_0 and conclude that the difference between the target and actual cumulative failures is not statistically significant; otherwise, accept H_1 and conclude there is a difference, and attempt to assign the cause and correct the problem (e.g., inadequate software testing).

Based on Table 2, we accept H_0 and conclude there is not a statistically significant difference between cumulative failures and the target values, based on the computed t = 1.2650 < t table = 1.7531. Thus with an error of α = .05 of rejecting H_0, when in fact it is true, we have confidence that quality based on cumulative failures meets its goal.

8. Conclusions

We have presented some process methods from the field of manufacturing with the objective that the methods will prove useful to software engineers. While software development has some unique characteristics, we can learn from other disciplines, where the methods have been applied extensively on an international scale as part of a total quality management plan. Such a plan recognizes that process and product are intimately related and that the objective of software process improvement should be to improve software product quality.

The loss function and signal to noise ratio inspired by Taguchi methods proved to be valuable techniques for identifying and reducing excessive variation in software

quality. In addition, statistical process control provided a method for identifying the test when the incidence of software failures was out of control. This allows the software engineer to estimate the number of tests to conduct in order to determine whether the software product is under control. Finally, design of experiments methodology allowed us to conduct hypothesis tests to estimate whether software product metrics were achieving their goals.

REFERENCES

[1] J. G. Monks, "Operations management," Second Edition, McGraw-Hill, 1996.

[2] A. L. Jacob and S. K. Pillai, "Statistical process control to improve coding and code review," IEEE Software, Vol. 20, No. 3, pp. 50–55, May/June, 2003.

[3] T. Keller and N. F. Schneidewind, "A successful application of software reliability engineering for the NASA space shuttle," Software Reliability Engineering Case Studies, International Symposium on Software Reliability Engineering, Albuquerque, New Mexico, November 4, pp. 71–82, 1997.

[4] J. Q. Ning, "Component-based software engineering (CBSE)," 5th International Symposium on Assessment of Software Tools (SAST'97), p. 0034, 1997.

[5] A. Sloane and W. Waite, "Issues in automatic software manufacturing in the presence of generators," Australian Software Engineering Conference, p. 134, 1998.

[6] Y. L. Yang, M. Li, and Y. Y. Huang, "The use of configuration conception in software development," IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, Vol. 2, pp. 963–967, 2008.

[7] R. V. Binder, "Can a manufacturing quality model work for software?" IEEE Software, Vol. 14, No. 5, pp. 101–102,105, September/October 1997.

[8] M. Eiklenborg, S. Ioannou, G. King II, and M. Vilcheck, "Taguchi methods for achieving quality," San Francisco State University, School of Engineering. http://userwww.sfsu.edu/~gtarakji/engr801/wordoc/taguchi.html.

[9] R. K. Roy, "Design of experiments using the taguchi approach: 16 steps to product and process improvement," John Wiley & Sons, Inc., 2001.

[10] G. Taguchi, S. Chowdhury, and Y. Wu, "Taguchi's quality engineering handbook," John Wiley & Sons, Inc., 2005.

[11] Handbook of Software Reliability Engineering, Edited by Michael R. Lyu, Published by IEEE Computer Society Press and McGraw-Hill Book Company, 1996.

[12] N. F. Schneidewind, "Reliability modeling for safety critical software," IEEE Transactions on Reliability, Vol. 46, No. 1, pp. 88–98, March 1997.

[13] http://highered.mcgraw-hill.com/sites/dl/free/0072498919/95884/sch98919_ch09.pdf.

[14] N. Eickelmann and A. Anant, "Statistical process control: What you don't measure can hurt you!" IEEE Software, Vol. 20, No. 2, pp. 49–51, March/April, 2003.

[15] W. C. Turner, J. H. Mize, and J. W. Nazemetz, "Introduction to industrial and systems engineering," Third Edition, Prentice Hall, 1993.

[16] D. M. Levine, P. P. Ramsey, and R. K. Smidt, "Applied statistics for engineers and scientists," Prentice-Hall, 2001.

[17] J. D. Musa, A. Iannino, and K. Okumoto, "Software reliability: Measurement, prediction, application," McGraw-Hill, 1987.

[18] N. F. Fenton and S. L. Pfleeger, "Software metrics: A rigorous & practical approach," Second Edition, PWS Publishing Company, 1997.

Intelligent Information Management, **2009, 1, 108-119**
doi:10.4236/iim.2009.12016 Published Online November 2009 (http://www.scirp.org/journal/iim)

Scientific Research

Mediative Fuzzy Logic for Controlling Population Size in Evolutionary Algorithms

Oscar MONTIEL[1], Oscar CASTILLO[2], Patricia MELIN[3], Roberto SEPULVEDA[4]
[1,4]*CITEDI-IPN, Av. del Parque, Mesa de Otay, Tijuana, México*
[2,3]*Department of Computer Science, Tijuana Institute of Technology, Chula Vista, USA*
Email: {*oross, rsepulve*}*@citedi.mx*, {*ocastillo, pmelin*}*@tectijuana.mx*

Abstract: In this paper we are presenting an intelligent method for controlling population size in evolutionary algorithms. The method uses Mediative Fuzzy Logic for modeling knowledge from experts about what should be the behavior of population size through generations based on the fitness variance and the number of generations that the algorithm is being stuck. Since, it is common that this kind of knowledge expertise can be susceptible to disagreement in a minor or a major part. We selected Mediative Fuzzy Logic (MFL) as a fuzzy method to achieve the inference. MFL is a novelty fuzzy inference method that can handle imperfect knowledge in a broader way than traditional fuzzy logic does.

Keywords: mediative fuzzy logic, dynamic population size, HEM

1. Introduction

This paper has two mixed goals: one is to present a novelty intelligent system for controlling dynamics of population size in evolutionary algorithms; the second goal is to show an application of mediative fuzzy logic (MFL), a novel fuzzy inference method which is able to handle doubtful and contradictory knowledge, with any degree of mismatch. The goals are mixed because the intelligent system is using MFL to control the population size.

In any evolutionary algorithm there are many parameters to adjust, and generally they are adjusted by trial and error, one of them is the population size, so it is desirable to have an intelligent evolutionary algorithm, with the capacity of optimizing the population size without the risk of letting the algorithm to get stuck.

In real world applications is common that two experts can disagree in some part of the knowledge. Traditional fuzzy logic is unable to handle directly contradiction and hesitation in the knowledge.

Uncertainty affects all decision making and appears in a number of different forms. The concept of information is fully connected with the concept of uncertainty; the most fundamental aspect of this connection is that uncertainty involved in any problem-solving situation is a result of some information deficiency, which may be incomplete, imprecise, fragmentary, not fully reliable, vague, *contradictory*, or deficient in some other way [1]. The general framework of fuzzy reasoning allows handling much of this uncertainty.

Nowadays, we can handle much of this uncertainty using Fuzzy logic type-1 or type-2 [2,3], also we are able to deal with hesitation using Intuitionistic fuzzy logic in the Atanassov sense, but we have several questions: what happens when the information collected from different sources is somewhat or fully contradictory? What do we have to do if the knowledge base changes with time, and non-contradictory information becomes into doubtful or contradictory information, or any combination of these three situations? What should we infer from this kind of knowledge? The answer to these questions is to use a fuzzy logic system with logic rules for handling non-contradictory, contradictory or information with a hesitation margin. Mediative fuzzy logic is a novel approach presented for the first time in [4] which is able to deal with this kind of inconsistent information providing a common sense solution when contradiction exists, this is a mediated solution.

There are a lot of applications where information is inconsistent. In economics for estimating the Gross Domestic Product (GDP), it is possible to use different variables; some of them are distribution of income, personal consummation expenditures, personal ownership of goods, private investment, unit labor cost, exchange rate, inflation rates, and interest rates. In the same area for estimating the exportation rates it is necessary to use a combination of different variables, for example, the annual rate of inflation, the law of supply and demand, the dynamic of international market, etc. [5]. In medicine, information from experiments can be somewhat inconsistent because living being might respond different to some experimental medication. Currently, randomized clinical trials have become the accepted scientific standard for evaluating therapeutic efficacy, and contradic-

tory results from multiple randomized clinical trials on the same topic have been attributed either to methodological deficiencies in the design of one of the trials or to small sample sizes that did not provide assurance that a meaningful therapeutic difference would be detected [6]. In forecasting prediction, uncertainty is always a factor, because to obtain a reliable prediction it is necessary to have a number of decisions, each one based on a different group, in [7] says: Experts should be chosen "whose combined knowledge and expertise reflects the full scope of the problem domain. Heterogeneous experts are preferable to experts focused in a single specialty".

This paper is organized as follows. In Section 2, we are giving some historical antecedent about different logic systems. In Section 3, we are giving a description of the intelligent system for controlling population size in evolutionary algorithms. In Section 4, we are explaining concepts of Mediative Fuzzy Logic (MFL). In Section 5, we are explaining the proposed intelligent system. In Section 6, we are showing results of some experiments that we performed. In Section 7, we are making a discussion about the experiment and the obtained results. Finally in Section 8, we are giving the conclusions.

2. Historical Background

Throughout history, distinguish good from bad arguments has been of fundamental importance to ancient philosophy and modern science. The Greek philosopher Aristotle (384 BC – 322 BC) is considered a pioneer in the study of logic and, its creator in the traditional way. The Organon is his surviving collected works on logic [8]. Aristotelian logic is centered in the syllogism, in Traditional logic, a syllogism (deduction) is an inference that basically consists of three things: the major and minor premises, and the proposition (conclusion) which follows logically from the major and minor premises [9]. Aristotelian logic is "bivalent" or "two-valued", that is, the semantics rules will assign to every sentence either the value "True" or the value "False". Two basic laws in this logic are the law of contradiction (*p cannot be both p and not p*), and the law of the excluded middle (*p must be either p or not p*).

In the Hellenistic period, the stoics work on logic was very wide, but in general, one can say that their logic is based on propositions rather than in logic of terms, like the Aristotelian logic. The Stoic treatment of certain problems about modality and bivalence are more significant for the shape of Stoicism as a whole. Chrysippus (280BC-206BC) in particular was convinced that bivalence and the law of excluded middle apply even to contingent statements about particular future events or states of affairs. The law of excluded middle says that for a proposition, p, and its contradictory, $\neg p$, it is necessarily true, while bivalence insists that the truth table that defines a connective like 'or' contains only two values, true and false [10].

In the mid-19th century, with the advent of symbolic logic, we had the next major step in the development of propositional logic with the work of the logicians Augustus DeMorgan (1806-1871) [11] and, George Boole (1815-1864). Boole was primarily interested in developing special mathematical to replace Aristotelian syllogistic logic. His work rapidly reaps benefits, he proposed "Boolean algebra" that was used to form the basis of the truth-functional propositional logics utilized in computer design and programming [12,13]. In the late 19th century, Gottlob Frege (1848-1925) claimed that all mathematics could be derived from purely logical principles and definitions and he considered verbal concepts to be expressible as symbolic functions with one or more variables [14].

L. E. J. Brouwer (1881-1966) published in 1907 in his doctoral dissertation the fundamentals of intuitionism [15]. His student Arend Heyting (1898-1980) did much to put intuitionism in mathematical logic. He created the Heyting algebra for constructing models of intuitionistic logic [16]. Gerhard Gentzen (1909-1945), in (1934) introduces systems of natural deduction for intuitionist and classical pure predicate calculus [17], his cornerstone was cut-elimination theorem which implies that we can put every proof into a (not necessarily unique) normal form. He introduces two formal systems (sequent calculi) LK and LJ. The LJ system is obtained with small changes into the LK system and it is suffice for turning it into a proof system for intuitionistic logic.

Nowadays, Intuitionistic logic is a branch of logic which emphasizes that any mathematical object is considered to be a product of a mind, and therefore, the existence of any object is equivalent to the possibility of its construction. This contrasts with the classical approach, which states that the existence of an entity can be proved by refuting its non-existence. For the intuitionist, this is invalid; the refutation of the non-existence does not mean that it is possible to find a *constructive* proof of existence. Intuitionists reject the *Law of the Excluded Middle* which allows proof by contradiction. Intuitionistic logic has come to be of great interest to computer scientists, as it is a constructive logic, and is hence a logic of what computers can do.

Bivalent logic was the prevailing view in the development of logic up to XX century. In 1917, Jan Łukasiewicz (1878-1956) developed the three-value propositional calculus, inventing ternary logic [18]. His major mathematical work centered on mathematical logic. He thought innovatively about traditional propositional logic, the principle of non-contradiction and the law of excluded middle. Łukasiewicz worked on multivalued logics, including his own three-valued propositional calculus, the first non-classical logical calculus. He is responsible for one of the most elegant axioma-

tizations of classical propositional logic; it has just three axioms and is one of the most used axiomatizations today [19].

Paraconsistent logic is a logic rejecting the principle of non-contradiction, a logic is said to be *paraconsistent* if its relation of logical consequence is not explosive. The first paraconsistent calculi was independently proposed by Newton C. da Costa (1929-) [20] and Jaśkowski, and are also related to D. Nelson's ideas [21]. Paraconsistent logic was proposed in 1976 by the Peruvian philosopher Miró Quesada, it is a non-trivial logic which allows inconsistencies. The modern history of paraconsistent logic is relatively short. The expression "paraconsistent logic" is at present time well-established and it will make no sense to change it. It can be interpreted in many different ways which correspond to the many different views on a logic which permits to reason in presence of contradictions. There are many different paraconsistent logics, for example, *non-adjunctive*, *non-truth-functional*, *many-valued*, and *relevant*.

Fuzzy sets, and the notions of inclusion, union, intersection, relation, etc, were introduced in 1965 by Dr. Lofti Zadeh [2], as an extension of Boolean logic. Fuzzy logic deals with the concept of partial truth, in other words, the truth values used in Boolean logic are replaced with degrees of truth. Zadeh is the creator of the concept Fuzzy logic type-1 and type-2. Type-2 fuzzy sets are fuzzy sets whose membership functions are themselves type-1 fuzzy sets; they are very useful in circumstances where it is difficult to determine an exact membership function for a fuzzy set [22].

K. Atanassov in 1983 proposed the concept of Intuitionistic fuzzy sets (IFS) [23], as an extension of the well-known Fuzzy sets defined by Zadeh. IFS introduces a new component, degree of non-membership with the requirement that the sum of membership and non-membership functions must be less than or equal to 1. The complement of the two degrees to 1 is called the hesitation margin. George Gargov proposed the name of intuitionistic fuzzy sets with the motivation that their fuzzification denies the law of excluded middle, wish is one of the main ideas of intuitionism [24].

3. Intelligent System Description

In Figure 1, we are showing in general terms a method for controlling population size in evolutionary algorithms. This is an intelligent system that adapts the population size according to some predefined behavioral rules. We used MFL to achieve the inference, we selected this method because it can handle doubtful and contradictory knowledge provided by human experts, and moreover, the method is auto-reducible, so if there is no contradiction or hesitation in the knowledge, the system will behaves as a traditional fuzzy inference system.

Basically, the method is using two variables, the variance in the fitness value of the best individual in previous generations; and the degree of cycling that the algorithm is having.

In order to calculate the variance, we considered the last ten generations to perform the calculation. The degree of cycling is a way to measure how many generations has passed without any significant change in the variance measure. This method can be applied practically to any evolutionary algorithm, in our case we used the Human Evolutionary Model (HEM) since this method was designed originally to be used with this evolutionary method. In Figure 1, we can appreciate in the first five lines that they are devote to the method initialization, here N is the initial population size; in the evolution process, we can vary this population between an upper and lower bound (ub, and lb). The evolutionary algorithm will run "maxGen" number of generations,

```
1  N=Initial population size;
2  ub=Upper bound;
3  lb=Lower bound;
4  maxGen=Maximum number of Generations;
5  actPs=N;
6  for gen=0 ; maxGen
       Evolutionary
       Algorithm
   (fitness evaluation,
    operators, etc.)
7     if gen>11
8        genBefore=10;
9        variance=var(Best_Fitness(genBefore:gen));
10          If variance < 0.05
11            Cycling=Cycling+1;
12          else
13            Cycling=0
14          end
              . . .
15          y_aiis=calculate_aiis_hem(Cycling,Variance);
16          Create=round(y_aiis(1))*actPs;
17          Delete=round(y_aiis(2))*actPs;
18          while (actPS > lb)
19             while (Delete)
20                Delete the individual with small fitness value;
21                Delete=Delete-1;
22                actPs=actPs-1;
23             end
24          end
25          while (actPS < ub)
26             while (Create )
27                Generete a new individual;
28                Create=Create-1;
29                actPs=actPs+1;
30             end
31          end
32    end
```

Figure 1. Generic method for controlling population size using the variance of the fitness and the degree of cycling; i.e. the number of generations that the algorithms has ran without any significant change in the fitness value

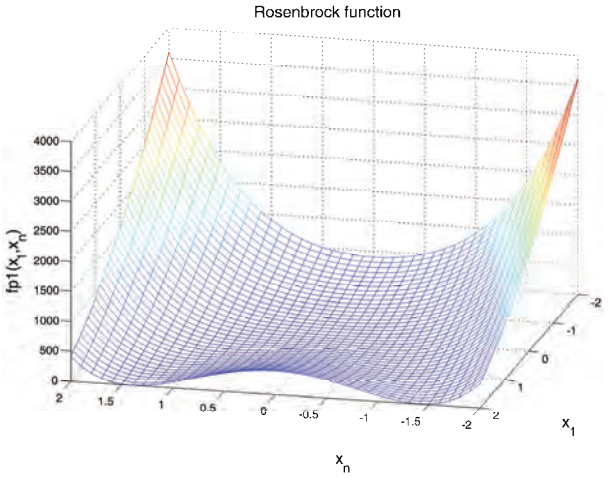


Figure 2. Inference system at the agreement side. Here, the system is defined using the agreement MFs

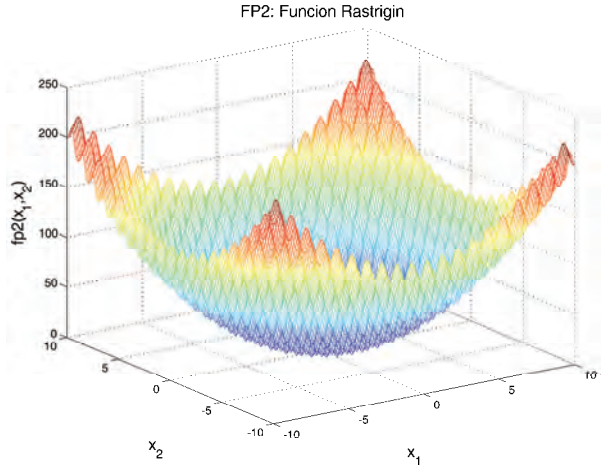


Figure 3. Inference system at the non-agreement side. The system is defined using the non-agreement MFs

although we can change this condition to a specific accuracy value. Next, the selected evolutionary algorithm will run one gener- ation at each cycle, after that we will measure the variance and degree of cycling that the algorithm is having. To measure the cyclyng we are using the variable "Cycling", an increment in this variable is done every time that the variance in the fitness is below a determined threshold value (0.05 in this case), otherwise the variable is reset.

The intelligent system will calculate the amount of individuals to delete and to create using the inference system based in MFL, this action is expressed in Figure 1 by means of the function "calculate_aiis_hem". This function has two input parameters, they are "Cycling" and "Variance"; and the output "y_aiis" is a vector with two values, they are proportional indexes. The valid range of each index is [0,1]. Hence, the amount of individuals to create is calculated using the formula of line 16, and the amount of individuals to delete using the formula of line 17. Next, the algorithm goes into a process of deleting the worst (less fit) individual in the actual population expressed by the variable "Delete", then we will create an amount of individuals expressed by the variable "Create". Both processes are controlled to be into a valid range delimited by the upper and lower bound in the population size.

4. Mediative Fuzzy Logic

Since knowledge provided by experts can have big variations and sometimes can be contradictory, we are proposing to use a Contradiction fuzzy set to calculate a mediation value for solving the conflict. Mediative Fuzzy Logic is proposed as an extension of traditional fuzzy logic and includes Intuistionistic Fuzzy Logic (IFL) in the Atanassov sense [23,25]. Hence, it is able to handle contradictory and doubtful information.

A traditional fuzzy set in X [25], is given by

$$A = \{(x, \mu_A(x)) | x \in X\} \quad (1)$$

where $\mu_A : X \to [0, 1]$ is the membership function of the fuzzy set A.

An IFL set B is given by

$$B = \{(x, \mu_B(x), \nu_B(x)) | x \in X\} \quad (2)$$

where $\mu_B : X \to [0, 1]$ and $\nu_B : X \to [0, 1]$ are such that

$$0 \le \mu_B(x) + \nu_B(x) \le 1 \quad (3)$$

and $\mu_B(x)$; $\nu_B(x) \in [0, 1]$ denote a degree of membership and a degree of non-membership of $x \in A$, respectively.

For each IFL set in X we have a "hesitation margin" $\pi_B(x)$, this is a fuzzy index of $x \in B$, it expresses a hesitation degree of whether x belongs to A or not. It is obvious that $0 \le \pi_B(x) \le 1$, for each $x \in X$.

$$\pi_B(x) = 1 - \mu_B(x) - \nu_B(x) \quad (4)$$

Therefore if we want to fully describe an IFL set, we must use any two functions from the triplet [10].

1) Membership function
2) Non-membership function
3) Hesitation margin

The application of IFL sets instead of fuzzy sets, means the introduction of another degree of freedom into a set description, in other words, in addition to μ_B we also have ν_B or π_B. IFL considers the fact that we have the membership functions μ as well as the non-membership functions ν. Hence, the output of an IFL system can be calculated as follows:

$$IFS = (1 - \pi) FS_\mu + \pi FS_\nu \quad (5)$$

where FS_μ is the traditional output of a fuzzy system using the membership function μ, and FS_ν is the output of a fuzzy system using the non-membership function ν. Note in Equation (6), when π=0 the *IFS* is reduced to the output of a traditional fuzzy system, but if we take into account the hesitation margin of π the resulting IFS will be different.

In similar way, a contradiction fuzzy set C in X is given by:

$$\zeta_C(x) = \min(\mu_C(x), \nu_C(x)) \quad (6)$$

where $\mu_C(x)$ represents the agreement membership function, and for the variable $\nu_C(x)$ we have the non-agreement membership function.

We are using the agreement and non-agreement instead membership and non-membership, because we think these names are more adequated when we have contradictory fuzzy sets.

For calculating the inference at the system's output, we are using

$$MFS = \left(1 - \pi - \frac{\zeta}{2}\right) FS_\mu + \left(\pi + \frac{\zeta}{2}\right) FS_\nu \quad (7)$$

5. Defining the Intelligent System

We used MFL to control the population size of HEM by killing and/or creating individuals in the new population. A Sugeno Inference System at the agreement side ($AIIS_a$) was used to calculate the fuzzy outputs Create and Delete. In the same way, at the non-Agreement side ($AIIS_{na}$) we have the fuzzy outputs Create and Delete. See in Figures 4 and 5 a block diagram of both inference systems.

For both inference systems, we defined two linguistic input variables: Cycling and Variance.

- The variable Cycling is representing the amount of generations that has passed with any significative change in the variance value of the fitness over a

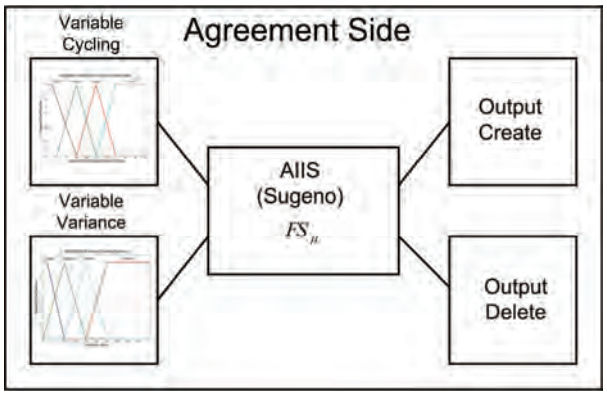


Figure 4. Inference system at the agreement side. Here, the system is defined using the agreement MFs

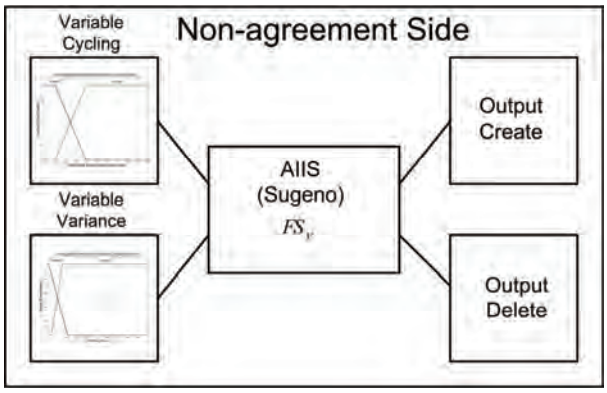


Figure 5. Inference system at the non-agreement side. The system is defined using the non-agreement MFs

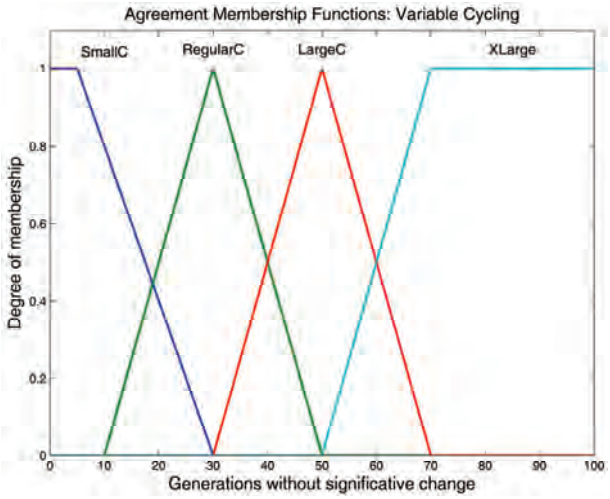


Figure 6. MFs for the variable Cycling at the agreement side in a traditional fuzzy system.

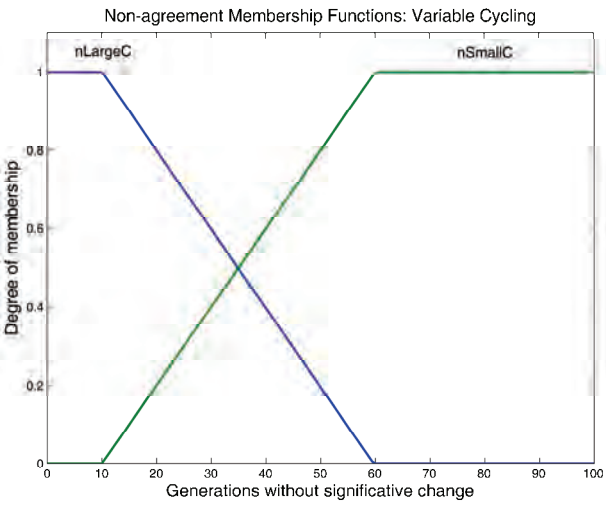


Figure 7. MFs for the variable Cycling a the non-agreement side

determined number of generations.

- The variable Variance is calculated using the actual and past values of the population's fitness. We used the fitness value obtained in the last ten generations (including the actual generation).

For taking into account an increment in the variable Cycling, we used a threshold variance value of 0.05, any value below the threshold increases the Cycling counter, otherwise the variable Cycling must be reset.

At the agreement side, the input variables Cycling has four terms: SmallC, RegularC, LargeC, XLargeC; see Figure 6. The input variable Variance has four terms: SmallV, RegularV, LargeV, XLargeV, see Figure 8.

At the non-agreement side, the input variable Cycling has two terms: nLargeC, nSmallC; see Figure 7. The input variable Variance also has two terms: nLargeV, nSmallV, see Figure 9.

In Table 1, we are summarizing the the names, type and parameters for each input variable.

Each Inference fuzzy system has two output variables: Delete and Create. They correspond to the amount of individuals that we have to delete in the actual population, and to create in the new population. Each output variable has three constant terms: little, regular and many. We used the same constant term names and values for every term at each output of both Sugeno type 0 inference systems. We assigned the values of "0", "0.5", and "1" to the variables little, regular and many, respectively.

At the agreement side we have the next fuzzy rules:

1. If (Cyicling is SmallC) and (Variace is SmallV) then (Create is little)(Delete is little)
2. If (Cyicling is SmallC) and (Variace is RegularV) then (Create is regular)(Delete is little)
3. If (Cyicling is SmallC) and (Variace is LargeV) then (Create is regular)(Delete is regular)
4. If (Cyicling is RegularC) and (Variace is SmallV) then (Create is many)(Delete is regular)
5. If (Cyicling is RegularC) and (Variace is RegularV) then (Create is regular)(Delete is regular)
6. If (Cyicling is RegularC) and (Variace is LargeV) then (Create is regular)(Delete is regular)
7. If (Cyicling is LargeC) and (Variace is SmallV) then (Create is many)(Delete is many)
8. If (Cyicling is LargeC) and (Variace is RegularV) then (Create is many)(Delete is regular)
9. If (Cyicling is LargeC) and (Variace is LargeV) then (Create is little)(Delete is little)
10. If (Cyicling is SmallC) and (Variace is XLargeV) then (Create is little)(Delete is regular)
11. If (Cyicling is RegularC) and (Variace is XLargeV) then (Create is regular)(Delete is little)
12. If (Cyicling is LargeC) and (Variace is XLargeV) then (Create is little)(Delete is little)
13. If (Cyicling is XLargeC) and (Variace is SmallV) then (Create is many)(Delete is many)
14. If (Cyicling is XLargeC) and (Variace is RegularV) then (Create is many)(Delete is many)
15. If (Cyicling is XLargeC) and (Variace is XLargeV) then (Create is many)(Delete is many)
16. If (Cyicling is XLargeC) and (Variace is LargeV) then (Create is many)(Delete is many)

Table 1. Linguistic input variables for the fuzzy system at the agreement side ($AIIS_a$), and for fuzzy system at the non-agreement side $AIIS_{na}$

FIS name	*Variable name*	*Term name*	*Type*	*Parameters*
$AIIS_a$ (FS_μ)	Cycling	SmallC	Trapezoidal	[-1,-1,5,30]
$AIIS_a$(FS_μ)	Cycling	RegularC	Triangular	[10,30,50]
$AIIS_a$ (FS_μ)	Cycling	LargeC	Triangular	[30,50,70]
$AIIS_a$ (FS_μ)	Cycling	XLargeC	Trapezoidal	[50,70,5e+15,5e+15]
$AIIS_a$ (FS_μ)	Variance	SmallV	Trapezoidal	[-1,-1,0.04,0.2]
$AIIS_a$ (FS_μ)	Variance	RegularV	Triangular	[0,0.2,0.4]
$AIIS_a$ (FS_μ)	Variance	LargeV	Triangular	[0.2,0.4,0.6]
$AIIS_a$ (FS_μ)	Variance	XLargeV	Trapezoidal	[0.4,0.6,5e+15,5e+15]
$AIIS_{na}$ (FS_ν)	Cycling	nLargeC	Trapezoidal	[-1,-1,10,60]
$AIIS_{na}$ (FS_ν)	Cycling	nSmallC	Trapezoidal	[10,60,5e+15,5e+15]
$AIIS_{na}$ (FS_ν)	Variance	nLargeV	Trapezoidal	[-1,-1,0.05,1]
$AIIS_{na}$ (FS_ν)	Variance	nSmallV	Trapezoidal	[-1,0.09,1,5e+15]

At the non-agreement side, we have the next fuzzy rules:

1. If (Cyicling is nLargeC) and (Variance is nLargeV) then (Create is little)(Delete is little)
2. If (Cyicling is nLargeC) and (Variance is nSmallV) then (Create is little)(Delete is little)
3. If (Cyicling is nSmallC) and (Variance is nLargeV) then (Create is many)(Delete is many)
4. If (Cyicling is nSmallC) and (Variance is nSmallV) then (Create is regular)(Delete is regular)

6. Experiments

For Experiments 1 and 2, we ran the evolutionary algorithm 1500 generations with the purpose of analyzing the dynamic behavior of the population size. We used an initial population size of 100 individuals, the selected upper and lower bounds (ub and lb) to control population size within a range were 600 and 20 individuals respectively.

In Experiment 3, we only changed the initial population size to 20.

6.1. Experiment #1. 4-Dimensions Rosenbrock Function

The Rosenbroc function is given by Equation (7). This function has its minimal value at $x^* = [1,1,1,1]$. After 1500 generations we found the value $x^* = [0.92,0.85,0.72,0.51]$, and the corresponding fitness value is 9.08, it last 31 seconds. Figure 22 shows the fitness behavior and Figure 23 shows how the size of population is changing in order to help the algorithm to obtain an increment in the overall fitness.

$$F2(\vec{x}) = \sum_{i=1}^{N-1} \left(100 * \left(x_{i+1} - x_i^2 \right)^2 + \left(x_i - 1 \right)^2 \right) \qquad (7)$$

6.2. Experiment #2. 4-Dimensions Rastrigin Function

The Rastriging function is given by Equation (8). This function has its minimal in $x^* = [0,0,0,0]$. After 1500 generations we found a fitness values around 1.16×10^9, in this example the minimal value founded is in $x^* = \left[0.0, 0.049 \times 10^{-5}, -0.2 \times 10^{-5}, -0.3 \times 10^{-5}\right]$. The algorithm last 33.8 seconds, but in fact the algorithm is able to find values with very high fitness more or less in 20 generations. In Figure 24, we are showing the fitness evolution through 1500 generations.

In the Rastrigin function, *n* is the dimension of the problem, and the variables A and *w* control the amplitude and frequency modulation respectively.

$$F(\vec{x}) = A \cdot n + \sum_{i=1}^{n} x_i^2 - A \cdot \cos(w \cdot x_i) \qquad (8)$$

6.3. Experiment #3. 4-Dimension Rastrigin Function

In Figure 24 we are showing a second run, here we used 20 generations, and the other test conditions were the same. In this case, the algorithm last 0.5540 seconds, and the values found were $x^* = [0.0117, 0.0062, 0.0014, 0.0040]$, with a fitness value of 26.03.

7. Discussion

The agreement membership functions and the non-agreement membership functions were collected from two different experts. It was easier to define the agree-

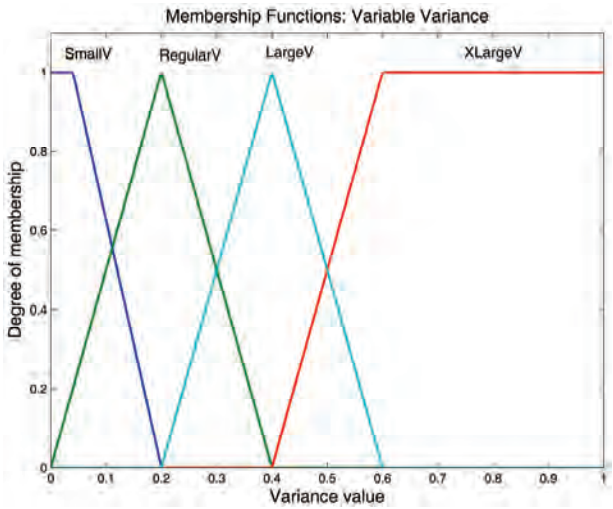


Figure 8. MFs for the variable variance at the agreement side

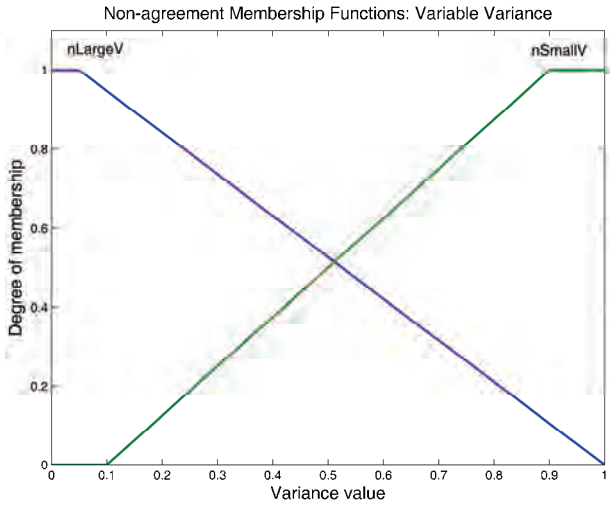


Figure 9. MFs for the variable variance at the non-agre ement side

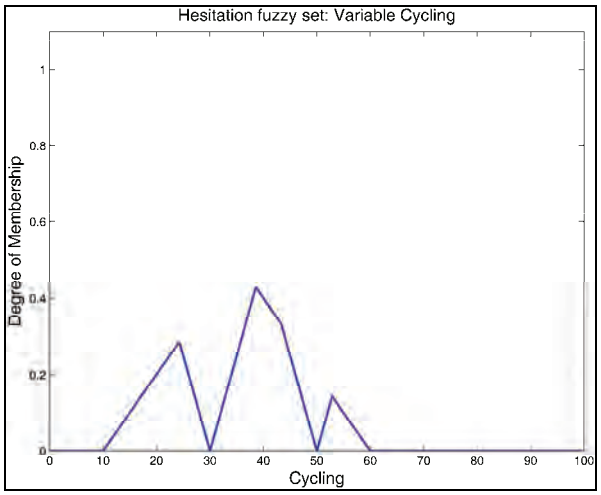


Figure 10. Hesitation fuzzy set for the variable cycling

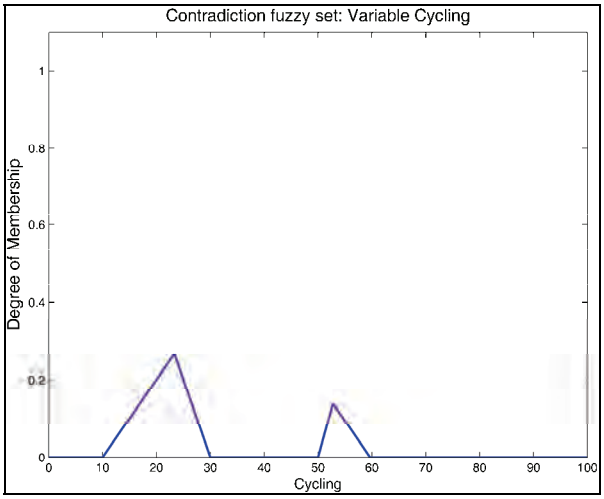


Figure 11. Contradiction fuzzy set for the variable cycling

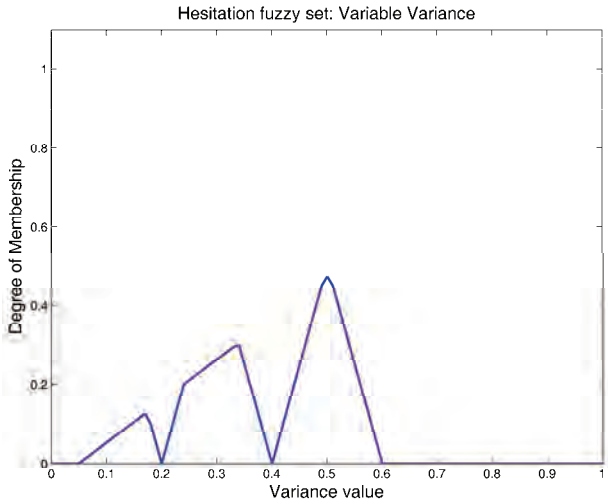


Figure 12. Hesitation fuzzy set for the variable variance

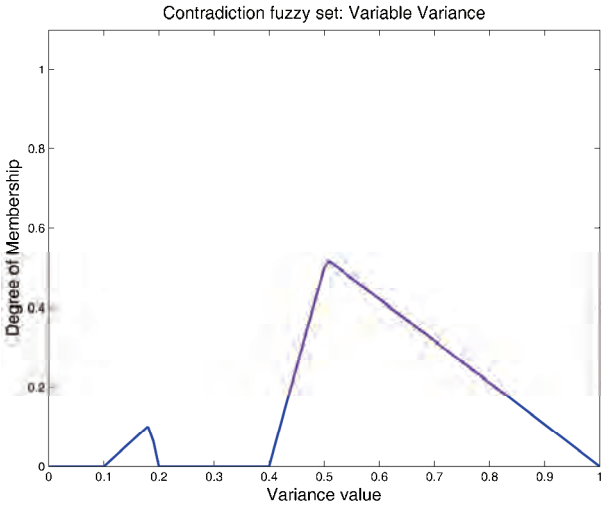


Figure13. Contradiction fuzzy set for the variable variance

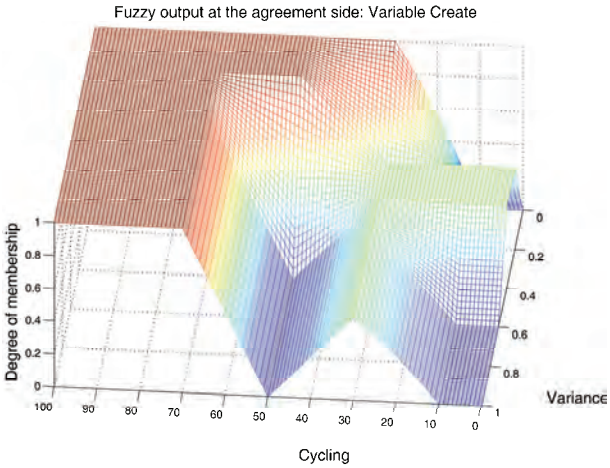


Figure 14. Surface of the fuzzy output create at the agreement side of a traditional fuzzy system

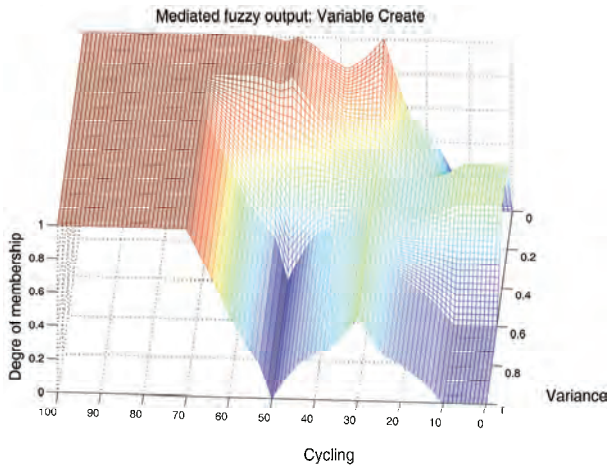


Figure 15. Surface of the mediated fuzzy output create

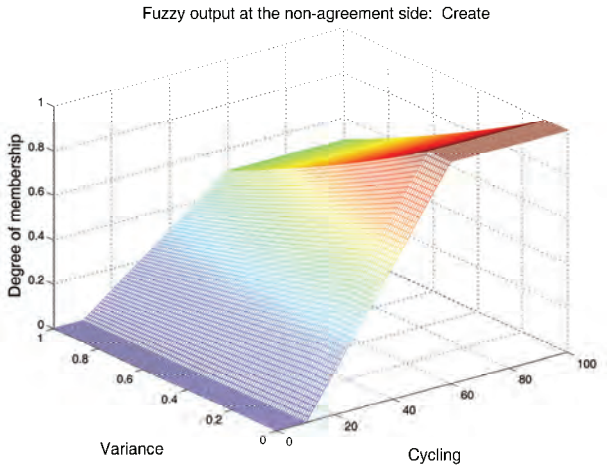


Figure 16. Surface of the fuzzy output create at the non-agreement side

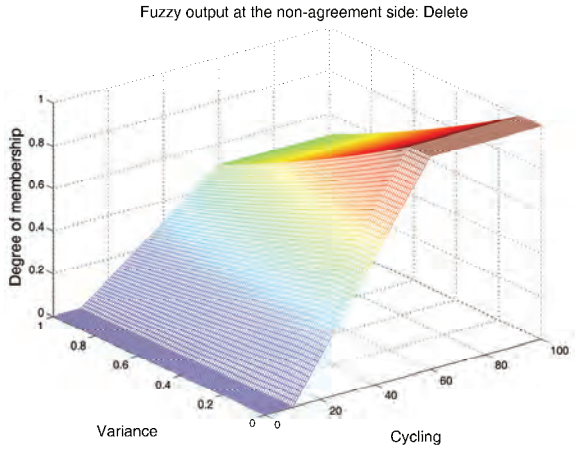


Figure 17. Surface of the mediated fuzzy output delete

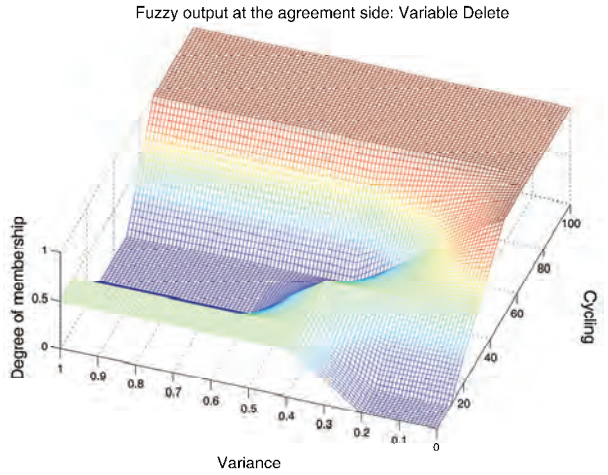


Figure 18. Surface of the variable delete at the agreement side in a traditional fuzzy system

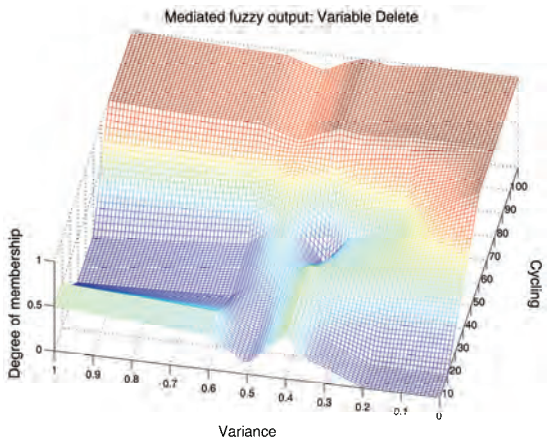


Figure 19. Surface of the mediated fuzzy output delete

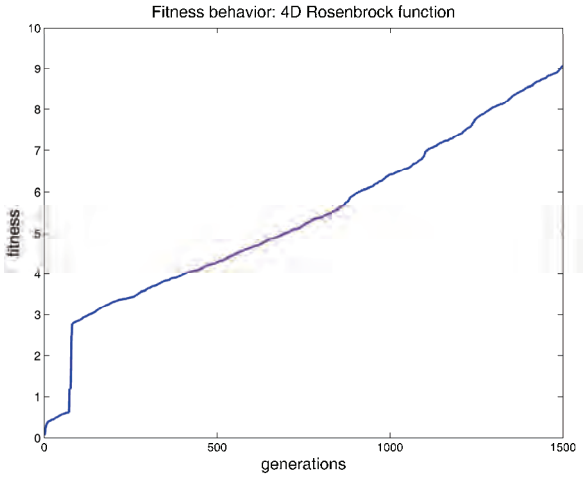


Figure 20. This fitness plot was obtained as a result of finding the minimizers for the 4-D Rosenbrock function. Experiment 1

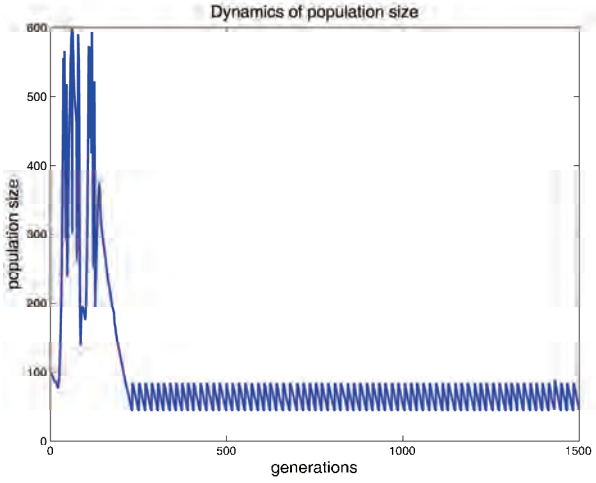


Figure 21. Dynamical behavior of population when the evolutionary algorithm is finding the function's minimizers. Experiment 1

ment function because is more natural to establish MFs using positive logic, so one expert defined five agreement functions for each input variable (Figures 6 and 8). Defining the non-agreement functions is less intuitive because we need to use negative logic to establish them, so the other expert only could establish two non-agreement membership functions for each input variable (Figures 7 and 9). In Figures 10 and 12, we are showing the hesitation fuzzy set for the variables Cycling and Variance respectively; these sets were obtained using Equation (4). The contradiction fuzzy sets for the abovementioned linguistic variables were obtained using Equation (5), they are shown in Figures 11 and 13. Note that the fuzzy sets hesitation and contradiction becomes bigger as long as disagreement between experts increase, at the other hand, these two fuzzy sets will be vanish as long as experts agree in their knowledge. For the ideal case, when the two experts agree in full, MFL will be reduced to a traditional fuzzy inference system, passing through IFL.

In Figures 14 and 18, we are showing the surface plot for the outputs Create and Delete of a traditional fuzzy inference system at the agreement side. For the same variables but at the non-agreement side, Figures 16 and 17 show their corresponding output surface plots. Figures 15 and 19 are the surface plot of the mediated fuzzy output for the variables Create and Delete. Comparing Figures 14 and 15, and Figures 18 and 19 we can observe that there are important differences between them. Differences for the variable Create are in the range of -0.12 to 0.18, and for the variable Delete are in the range of -0.43 to .36. Although at first sight differences in some parts of the surface are small they could be very significant in some applications. In this problem for example, considering for the variable Delete the value of -0.43 in a population of 100 individuals means a reduction of 43 individuals more than the proposed by the traditional fuzzy inference system, this reduction in population size will speed up the algorithm execution.

We tested the complete D'Jong's test function set using the intelligent control system for controlling the population size of evolutionary algorithms. From this test set, we considered interesting the results obtained in the Rosenbrock and Rastrigin function with four variables to optimize (4 dimensions).

We selected the Rosenbrock function because most of the optimization methods exhibits slow convergence rate when they are trying to solve this problem [27]. In Experiment 1, in Figure 20 we can see how the fitness of the best individual at each generation behaves trough 1500 generations, and in Figure 21 its corresponding plot of population dynamics. In these figures, we can see that at early generations the EA got stuck and the intelligent system inferred that it was necessary to create more individuals for going out this condition. After several increment and decrement in population size the algorithm found an optimal operational range in population size below the initial population size (100 individuals) proposed to solve this problem. This global decrement in population size will reduce the execution time of each generation.

In Experiments 2 and 3, we use the Rastrigin function. In Figure 22 of Experiment 2, it is shown the fitness behavior of the best individual through 1500 generations; in Figure 23 we are showing a plot of the dynamic behavior of population size for this experiment, here the populations size grows when the intelligent system de

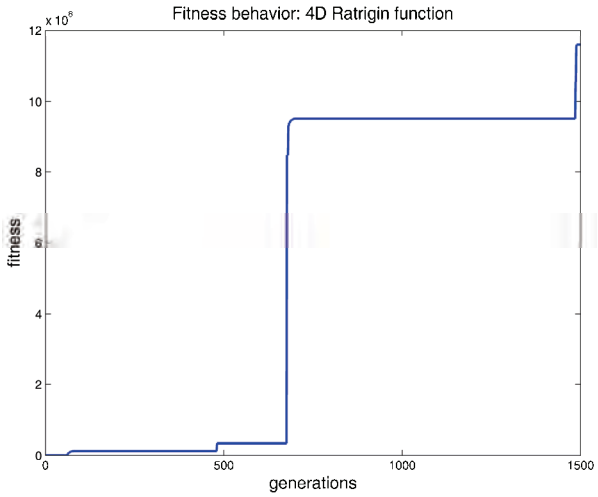


Figure 22. Fitness plot of the Rastrigin function. Note in the fitness value, that the evolutionay algorithm several times got stuck, but it went out thanks to the delete/create mechanism implemented via the intelligent system. Experiment 2

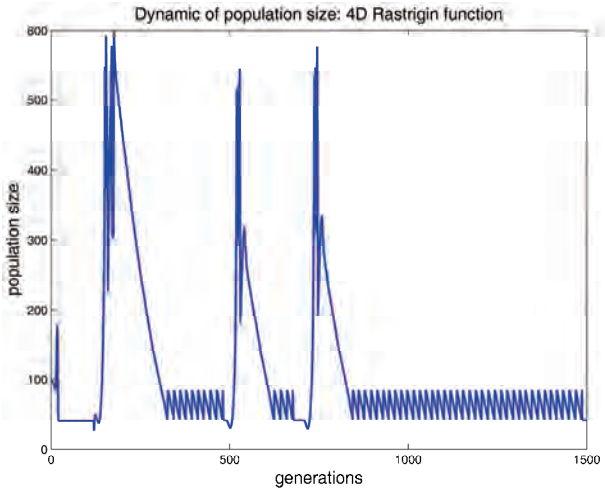


Figure 23. Plot of the dynamic behavior of population in Experiment 2. Compare this figure with Figure 20, and note that when the EA got stuck, the intelligent system decided to increase the population size

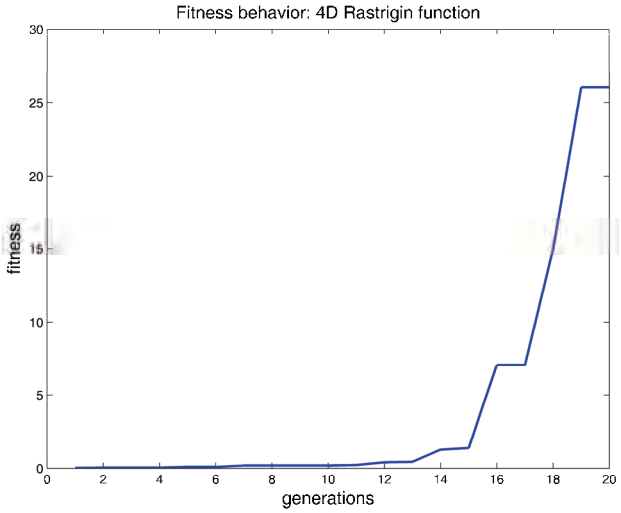


Figure 24. Here we are showing the fitness value behavior of 20 generations, they are enough to obtain an acceptable minimizer for explanatory purpose. Experiment 3

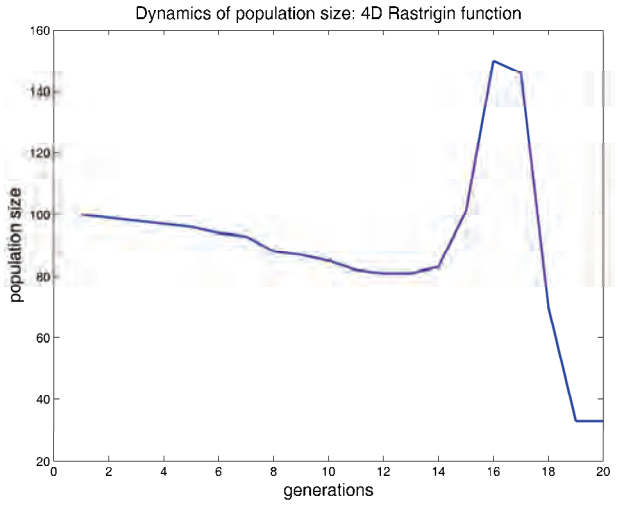


Figure 25. Behavior of population size in 20 generations when we are optimizing the 4-D Rastrigin function

tects that the algorithm is getting stuck, once the problem has been solved the intelligent system infers that it is necessary to decrease the population size.

In Experiment 3, we change the number of maximal generations that we are going to let the algorithm to evolve, in Figure 24 we are showing the fitness plot, and in Figure 25 the dynamics of population size for this experiment. Note that we needed only 20 generations to find an "acceptable" solution, and how the population size behaves.

We did comparative test of this method against the genetic algorithm of Matlab vers. 7.1 using a Pentium 4 based system (HP Pavillon zd8000), and we observed that we obatained better results in precision and time when we used the intelligent system for reducing population size. These results were more noticeable when we used more variables to optimize.

8. Conclusions

Through time fuzzy logic type-1 and type-2 have demonstrated their usefulness for handling uncertainty in uncountable applications. Mediative fuzzy logic is a novel approach that enables us to handle imperfect knowledge in a broader way than traditional and IF fuzzy logic do. MFL is a sort of paraconsistent fuzzy logic because it can handle contradictory knowledge using fuzzy operators. MFL is a concept that takes into account the contradiction fuzzy set to provide a mediated solution in case of a contradiction, moreover it can be reduced automatically to intuitionistic and traditional fuzzy logic in an automated way, this is depending on how the membership functions (agreement and non-agreement functions) are established. MFL is a good option when we have knowledge from different human experts, because it is common that experts do not fully agree all the time, so we can obtain contradiction fuzzy sets to represent the amount of disagree with the purpose of impacting the inference result. Traditional FL, and IFL will not impact the output when we have contradictory knowledge. Moreover, MFL allows us to watch for the exception to the rule because we can establish membership functions using negative logic (disagreement functions), this ability give us the power of including knowledge that sometimes is difficult to describe because we need to consider many situations that are sometimes imperceptible. Hence, the fact of having the possibility of complementing the knowledge with non-agreement functions will give us the possibility of implementing a more realistic fuzzy inference system.

The intelligent system for controlling the population size performs very good using MFL. Ideas for improving this inference system are:

1) Tuning the inference system: MFs and rules

2) Consider if we are having positive or negative change in the variance values.

3) Test with other equation to calculate the mediated output.

REFERENCES

[1] G. J. Klir, B. Yuan, "Fuzzy sets and fuzzy logic theory and applications," Edition, Prentice Hall USA, 1995.

[2] L. A. Zadeh, "Fuzzy sets," Information and Control, Vol. 8, pp. 338–353, 1965.

[3] J. M. Mendel, "Uncertain rule-based fuzzy logic systems introduction and new directions," Edition, Prentice Hall, USA, 2000.

[4] O. Montiel, O. Castillo, P. Melin, A. Rodríguez Días, and

R. Sepúlveda, ICAI, 2005.

[5] D. A. Bal, and W. H. McCulloch, Jr., "International business introduction and essentials," Fifth Edition, pp. 138–140, 225, USA, 1993.

[6] R. I. Horwitz, "Complexity and contradiction in clinical trial research," American Journal of Medicine, Vol. 8, pp. 498–510, 1987.

[7] J. S. Armstrong, "Principles of forecasting, A handbook for researchers and practitioners," Edited by J. Scott Armstrong, University of University of Pennsylvania, Wharton School, Philadelphia, PA., USA, 2001.

[8] Aristotle, "The basic works of Aristotle," Modern Library Classics, Richard McKeon Edition, 2001.

[9] R. Smith, "Aristotle's logic, Stanford encyclopedia of philosophy," 2004. http://plato.stanford.edu/entries/aristotle-logic/.

[10] D. Baltzly, "Stanford encyclopedia of philosophy," 2004. http://plato.stanford.edu/entries/stoicism/.

[11] J. J. O'Connor E. F. Robertson, and A. DeMorgan, "MacTutor history of mathematics: Indexes of biographies (University of St. Andrews)," 2004, http://www-groups.dcs.st-andrews.ac.uk/~history/Mathematicians/De_Morgan.htm.

[12] G. Boole, "The calculus of logic," Cambridge and Dublin Mathematical Journal, Vol. 3, pp. 183–98, 1848.

[13] J. J. O'Connor, E. F. Robertson, G. Boole, and MacTutor history of mathematics: Indexes of biographies," University of St. Andrews, 2004. http://www-groups.dcs.st-andrews.ac.uk/~history/Mathematicians/Boole.html.

[14] J. J. O'Connor, E. F. Robertson, F. L. G. Frege, and Mac Tutor history of mathematics: Indexes of Biographies University of St. Andrews, http://www-groups.dcs.st-andrews.ac.uk/~history/Mathematicians/Frege.html.

[15] J. J. O'Connor and E. F. Robertson , L. E. J. Brouwer, MacTutor history of mathematics: Indexes of biographies," University of St. Andrews, http://www-groups.dcs.st-adrews.ac.uk/~history/Mathematicians/Brouwer.html.

[16] J. J. O'Connor, E. F. Robertson, and A. Heyting, "MacTutor history of mathematics: Indexes of biographies," University of St. Andrews, 2004. http://www-history.mcs.st-adrews.ac.uk/Mathematicians/Heyting.html.

[17] J. J. O'Connor, E. F. Robertson, and G. Gentzen, "MacTutor history of mathematics: Indexes of biographies," University of St. Andrews, 2004. http://www-history.mcs.st-andrews.ac.uk/Mathematicians/Gentzen.html.

[18] J. J. O'Connor, E. F. Robertson, and J. Lukasiewicz, "MacTutor history of mathematics: Indexes of biographies," University of St. Andrews, 2004. http://www-history.mcs.st-andrews.ac.uk/Mathematicians/Lukasiewicz.html.

[19] Wikipedia the free encyclopedia, http://en.wikipedia.org/wiki/Jan_Lukasiewicz.

[20] Wikipedia the free encyclopedia, http://en.wikipedia.org/wiki/Newton_da_Costa.

[21] W. A. Carnielli, "How to build your own paraconsistent logic: An introduction to the logics of formal (in) consistency," In: J. Marcos, D. Batens, and W. A. Carnielli, organizers, Proceedings of the Workshop on Paraconsistent Logic (WoPaLo), held in Trento, Italy, as part of the 14th European Summer School on Logic, Language and Information (ESSLLI'02), pp. 58–72, 5–9 August 2002.

[22] J. M. Mendel and R. B. John, "Type-2 fuzzy sets made simple," IEEE Transactions on Fuzzy Systems, Vol. 10, No. 2, April 2002.

[23] K. Atanassov, "Intuitionistic guzzy dets: Theory and spplications," Springer-Verlag, Heidelberg, Germany, 1999.

[24] M. Nikilova, N. Nikolov, C. Cornelis, and G. Deschrijver, "Survey of the research on intuitionistic fuzzy sets," In: Advanced Studies in Contemporary Mathematics, Vol. 4, No. 2, , pp. 127–157, 2002.

[25] C. P. Melin, "A new method for fuzzy inference in intuitionistic fuzzy systems, proceedings of the international conference," NAFIPS'03, IEEE Press, Chicago, Illinois, USA, Julio, pp. 20–25, 2003.

[26] O. Nelles, "Nonlinear system identification from classical approaches to neural networks and fuzzy models," Springer-Verlag, Germany, 2001.

Intelligent Information Management, **2009, 1, 120-121**
doi:10.4236/iim.2009.12017 Published Online November 2009 (http://www.scirp.org/journal/iim)

Frucht Graph is not Hyperenergetic

S. PIRZADA
Department of Mathematics, University of Kashmir, Kashmir, India
Email: *sdpirzada@yahoo.co.in*

Abstract: If $\lambda_1, \lambda_2, ..., \lambda_p$ are the eigen values of a p-vertex graph G, the energy of G is $E(G) = \sum_{i=1}^{p} \lambda_i$. If $E(G) > 2p-2$, then G is said to be hyperenergetic. We show that the Frucht graph, the graph used in the proof of well known Frucht's theorem, is not hyperenergetic. Thus showing that every abstract group is isomorphic to the automorphism group of some non-hyperenergetic graph. AMS Mathematics Subject Classification: 05C50, 05C35

Keywords: energy of a graph, hyperenergetic, frucht graph, automorphism

1. Introduction

The concept of hyperenergetic graphs was introduced by Gutman [1]. The existence of hyperenergetic graphs has been known for quite some time, their systematic design was first achieved by Walikar *et al.* [2]. Details can also be seen in [3–6].

The following result can be found in [4].

Theorem 1. A graph with p vertices and m edges such that $m < 2p-2$ is not hyperenergetic.

In this paper, we give the existence of one more class of hyperenergetic graphs, called the Frucht graphs.

2. Frucht Graph is not Hyperenergetic

Let Γ be a group with n elements and Λ be a set of Γ not containing the identity e. The Cayley digraph is defined to be the digraph with vertex set $V = \Gamma$ and arc set $A = \{(g, gh) : h \in \Lambda\}$. It is denoted by $D = D(\Gamma, \Lambda)$. If $\Lambda = \Gamma - e$, the resulting Cayley digraph is complete and is denoted by $K = K(\Gamma, \Lambda)$. If Λ is a set of generators for Γ, the Cayley digraph is called the basic Cayley digraph.

In the Cayley digraph $D = D(\Gamma, \Lambda)$, if (g, h) is an arc, $k = gh$ for some $h \in \Lambda$, that is $g^{-1}k \in \Lambda$ and $g^{-1}k$ is called the color of (g, k).

An automorphism of D is said to be color preserving if it preserves the colors of the arcs. It is well known that the group $C(D)$ of color preserving automorphisms of the Cayley digraph $D = D(\Gamma, \Lambda)$ is isomorphic to Γ.

The following result is the Frucht's Theorem [7–9].

Theorem 2. Every group is isomorphic to the automorphism group of some graph.

While proving Theorem 2, Frucht obtained the graph G_1 from $D(\Gamma, \Lambda)$ called as Frucht graph, whose automorphism group is isomorphic to $C(D)$.

The following is the construction of Frucht graph G_1.

Replace each arc $g_i g_j$ of D by a figure joining vertices g_i and g_j. The figure consists of the 3-path $g_i u_i v_i g_j$, and two paths- a path p_{2k} (containing $2k$ vertices) rooted at u_i, and a path p_{2k+1} (containing $2k+1$ vertices) rooted at v_i, where $g_i^{-1} g_j = g_k$ is the color of $g_i g_j$. (Note that there will be a similar figure corresponding to $g_j g_i$ for a different k).

Clearly, the Frucht graph $G_1(\Gamma)$ with $\Lambda = s$ has $n(s+1)(2s+1)$ vertices.

Theorem 3. The Frucht graph G_1 has $ns(2s+1)$ edges.

Proof. The number of edges m in $G_1(\Gamma)$ is given by

$$m = \sum_{i=1}^{n} \sum_{k=1}^{s} (4k-1) = n\left[\frac{4s(s+1)}{2} - s\right] = ns(2s+1)$$

Theorem 4. The Frucht graph G_1 is not hyperenergetic.

Proof. We observe that,

$$m = ns(2s+1) < 2\left[n(s+1)(2s+)\right] - 2 = 2p-2.$$

Thus the result follows from Theorem 1.

Lovasz [10] gives an alternate construction of the

graph $G_2(\Gamma)$ used to prove the Fruchts theorem. In this case the figure of color k is a path of length $k+2$, including the end vertices g_i and g_j, in which to the first k internal vertices are attached a path P_2 and to the last internal vertex (near g_j) is attached a path P_3.

Theorem 5. $G_2(\Gamma)$ has $n(s^2+4s+1)$ vertices, where $|\Lambda| = s$.

Proof. In $G_2(\Gamma)$ each arc $g_i g_j$ is replaced by a figure with $k+1+2k+3=3k+4$ internal vertices(that is excluding g_i and g_j) if $g_i^{-1} g_j = g_k$.

Therefore total number of extra vertices introduced to form $G_2(\Gamma)$ is equal to

$$\sum_{i=1}^{n}\sum_{k=1}^{s}(2k+3) = \sum_{i=1}^{n}(s^2+4s) = n(s^2+4s)$$

Hence $|V(G_2(\Gamma))| = n(s^2+4s)+n = n(s^2+4s+1)$.

Theorem 6. $G_2(\Gamma)$ has $n(s^2+5s)$ edges, where $|\Lambda| = s$.

Proof. The number of edges in $G_2(\Gamma)$ is given by

$$m = \sum_{i=1}^{n}\sum_{k=1}^{s}(k+2+k+2) = \sum_{i=1}^{n}(s^2+5s) = n(s^2+5s).$$

Theorem 7. $G_2(\Gamma)$ is not hyperenergetic.

Proof. We see that

$$m = n(s^2+5s) < 2\left[n(s^2+4s+1)\right] - 2 = 2p-2.$$

Thus the result follows from Theorem 1.

Combining the above observations, we conclude with the following result.

Theorem 8. Every group is isomorphic to the automorphism group of some non-hyperenergetic graph.

REFERENCES

[1] I. Gutman, "The energy of a graph," Journal of the Serbian Chemical Society, Vol. 64, pp. 199–205, 1999.

[2] L. Lovasz, "Combinatorial problems and exercises," North-Holland, Amsterdam, 1979.

[3] I. Gutman and L. Pavlovic, "The energy of some graphs with large number of edges," Bull. Acad. Serbe Sci. Arts, Vol. 118, pp. 35–50, 1999.

[4] I. Gutman Y. Hou, H. B. Walikar, H. S. Ramane, and P. R. Hamphiholi, "No huckel graph is hyperenergetic," Journal of the Serbian Chemical Society, Vol. 65, No. 11, pp. 799–801, 2000.

[5] I. Gutman, "The energy of a graph, old and new results," In: A. Betten, *et al*., Algebraic Combinatorics and its Applications, Springer-Verlag, Berlin, pp.196–211, 2001.

[6] I. Gutman and L. Pavlovic, "The energy of some graphs with large number of edges," Bull. Acad. Serbe Sci. Arts, Vol. 118, pp. 35–50, 1999.

[7] R. Frucht, "Herstellung von graphin mit vorgege bener abstakten Gruppe," Compositio Mathematica, Vol. 6, pp. 239–250, 1938.

[8] R. Frucht, "Graphs of degree three with a given abstract group," Canadian Journal Mathematics, Vol. 1, pp. 365–378, 1949.

[9] R. Frucht and F. Harary, "On the corona of two graphs," Aequationes mathematicae, Basel, Vol. 4, pp. 322–325, 1970.

[10] J. Koolen, V. Moulton, I. Gutman, and D. Vidovic, "More hyperenergetic molecular graphs," Journal of the Serbian Chemical Society, Vol. 65, pp. 571–575, 2000.

[11] H. B. Walikar, H. S. Ramane, and P. R. Hamphiholi, "On the energy of a graph," Proceedings of Conference on Graph Connections (R. Balakrishnan *et al*. eds.), Allied Publishers, New Delhi, pp. 120–123, 1999.

Intelligent Information Management, **2009, 1, 122-127**
doi:10.4236/iim.2009.12018 Published Online November 2009 (http://www.scirp.org/journal/iim)

Scientific Research

Word Sense Disambiguation in Information Retrieval

Francis de la C. Fernández REYES, Exiquio C. Pérez LEYVA, Rogelio Lau FERNÁNDEZ
Instituto Superior Politécnico, José Antonio Echeverría, Marianao, Cuba
Email:{*ffernandez, exiquio, lau*}*@ceis.cujae.edu.cu*

Abstract: The natural language processing has a set of phases that evolves from lexical text analysis to the pragmatic one in which the author's intentions are shown. The ambiguity problem appears in all of these tasks. Previous works tries to do word sense disambiguation, the process of assign a sense to a word inside a specific context, creating algorithms under a supervised or unsupervised approach, which means that those algorithms use or not an external lexical resource. This paper presents an approximated approach that combines not supervised algorithms by the use of a classifiers set, the result will be a learning algorithm based on unsupervised methods for word sense disambiguation process. It begins with an introduction to word sense disambiguation concepts and then analyzes some unsupervised algorithms in order to extract the best of them, and combines them under a supervised approach making use of some classifiers.

Keywords: disambiguation algorithms, natural language processing, word sense disambiguation

1. Introduction

The natural language processing involves a set of tasks and phases that evolves from the lexical text analysis to the pragmatic one in which the author's intentions are shown. One natural language problem is ambiguity, as we can see in the following sentence: "I made her duck". This is a classical example of ambiguity; someone who hears this phrase understands the speaker's intention, but it is harder make the computer understands it. First, the words duck and her are morphologically or syntactically ambiguous in their part-of-speech. Duck can be a verb or a noun, while her can be an object pronoun or a possessive adjective. Second, the word make is semantically ambiguous; it can mean create or cook. Finally, the verb make is syntactically ambiguous in a different way. Make can be transitive, that is, taking a single direct object, or it can be intransitive, that is, taking two objects, meaning that the first object (her) got made into the second object (duck). Finally, make can take a direct object and a verb, meaning that the object (her) got caused to perform the verbal action (duck) [1].

There are already a lot of works that resolve, almost complete, the lexical ambiguity, as part of syntactic analysis in natural language processing. However, a current problem, without a complete solution yet, is semantically ambiguity, according to Alexander Gelbukh and Grigori Sidorov [2]. Nowadays, algorithms that attempt resolving this problem are divided into two groups: discriminative algorithms and disambiguation algorithms.

One of the natural language processing applications is the information retrieval. On the one hand, Internet and digital libraries have a huge amount of knowledge that can answer a lot of questions that people may have. On the other hand, the amount of information is so huge that interferes in its proficiency because it is impossible to process it easily. At present, more used techniques for information retrieval implicate the search of keywords: Files that contain the words that the user indicates are being found. Another idea to set relevant knowledge, in front of a question, will be attending to synonym relations to establish similar documents; besides, the use of relations between words under a specific context create the necessity of employ a word sense disambiguation algorithm to understand the sense of the words that user is using in his search.

This paper presents an analysis of the existing disambiguation algorithms and it analyses the quality of each of them taking into account the metrics that have been establish for the evaluation. At the same time, it shows a possible combination of features and classifiers to propose a word sense disambiguation algorithm that resolves some deficiencies detected before and improve the evaluation parameters.

2. Word Sense Disambiguation Algorithms

The word-sense disambiguation process consists of assigning to each given word in a context, one definition or meaning (predefine sense or not), that is distinguishable from others that it can have.

The disambiguation techniques may be classified into

the way that is shown in Figure 1.

In the case of discrimination algorithms, a meaning for the word is not enough, it is necessary to determine which occurrences have the same sense, without the need of establish which it is. Besides, they work with no linguistic resources. On the contrary, disambiguation algorithms reach the meaning of the word using external linguistic resources (corpus or knowledge bases). The discrimination algorithms identify context vectors for all given word occurrences, divide the vectors into groups and interpret each of them as a sense.

Corpus based methods (supervised) collect a set of examples, manually tagged, for each sense of disambiguation words and induce a classifier (Support Vector Machine, Näive Bayes [3] from these examples, then the disambiguation is reduced to the process of classifying the word in one of his possible senses. Among the limitations you can find in those methods we have the knowledge domain dependency (due to the example set use) and the manual tagging is extremely expensive.

Knowledge based methods (not supervised) do not require tagged corpus, rather use external linguistic resources, and therefore, the disambiguation can be considered at any knowledge domain, as long as that external resource accept it. The supervised methods obtain better results than the not supervised ones, but they prefer the seconds because they are not restricted to a specific knowledge domain. Knowledge based methods can make use of dictionaries as Lesk's algorithm [4], of thesaurus as Yarowsky's [5] and of WordNet as Resnik's [6].

As not supervised algorithms can be applied to any knowledge domain, as long as that external resource allows it, the authors analyze just them in the rest of the document. Specifically we will examine the following methods: based on grouping, Fuzzy Borda Voting, Extended Lesk, Conceptual density and Sense probability. There are others algorithms that also result interesting, but we don't analyze them in the present paper, examples of these are: Meaning affinity model [7], using automatically acquired predominant senses [8] and based on lexical cohesion [9].

2.1. Using Sense Clustering for Word Sense Disambiguation

This method doesn't require the use of a training set, just use WordNet. The algorithm begins to grouping all senses of the disambiguation target words. This process tries to identify cohesive senses' groups; those are created by means of Extended Star cluster algorithm and β_0-similarity graph between different senses [10]. Then, the method filters the groups to select those that match the best with the context. If the selected groups disambiguate all the words (each group show only one sense), then the process stops and the senses belonging to the selected groups are interpreted as the disambiguated ones. Otherwise, the clustering and filtering steps are performed again (regarding the remaining senses) until the disambiguation is achieved or when it is impossible to raise β_0 threshold [10].

The algorithm input is the disambiguation target words set *W* and the context represented as topic signature *T* [10].

There are two sub process cluster and filter. The first one is carried out by the Extended Star Clustering Algorithm, which builds star-shaped and overlapped clusters. Each cluster consists of a star and its satellites, where the star is the sense with the highest connectivity of the cluster, and the satellites are those senses connected with the star. The connectivity is defined in terms of the β_0-similarity graph, which is obtained using the cosine similarity measure between topic signatures and the minimum similarity threshold β_0.

Once clustering is performed over the senses of words in W, a set of sense clusters is obtained. As some clusters can be more appropriate to describe the semantics of W than others, they are ranked according to a textual measure context T.

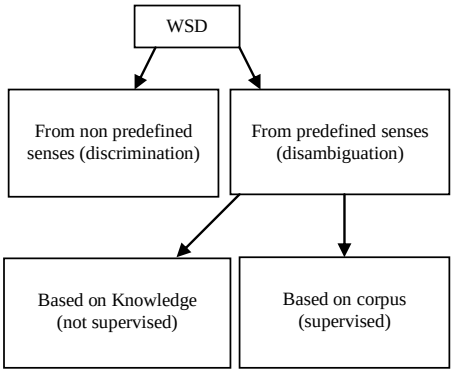


Figure 1. Classification of word sense disambiguation algorithms

After grouping the algorithm is obtained a set S with the senses of the selected group. Then, the method compares if the sense quantity in S matches the word quantity in W, if this is true, the disambiguation process stops, on the contrary, the cluster algorithm starts again with other threshold β_0.

The algorithm behaves adequately in front of nouns instances, although it shows an inconvenience, it does not disambiguate proper nouns due to the lack of senses in WordNet. Although, it behaves in a non correctly way for verbs, of 304 tested, the algorithm just disambiguated the 30% in a correct way. This result is a consequence of the high grade of polysemy that verbs had and the few relations that appear between them in WordNet, therefore, the cluster with high polysemy level can not create cohesive groups and fail the disambiguation process. Besides, the context is reduced to the sentence, conesquently, become interesting extending the context to paragraph, this approach includes more verbs and the search of WordNet relations increases, the effect is the improvement of the clustering process.

2.2. Combining Different Methods by Means of Fuzzy Borda Voting

The original scheme, Borda voting score, was introduced in 1770 by the mathematician Jean Charles de Borda of the French Academy. This method consists of a pondering system, that fight the general believes that the candidate which obtains the majority of votes is the one that voters prefer and show examples of contradictions in votes' system used before.

Borda voting score establishes a punctuation system where each voter gives a mark to each one of the candidates, following the order of their preferences. In order to find the winner, add the obtained scores for each candidate. This system has an inconvenience it does not perform rules for the weights of the candidates, which could imply an arbitrary assignment that changes the result.

The fuzzy variant allows the experts (voters on the original scheme) gives a numerical value that indicates how some alternatives (candidates on the original scheme) are preferred among others, evaluating the preferences in a range between 0 and 1. In Rosso & Buscaldi [11], each expert gives a mark to each alternative, according to the number of alternatives worse than it. The algorithm establishes the use of experts (other disambiguation methods) to achieve the disambiguation process, and the ones considered are: Sense probability, Extended Lesk, Conceptual density (for verbs) and WordNet domains.

To obtain fuzzy preferences relations, the output weights $w_1, w_2, ..., w_n$ of each expert *k* are transformed to fuzzy confidence values by means of the following transformation [11]:

$$r_{ij}^k = \frac{w_i}{w_i + w_j}$$

where $r^k{}_{ij}$ is considered as the degree of confidence with which the expert k prefers alternative x_i to x_j.

With the fuzzy preferences relations of m experts over n alternatives x1, x2, . . ., xn. For each expert k we obtain a matrix of preference intensities [11]:

$$\begin{pmatrix} r_{11}^k & r_{12}^k & \cdots & r_{1n}^k \\ r_{21}^k & r_{22}^k & \cdots & r_{21}^k \\ \cdots & \cdots & \cdots & \cdots \\ r_{n1}^k & r_{n2}^k & \cdots & r_{nn}^k \end{pmatrix}$$

The final value assigned by the expert k to each alternative xi coincides with the sum of the entries greater than 0.5 in the i-th row in the preference matrix.

Therefore, the definitive fuzzy Borda count for an alternative xi is obtained as the sum of the values assigned by each expert *k* [11]:

$$r(x_i) = \sum_{k=1}^{m} rk(x_i)$$

In order to apply this system on word sense disambiguation the first thing to do is determine the target word and the senses set of it, afterward, using an expert (for example, "Sense probability"), calculate the weights of each sense and begin the competition between them. Establish the confidence degree $r^k{}_{ij}$ of the first sense with the second one, then with the third one, and so on in order to make the competition. These fuzzy values are part of the first row of the preferences matrix, each row of the matrix match an analyzed sense. In this way are go establishing the different confidence degrees with the other senses till complete the matrix. To end with the expert, establish the value assigned to the sense considered, this value is calculated adding the row that matches with the alternatives proposed, but avoiding the value founded in matrix principal diagonal because it is the alternative analyzed with it self. Then proceed in same way with the other experts, at the end add, for each sense, the values that each one of them assigned and the biggest score is the winner, considering this one as the correct sense of the target word.

The algorithm has as principal idea the usage of a voting system with fuzzy bases where each expert is word sense disambiguation algorithm knowledge based, these is a positive issue, the approach put under competition various algorithms and the correct sense is that one who receive the biggest score. The ambiguity resolution for verbs behaves adequately, the experts that are considered work in the sentence context. Will be interesting find ano-

ther set of experts, this selection should be doing according to the algorithm results taking into account the part of speech and considering also a baseline as expert, then realize tests changing the competition scheme to analyze the behavior of disambiguation process.

2.3. Sense Probability (Most Frequently Sense)

In nineties grow up an interest on the line of the establishment of methods for automatic word sense disambiguation evaluation that offers a reference point to measure the quality of done work. A starter point is the set of a superior and inferior limit for the disambiguation results: the inferior limit match the choice of the most frequently sense, while the superior limit match the human tagged. The authors asseverate as correct choice the most frequently sense in a 75% of the cases. About superior limit, human tagged, exist many controversies about his reliability, fed by contradictory results obtained in several experiments. Other contribution is the morphsyntactic and semantic tagged words of open class from Brown corpus, making benchmarks for disambiguation systems in senses choosing: the chance, the frequency (most frequency sense), the concurrence [12].

Those basic disambiguation techniques, that usually not imply any kind of linguistic knowledge, are used as a reference point for the evaluation of those systems [12].

One of the baseline measure used is Sense probability. It consists in assign the first sense of WordNet to each target word and it is calculated as follow:

$$BL_{MFS} = \frac{1}{|T|}\sum_{i=1}^{|T|}\delta(w_i,k)$$

where $\delta(w_i,k)$ is equal to 1 when the k-th sense of the word w_i belongs to the group manually tagged by the lexicographer (example, SemCor) for word w_i and it is 0 otherwise. The Most Frequency Sense (MFS) calculations are based on the frequencies of SemCor corpus terms. T is the test corpus where are contained the instances w_i. WordNet rank the senses taking into account the appearing frequencies of them inside SemCor corpus. This baseline always resolves the semantic ambiguity problem with a 78% precision [12].

This technique lack totally of linguistic information, however, is a good idea has a tagged corpus to find the most frequently sense. Actually, it is not a knowledge based disambiguation algorithm but it is used as a reference point to compare the algorithm done.

2.4. Extended Lesk Algorithm

This algorithm is an improvement version (WordNet based) of the well known Lesk procedure and based it on the use of dictionaries. The original algorithm was supported in the comparison of the gloss target word with the context words and their gloss. The improvement consists in taking into account also the gloss of the concepts related to the target word, by means of various WordNet relations [13].

Then, the similarity between a word sense and his context is calculated by overlapping. Overlapping is the intersection between to synsets set (a synset is a number that implies sense for WordNet), on one hand, the ones for the target word and on the other hand the gloss of the context synsets. To the target word is assigned the sense obtained that betters overlap with the gloss of the context words and their related synsets [13].

The heterogeneous approach of the Extended Lesk algorithm show better results than the homogeneous ones, consider the part of speech not improve the precision, excluding the case of adjectives. The context of this algorithm is a window of size 7, 9 or 11 around the target word, does not take into account the sentence context and it uses various WordNet relations and glosses. The size of the window depends directly from the part of the speech, while biggest is the window more decrease the polysemy and reach better precision results, most of all in verbs.

2.5. Conceptual Density Algorithm

Conceptual density was originally introduced by Agirre and Rigau in 1996 above WordNet. It is calculated over sub-graphs of this lexical database, determined by hypernymy relations. The proposed formula gives a measure of the conceptual density between the word and their senses. Besides, this formula has the following characteristics [14]:

- ✓ The length of the shortest path that connects the concepts involved.
- ✓ The independent measure of the number of concepts we are measuring.
- ✓ The depth in the hierarchy: concepts in a deeper part of the hierarchy should be ranker closer.

Given a concept *c*, at the top of a sub-hierarchy, and given *nhyp* and *h* (mean number of hyponyms per node and height of the sub-hierarchy, respectively), the Conceptual Density for *c* when its sub-hierarchy contains a number *m* (marks) of senses of the words to disambiguate is given by the formula below [14]:

$$CD(c,m) = \frac{\sum_{i=0}^{m-1} nhyp^{i^{0.20}}}{descendientes_c}$$

Rosso and Buscaldi [11] transform this conception of Agirre and Rigau to use the formula as an expert in the Fuzzy counting Borda system. The formulation of Conceptual Density for a sub-graph S of WordNet is defined by the following formula:

$$CD(m, f, n) = m^{\alpha}(\frac{m}{n})$$

where m is the relevant synsets in the sub-graph and n is the total number of them. The constant α takes values around 0.1, and this value was obtained through experimental results from Rosso and colleagues work [15]. The argument value f is obtained from the frequency of the synsets related in the sub-graph, which appears in WordNet. The relevant synsets match with the ones of the target word and the ones of the context words. At Buscaldi and Rosso [11] the expert that perform over Conceptual Density use as context two nouns for the word sense disambiguation process. The weights that are calculated by the previous formula are used to compute the confidence values used for fill the preference matrix. It proposes, besides, a second expert that exploits the holonymy relations instead of hypernymy. This expert uses as context all nouns in the sentence where appears the target word [11].

Buscaldi and Rosso proposition reach good precision (around 75%) over nouns and use a window of two nouns. For verbs does not behaves in a adequately way, taking into account Banerjee analysis [13] we can corroborate that to resolve correctly the verb ambiguity it is necessary use windows of at least 11 context words and besides including the major quantity of linguistic information.

3. Comparison of the Word Sense Disambiguation Algorithms

As it is known, the metrics that are used to evaluate a disambiguation algorithm are the following [12]:

Precision= # correctly disambiguated words / # disambiguated words

Recall = # correctly disambiguated words / # tested set words

Coverage = # disambiguated words / # tested set words

The combination of precision and recall it is known as F1 measure and it is calculated by the following formula [12]:

F1 = 2 * precision * recall / (precision + recall)

Table 1 shows a comparison between the not supervised word sense disambiguation algorithms according to the previous metrics [12]:

Even when Sense probability, using always the most frequently sense, is the one that occupies the first place, has a limitation, do not take into account linguistic information. Therefore, the best systems are those that use clustering and Fuzzy Borda counting. It is part of supposing then that the combination of them offer better results.

4. Word Sense Disambiguation Algorithm Propose

Supervised methods offers better solution than not supervised ones, however, the first ones need a training tagged set, this implies human factor for tagging and lacks a representative corpus for Spanish. Not supervised methods, on the other hand, do not require a training corpus, they use eternal resources as lexical data bases, thesaurus, and dictionaries, those are available in Internet and they abound most of all for the English language.

There are propositions that try to combine not supervised methods by a voting system [11,15] obtaining good results. It would be interesting combine not supervised methods under a supervised approach, that means, make a proposition where the first step is the selection of algorithm set taking into account their results in the word sense disambiguation process and then apply to them a classifier set to learn which method has to use under a determined circumstance. Previously we analyze some algorithms propositions that show satisfactory results and we conclude that ambiguity on verbs is harder to result, that is why it makes necessary to include as much linguistic information as it is possible and enlarge the sentence context to paragraph context.

The method we propose is based on the combination of various not supervised algorithms and baselines. It creates a feature vector (some features are obtained from WordNet) for each sense of the target word. Each algorithm is a feature for each sense, the output of the algorithm will be normalized in a measure between 0 and 1, then it combines certain classifiers to search the winner sense. If no exit one, then it uses a baseline (for instance MFS) in order to search the correct sense of the word.

Table 1. Not supervised system score ranked by F1 measure. (C=Coverage, P=Precision, R=Recall, F1=F1 measure)

System	C	P	R	F1
BL_{MFS}	100.0	78.89	78.89	78.89
Fuzzy Borda	100.0	78.63	78.63	78.63
Clustering based	100.0	70.21	70.21	70.21
Conceptual density	86.2	71.2	61.4	65.94
Extended Lesk	100.0	62.4	62.4	62.4

The method defines a target word as a set of sense vectors. Each vector contains in the beginning all kind of linguistic information and the normalized result of applying some algorithms (the first experiments will use Extended Lesk, Conceptual Density and Clustered based algorithms). Then the vectors are reduced by use of feature selection eliminating linguistic redundancies among vectors. To find the winner sense we suggest the use of the combination of algorithms results that we found on each feature vector, then apply a classification technique such as Support Vector Machines or Näives Bayes which performs good over two class, this classes proposed are best and bad senses. We hope this method increase the precision in word sense disambiguation process.

5. Conclusions

The supervised methods offer better solutions than not supervised ones, but they do not make use of external resources, that is why they are applied on specific domains, using language characteristics and syntactic resolution, making them language dependents.

The algorithm proposed tries to combine positive features of the not supervised method, establishing a classifier system to determine which of them is better, and taking into account the winner algorithm, the proposition output the correct sense of the word. Even when this method is in a development phase and determination of the classifiers to use, in order to validate it in some task of the SemEval 2007 competition, empirically is well suppose that this proposition shows better results taking into account the enlargement of the considered linguistic information in the algorithm previously analyzed and they are combined under a automatic learning approach.

REFERENCES

[1] E. Agirre and G. Rigau, "Word sense disambiguation using conceptual density," International Conference on Computational Linguistics (COLING), Copenhagen, Denmark 1996.

[2] H. Anaya-Sánchez, A. Pons-Porrata, *et al.*, "TKB-UO: Using sense clustering for WSD," 4th International Workshop on Semantic Evaluations (SemEval), Prague, Czech Republic, Association for Computational Linguistics, 2007.

[3] S. Banerjee, "Adapting the lesk algorithm for word sense disambiguation using wordnet," Department of Computing Science. Minnesota, USA, University of Minnesota, MSc.: 98, 2002.

[4] D. Buscaldi and P. Rosso, "UPV-WSD: Combining different WSD methods by means of fuzzy borda voting," 4th International Workshop on Semantic Evaluations (SemEval), Prague, Czech Republic, Association for Computational Linguistics, 2007.

[5] Y. Chali and S. R. Joty, "UofL: Word dense disambiguation using lexical cohesion," 4th International Workshop on Semantic Evaluations (SemEval), Prague, Czech Republic, Association for Computational Linguistics, 2007.

[6] A. Gelbukh and G. Sidorov, "Procesamiento automático del español con enfoque en recursos léxicos grandes," México, Centro de Investigación en computación, Instituto Politécnico Nacional, 2006.

[7] J. Huang, J. Lu, *et al.*, "Comparing naive Bayes, decision trees, and SVM with AUC and accuracy," Third IEEE International Conference on Data Mining, 2003.

[8] R. Ion and D. Tufis, "RACAI: Meaning affinity models," 4th International Workshop on Semantic Evaluations (SemEval), Prague, Czech Republic, Association for Computational Linguistics, 2007.

[9] D. Jurafsky and J. H. Martin, "Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition," Prentice Hall, 2000.

[10] R. Koeling and D. McCarthy, "Sussx: WSD using automatically acquired predominant senses," 4th International Workshop on Semantic Evaluations (SemEval), Prague, Czech Republic, Association for Computational Linguistics, 2007.

[11] M. Lesk, "Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone," 5th Annual International Conference on Systems Documentation, Ontario, Canada, ACM, 1986.

[12] I. Nica, "El conocimiento lingüístico en la desambiguación semántica automática," España, 2006.

[13] P. Resnik, "Disambiguating noun groupings with respect to wordnet senses," Third Workshop on Very Large Corpora, Massachusetts Institute of Technology Cambridge, Massachusetts, USA, 1995.

[14] D. Yarowsky, "Word-sense disambiguation using statistical models of roget's categories trained on large corpora," 15th International Conference on Computational Linguistics (COLING), Nantes, France, Association for Computational Linguistics, 1992.

[15] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," 33th Annual Meeting of the Association for Computational Linguistics (ACL'95), Cambridge, Massachussets, 1995.

Intelligent Information Management, **2009, 1, 128-138**
doi:10.4236/iim.2009.12019 Published Online November 2009 (http://www.scirp.org/journal/iim)

Scientific Research

Evolutionary Algorithm for Extractive Text Summarization

Rasim ALGULIEV, Ramiz ALIGULIYEV
Institute of Information Technology, Azerbaijan National Academy of Sciences, Baku, Azerbaijan
Email: *rasim@science.az, a.ramiz@science.az*

Abstract: Text summarization is the process of automatically creating a compressed version of a given document preserving its information content. There are two types of summarization: extractive and abstractive. Extractive summarization methods simplify the problem of summarization into the problem of selecting a representative subset of the sentences in the original documents. Abstractive summarization may compose novel sentences, unseen in the original sources. In our study we focus on sentence based extractive document summarization. The extractive summarization systems are typically based on techniques for sentence extraction and aim to cover the set of sentences that are most important for the overall understanding of a given document. In this paper, we propose unsupervised document summarization method that creates the summary by clustering and extracting sentences from the original document. For this purpose new criterion functions for sentence clustering have been proposed. Similarity measures play an increasingly important role in document clustering. Here we've also developed a discrete differential evolution algorithm to optimize the criterion functions. The experimental results show that our suggested approach can improve the performance compared to sate-of-the-art summarization approaches.

Keywords: sentence clustering, document summarization, discrete differential evolution algorithm

1. Introduction

Text summarization is the process of automatically creating a compressed version of a given document preserving its information content. Automatic document summarization is an important research area in natural language processing (NLP). The technology of automatic document summarization is developing and may provide a solution to the information overload problem [1–3].

The process of text summarization can be decomposed into three phases: analysis, transformation, and synthesis. The analysis phase analyzes the input text and selects a few salient features. The transformation phase transforms the results of the analysis into a summary representation. Finally, the synthesis phase takes the summary representation, and produces an appropriate summary corresponding to users' needs. In the overall process, compression rate, which is defined as the ratio between the length of the summary and that of the original, is an important factor that influences the quality of the summary. As the compression rate decreases, the summary will be more concise; however, more information is lost. While the compression rate increases, the summary will be larger; relatively, more insignificant information is contained. In fact, when the compression rate is 5–30%, the quality of the summary is acceptable [1–4].

Text summarization can be categorized into two approaches: extractive and abstractive. Extractive summarization methods simplify the problem of summarization into the problem of selecting a representative subset of the sentences in the original documents. Abstractive summarization may compose novel sentences, unseen in the original sources [1]. However, abstractive approaches require deep NLP such as semantic representation, inference and natural language generation, which have yet to reach a mature stage nowadays [5].

Extractive summarization systems are commonly used in automatic summarization to produce extractive summaries. Systems for extractive summarization are typically based on technique for sentence extraction, and attempt to identify the set of sentences that are most important for the overall understanding of a given document. Most commonly, such systems use some kind of similarity or centrality metric to identify the set of sentences to include in the summary [1,4,6–15]. For example, in [14] a paragraph extraction from a document based on intra-document links between paragraphs is proposed. It yields a TRM (Text Relationship Map) from intra-links, which indicate that the linked texts are semantically related. It proposes four strategies from the TRM: bushy path, depth-first path, segmented bushy path, augmented segmented bushy path. An improved version of this approach is proposed in [1,4,6].

In this paper we demonstrate an extractive text sum-

marization method which is based on sentence clustering. For this purpose new criterion functions for sentence clustering have been offered. In our study we developed a discrete differential evolution algorithm to optimize the criterion functions. The experimental results on an open benchmark datasets from DUC2001 and DUC2002 show that our suggested approach can improve the performance compared to state-of-the-art summarization approaches.

The rest of this paper is organized as follows: Section 2 introduces related works. The proposed sentence clustering based approach for generic single-document summarization is presented in Section 3. The discrete differential evolution algorithm for optimization procedure is given in Section 4. The experiments and results are given in Section 5. Finally, we conclude our paper in Section 6.

2. Related Work

Automatic document summarization has been actively investigated in recent years, and most researchers have concentrated on the extractive summarization method, but not the abstractive summarization method – see for example, the 6th issue of the Information Processing and Management: an International Journal of 2007. The current paper contains four references to the articles published in this edition [5,16–18]. A comprehensive survey of document summarization can be found in [17].

The centroid-based method [19,13] is one of the most popular extractive summarization methods. MEAD (http://www.summarization.com/mead/) is an implementation of the centroid-based method for either single- or multi-document summarizing. It is based on sentence extraction. For each sentence in a cluster of related documents, MEAD computes three features and uses a linear combination of the three to determine what sentences are most salient. The three features used are centroid score, position, and overlap with first sentence (which may happen to be the title of a document). For single documents or (given) clusters it computes centroid topic characterizations using tf-idf-type data. It ranks candidate summary sentences by combining sentence scores against centroid, text position value, and tf-idf title/lead overlap. Sentence selection is constrained by a summary length threshold, and redundant new sentences avoided by checking cosine similarity against prior ones [20]. In [21] each document is considered as a sequence of sentences and the objective of extractive summarization is to label the sentences in the sequence with 1 and 0, where a label of 1 indicates that a sentence is a summary sentence while 0 denotes a non-summary sentence. To accomplish this task, a conditional random field is applied [22]. A novel extractive approach based on manifold-ranking of sentences to query-based multi-document summarization proposed in [23]. This approach first uses the manifold-ranking process to compute the manifold-ranking score for each sentence that denotes the biased information richness of the sentence, and then uses greedy algorithm to penalize the sentences with highest overall scores, which are considered both informative and novel, and highly biased to the given query.

The summarization techniques can be classified into two groups: supervised techniques, that rely on machine learning algorithms trained on pre-existing document-summary pairs, and unsupervised techniques, based on properties and heuristics derived from the text. Supervised extractive summarization techniques [1,4–6,21, 24–26] treat the summarization task as a two-class classification problem at the sentence level, where the summary sentences are positive samples while the non-summary sentences are negative samples. After representing each sentence by a vector of features, the classification function can be trained in two different manners [21]. The first is in a discriminative way with well-known algorithms such as SVM (Support Vector Machine) [4]. In [1], the use of genetic algorithm (GA), mathematical regression (MR), feed forward neural network (FFNN), probabilistic neural network (PNN) and Gaussian mixture model (GMM) for automatic text summarization task have been investigated. This approach is a trainable summarizer, which takes into account several features, including sentence position, positive keyword, negative keyword, sentence centrality, sentence resemblance to the title, sentence inclusion of name entity, sentence inclusion of numerical data, sentence relative length, bushy path of the sentence and aggregated similarity for each sentence to generate summaries. The article [25] presents a multi-document, multilingual, theme-based summarization system based on modeling text cohesion (story flow). Many unsupervised methods have been developed for document summarization by exploiting different features and relationships of the sentences, such as clustering of sentences [7–11], the hidden topics in the documents [12], graphs based on the similarity of sentences[13,19,27].

Recently, graph-based methods have been offered to rank sentences. Lexrank [19] and [27] are two such systems using the algorithms PageRank and HITS to compute sentence importance. Lexrank is used to compute sentence importance based on the concept of eigenvector centrality in a graph representation of sentences for multi-document summarization task. The graph-based extractive summarization algorithms succeed in identifying the most important sentences in a text based on information exclusively drawn from the text itself. Unlike other systems, which attempt to find out what makes a good summary by training on collections of summaries built for other articles, the graph-based methods are fully unsupervised, and rely on the given texts to derive an extractive summary[23].

On the other hand, summarization task can also be categorized as either generic or query-based. A query-based summary presents the information that is most relevant to the given queries [16,23,24,28,29] while a generic summary gives an overall sense of the document's content [7–14,19,27,30]. The QCS system (Query, Cluster, and Summarize)[16] performs the following tasks in response to a query: retrieves relevant documents; separates the retrieved documents into clusters by topic, and creates a summary for each cluster. QCS is a tool for document retrieval that presents results in a format so that a user can quickly identify a set of documents of interest. In [29] are developed a generic, a query-based, and a hybrid summarizer, each with differing amounts of document context. The generic summarizer used a blend of discourse information and information obtained through traditional surface-level analysis. The query-based summarizer used only query-term information, and the hybrid summarizer used some discourse information along with query-term information.

Automatic document summarization is a highly interdisciplinary research area related with computer science, multimedia, statistics, as well as cognitive psychology. In [31] is introduced an intelligent system, the event indexing and summarization (EIS) system, for automatic document summarization, which is based on a cognitive psychology model (the event-indexing model) and the roles and importance of sentences and their syntax in document understanding. The EIS system involves syntactic analysis of sentences, clustering and indexing sentences with five indices from the event-indexing model, and extracting the most prominent content by lexical analysis at phrase and clause levels.

3. Sentence Clustering

Clustering is the process of discovering natural groupings or clusters and identifying interesting distributions and patterns within multidimensional data based on some similarity measure. The topic of clustering has been extensively studied in many scientific disciplines such as text mining, pattern recognition, IR etc. Document clustering is a central problem in text mining which can be defined as grouping documents into clusters according to their topics or main contents. Document clustering has many purposes including expanding a search space, generating a summary, automatic topic extraction, browsing document collections, organizing information in digital libraries and detecting topics. In the literature a wide variety of clustering algorithms have been proposed for different applications and sizes of data sets. The surveys on the topics [32–35] offer a comprehensive summary of the different applications and algorithms.

Generally clustering problems are determined by four basic components [36,37]: 1) the (physical) representation of the given data set; 2) the distance/dissimilarity measures between data points; 3) the criterion/objective function which the clustering solutions should aim to optimize; and, 4) the optimization procedure. For a given data clustering problem, the four components are tightly coupled. Various methods/criteria have been proposed over the years from various perspectives and with various focuses.

3.1. Sentence Similarity Measure Based on Terms Co-Occurrence

Let a document D is decomposed into a set of sentences $D=\{S_1,S_2,...,S_n\}$, where n is the number of sentences. Let $T=\{t_1,t_2,...,t_m\}$ represents all the distinct words (terms) occurring in a document D, where m is the number of words. In most existing document clustering algorithms, documents are represented using the vector space model (VSM) [33]. Each document is represented using these words as a vector in m-dimensional space. A major characteristic of this representation is the high dimensionality of the feature space, which imposes a big challenge to the performance of clustering algorithms. They could not work efficiently in high-dimensional feature spaces due to the inherent sparseness of the data [17]. The vector dimension *m* is very large compared to the number of words in a sentence, thus the resulting vectors would have many null components [38]. In our method, a sentence S_i is represented as a set of distinct terms appearing in it, $S_i=\{t_1,t_2,...,t_{m_i}\}$, where m_i is the number of distinct terms in the sentence S_i.

Similarity measures play an increasingly important role in NLP and IR. Similarity measures have been used in text-related research and application such as text mining, information retrieving, text summarization, and text clustering. These applications show that the computation of sentence similarity has become a generic component for the research community involved in knowledge representation and discovery. There are more papers on similarity between documents than between sentences or short texts, but not few [38,39]. The paper [39] presents a method for measuring the similarity between sentences or very short texts, based on semantic and word order information. First, semantic similarity is derived from a lexical knowledge base and a corpus. Second, the proposed method considers the impact of word order on the sentence meaning. The overall sentence similarity is defined as a combination of semantic similarity and word order similarity. Liu *et al*. [40] present a novel method to measure similarity between sentences by analyzing parts of speech and using Dynamic Time Warping technique. In [41] proposed a novel measure based on the earth

mover's distance (EMD) to evaluate document similarity by allowing many-to-many matching between subtopics. First, each document is decomposed into a set of subtopics, and then the EMD is employed to evaluate the similarity between two sets of subtopics for two documents by solving the transportation problem. The proposed measure is an improvement of the previous optimal matching (OM)-based measure, which allows only one-to-one matching between subtopics. The study of semantic similarity between words has long been an integral part of IR and NLP [42]. The method, proposed in [42], integrates both page counts and snippets to measure semantic similarity between a given pair of words. In this paper modified four popular co-occurrence measures; Jaccard, Overlap (Simpson), Dice and PMI (Point-wise Mutual Information), to compute semantic similarity using page counts.

In this section we present a method to measure similarity between sentences using the Normalized Google Distance (NGD) [43]. First we calculate a similarity measure between the terms before defining the similarity measure between the sentences. Using the NGD [43] the similarity measure between terms t_k and t_l we define as:

$$sim_{\mathrm{NGD}}(t_k, t_l) = \exp(-\mathrm{NGD}(t_k, t_l)) \quad (1)$$

where

$$\mathrm{NGD}(t_k, t_l) = \frac{\max\{\log(f_k), \log(f_l)\} - \log(f_{kl})}{\log n - \min\{\log(f_k), \log(f_l)\}}, \quad (2)$$

f_k is the number of sentences containing the term t_k, f_{kl} denotes the number of sentences containing both terms t_k and t_l, n is the number of sentences in the document.

From the properties of NGD follows that [43]:

1) The range of the $diss_{\mathrm{NGD}}(t_k, t_l)$ is between 0 and 1;

- If $t_k = t_l$ or if $t_k \neq t_l$ but $f_k = f_l = f_{kl} > 0$, then $sim_{\mathrm{NGD}}(t_k, t_l) = 1$. That is, the semantics of t_k and t_l, in the Google sense is the same.
- If $f_k > 0$, $f_l > 0$ and $f_{kl} = 0$, we take $\mathrm{NGD}(t_k, t_l) = 1$, then $0 < sim_{\mathrm{NGD}}(t_k, t_l) < 1$.
- If $0 < f_{kl} < f_l < f_k < n$ and $f_k \cdot f_l > n \cdot f_{kl}$, then $0 < sim_{\mathrm{NGD}}(t_k, t_l) < 1$.

2) $sim_{\mathrm{NGD}}(t_k, t_k) = 1$ for any t_k. For every pair t_k and t_l, we have $sim_{\mathrm{NGD}}(t_k, t_l) = sim_{\mathrm{NGD}}(t_l, t_k)$: It is symmetric.

Using the formula (1) we define a similarity measure between sentences S_i and S_j as follows:

$$sim_{\mathrm{NGD}}(S_i, S_j) = \frac{\sum_{t_k \in S_i} \sum_{t_l \in S_j} sim_{\mathrm{NGD}}(t_k, t_l)}{m_i m_j} \quad (3)$$

From the properties of $sim_{\mathrm{NGD}}(t_k, t_l)$ follows that: 1) the range of the $sim_{\mathrm{NGD}}(S_i, S_j)$ is in 0 and 1; 2) $sim_{\mathrm{NGD}}(S_i, S_i) \geq 0$ for every S_i; 3) for every pair S_i and S_j $sim_{\mathrm{NGD}}(S_i, S_j) = sim_{\mathrm{NGD}}(S_j, S_i)$: it is exchangeable.

3.2. Objective Functions

Typically clustering algorithms can be categorized as agglomerative or partitional based on the underlying methodology of the algorithm, or as hierarchical or flat (non-hierarchical) based on the structure of the final solution [32–35]. A key characteristic of many partitional clustering algorithms is that they use a global criterion function whose optimization drives the entire clustering process. In recent years, it has been recognized that the partitional clustering technique is well suited for clustering a large document database due to their relatively low computational requirements [44].

Automatic clustering is a process of dividing a set of objects into unknown groups, where the best number k of groups (or clusters) is determined by the clustering algorithm. That is, objects within each group should be highly similar to each other than to objects in any other group. The automatic clustering problem can be defined as follows [32–35,45]:

The set of sentences $D = \{S_1, S_2, ..., S_n\}$ are clustered into non-overlapping groups $C = \{C_1, ..., C_k\}$, where C_k is called a cluster, k is the unknown number of clusters. The partition should possess three properties:

1) Two different clusters should have no sentences in common, i.e. $C_p \cap C_q = \varnothing$ for $\forall p \neq q \quad p, q \in \{1,2,...,k\}$;

2) Each sentence should definitely be attached to a cluster, i.e. $\bigcup_{p=1}^{k} C_p = D$;

3) Each cluster should have at least one sentence assigned, i.e. $C_p \neq \varnothing \quad \forall p \in \{1,2,...,k\}$.

Partitional clustering can be viewed as an optimization procedure that tries to create high-quality clusters according to a particular criterion function. Criterion functions used in partitional clustering reflect the underlying definition of the "goodness" of clusters. Many criterion functions have been proposed in the literature [32–35,44] to produce more balanced partitions.

We introduce a criterion function that is defined as follows:

$$F = (1 + sigm(F_1))^{F_2} \rightarrow \max \quad (4)$$

where $sigm(z)$ is a sigmoid function that maps from the real numbers into $[0,1]$, $sigm(z) = \frac{1}{1+\exp(-z)}$.

The criterion function (4) balances both intra-cluster similarity and inter-cluster dissimilarity. This function is obtained by combining two criteria:

$$F_1 = \sum_{p=1}^{k} \left|C_p\right| \sum_{S_i, S_l \in C_p} sim_{\mathrm{NGD}}(S_i, S_l) \to \max \qquad (5)$$

and

$$F_2 = \sum_{p=1}^{k-1} \frac{1}{\left|C_p\right|} \sum_{q=p+1}^{k} \frac{1}{\left|C_q\right|} \sum_{S_i \in C_p} \sum_{S_l \in C_q} sim_{\mathrm{NGD}}(S_i, S_l) \to \min \qquad (6)$$

The F_1 criterion Function (5) maximizes the average sum of the pairwise similarity between the sentences assigned to each cluster. The F_2 criterion Function (6) computes the clustering by finding a solution that separates each cluster from other clusters. It minimizes the similarity between the sentences S_i and S_l assigned to different clusters C_p and C_q, respectively.

4. Modified Discrete Differential Evolution Algorithm (MDDE)

There are many techniques that can be used to optimize the criterion Functions (4)-(6) described in the previous Section 3. The paper [17] presents an up-to-date survey on evolutionary algorithms for clustering. It tries to reflect the profile of this area by focusing more on those subjects that have been given more importance in the literature. Particularly, the paper has focused mainly on hard partitional algorithms, though overlapping (soft/ fuzzy) approaches have also been covered. An original contribution of the present paper is that it discusses key issues on the design of evolutionary algorithms for data partitioning problems, such as usually adopted representations, evolutionary operators, and fitness functions, just to mention a few. In particular, mutation and crossover operators commonly described in the literature are conceptually analyzed, giving especial emphasis to those genetic operators specifically designed for clustering problems (i.e., cluster-oriented and context-sensitive operators). In our study these criterion functions were optimized using a differential evolution (DE) [45,46]. The execution of the differential evolution is similar to other evolutionary algorithms like genetic algorithms or evolution strategies. The evolutionary algorithms differ mainly in the representation of parameters (usually binary strings are used for genetic algorithms while parameters are real-valued for evolution strategies and differential evolution) and in the evolutionary operators.

Like to other evolutionary algorithm, DE also starts with a population of N n-dimensional search variable vectors. The classical DE [45,46] is a population-based global optimization that uses a real-coded representation. In our study we use a genetic encoding that deals with discrete variables (clusters), such that each component of the chromosome takes a value between 1 and k and represents the cluster to which the sentence is assigned. Potential set of solutions to the optimization problem are represented by a population of chromosomes. The initial population of chromosomes is generated by producing the series of integer random numbers. These numbers are uniformly generated between 1 and k inclusively. Potential solutions (chromosomes) to the target problem are encoded as fixed length discrete strings, i.e., $X_r(t) = [x_{r,1}(t), x_{r,2}(t), ..., x_{r,n}(t)]$ where $x_{r,s}(t) \in \{1, 2, ..., k\}$, $r = 1, 2, ..., N$, and $s = 1, 2, ..., n$, N is the size of the population. For example, given the number of clusters $k = 4$, the number of sentences $n = 8$ and the population size $N = 3$, a population can be as: $X_1(t) = [2,1,3,4,2,1,4,3]$, $X_2(t) = [4,3,4,2,2,3,1,1]$, and $X_3(t) = [1,2,4,2,3,1,3,4]$. In this example, chromosome $X_1(t)$ represents a candidate solution where sentences S_2 and S_6 are assigned to cluster C_1; sentences S_1 and S_5 are located in cluster C_2; sentences S_3 and S_8 are assigned to cluster C_3, and S_4 and S_7 are located in cluster C_4.

For each individual vector $X_r(t)$ that belongs to the current population, DE randomly samples three other individuals $X_{r1}(t)$, $X_{r2}(t)$, and $X_{r3}(t)$ from the same generation (for mutually different $r \neq r1 \neq r2 \neq r3$). It then calculates the difference of $X_{r2}(t)$ and $X_{r3}(t)$, scales it by a scalar λ, and creates a trial offspring $Y_r(t+1) = [y_{r,1}(t+1), y_{r,2}(t+1), ..., y_{r,n}(t+1)]$ by adding the result to $X_{r1}(t)$. Thus, for the sth component of each vector

$$y_{r,s}(t+1) = \begin{cases} x_{r1,s}(t) + \lambda(x_{r2,s}(t) - x_{r2,s}(t)), \text{ if } \mathrm{rnd}_s < \mathrm{CR} \\ x_{r,s}(t), \text{ otherwise} \end{cases} \qquad (7)$$

The scaling factor (λ) and the crossover rate (CR) are control parameters of DE, are set by the user. Both values remain constant during the search process. λ is a real-valued factor (usually in range $[0,1]$), that controls the amplification of differential variations and CR is a real-valued crossover factor in range $[0,1]$ controlling the probability to choose mutated value for x instead of its current value. rnd_s is the uniformly distributed random numbers within the range $[0,1]$ chosen

once for each $s \in \{1,2,...,n\}$.

If the new offspring yields a better value of the objective function, it replaces its parent in the next generation; otherwise, the parent is retained in the population, i.e.,

$$X_r(t+1) = \begin{cases} Y_r(t+1), \text{ if } f(Y_r(t+1)) > f(X_r(t)) \\ X_r(t), \text{ if } f(Y_r(t+1)) \le f(X_r(t)) \end{cases} \quad (8)$$

where $f(\cdot)$ is the objective function to be maximized.

After initialization of the population the process of calculation of the fitness is performed. To judge the quality of a partition provided by a chromosome, it is necessary to have a fitness functions. The fitness functions are defined as

$$fitness_1(X_a(t)) = F_1(X_a(t)) \quad (9)$$

$$fitness_2(X_a(t)) = \frac{1}{F_2(X_a(t))}, \quad (10)$$

$$fitness(X_a(t)) = F(X_a(t)). \quad (11)$$

Maximization of the fitness Functions (9)-(11) leads to maximization (or minimization) of the objective Functions (4)-(6), respectively.

The main difference between the traditional discrete DE (DDE) algorithms and the algorithm proposed here is that it uses the mutation operation adopted from genetic algorithm. The algorithm proposed is based on the mutation adopted from genetic algorithms. In our modification, at the iteration $t+1$ for each vector $X_r(t)$ creates a vector $m_r(t+1) = [m_{r,1}(t+1), m_{r,2}(t+1), ..., m_{r,n}(t+1)]$, which is defined as:

$$m_{r,s}(t+1) = \begin{cases} 1, \text{ if } \text{rand}_s < sigm(y_{r,s}(t+1)) \\ 0, \text{ otherwise} \end{cases} \quad (12)$$

The vector $m_r(t+1)$ represents the changes that will be needed to move the particle from $X_r(t)$ to $X_r(t+1)$. If the component of vector $m_r(t+1)$ is one, it means that this component will be copied from the $X_r(t)$ to $X_r(t+1)$. If the component of $m_r(t+1)$ is null, it means that this component will be mutated. The inversion operator has been used as a mutation operator. The inversion operator takes the components from the $X_r(t)$ corresponding to the null components of the vector $m_r(t+1)$ and puts them to form a new particle. The pseudo-code of an inversion operator is described in Figure 1 ($|S|$ is the cardinality of the set S) [11]:

Let's consider a following example (Figure 2):

```
procedure inversion operator
S = {∅}
for s = 1:n
    if m_{r,s}(t+1) = 1 then x_{r,s}(t+1) = x_{r,s}(t)
      else
          S = S ∪ {s}
    end_if
end_for
    if |S| > 0 then
        s^+ = max S
        s^- = min S
        x_{r,s^-}(t+1) = x_{r,s^+}(t)
        x_{r,s^+}(t+1) = x_{r,s-}(t)
        S = S \ {s^+, s^-}
      else if |S| = 0 then stop
    end_if
```

Figure 1. Pseudo-code of an inversion operator

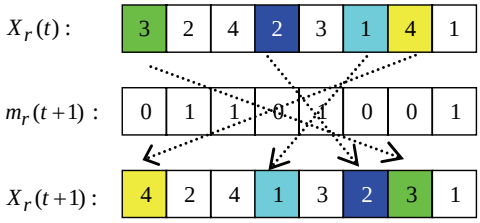


Figure 2. Inversion operator

The components of a vector $X_r(t+1)$, using the pseudo-code described in Figure 1, are calculated as follows:

Step 1. if $m_{r,s}(t+1)=1$ then $x_{r,s}(t+1)=x_{r,s}(t)$, i.e. $x_{r,2}(t+1)=x_{r,2}(t)=2$, $x_{r,3}(t+1)=x_{r,3}(t)=4$, $x_{r,5}(t+1)=x_{r,5}(t)=3$, and $x_{r,8}(t+1)=x_{r,8}(t)=1$

Step 2. $\mathrm{S}=\{s:m_{r,s}(t+1)=0\}=\{1,4,6,7\}$

Step 3. $s^+=\max \mathrm{S}=\max\{1,4,6,7\}=7$, $s^-=\min \mathrm{S}=\min\{1,4,6,7\}=1$

Step 4. $x_{r,s^-}(t+1)=x_{r,s^+}(t)$ and $x_{r,s^+}(t+1)=x_{r,s^-}(t)$, i.e. $x_{r,1}(t+1)=x_{r,7}(t)=4$ and $x_{r,7}(t+1)=x_{r,1}(t)=3$

Step 5. $\mathrm{S}=\mathrm{S}\setminus\{s^+,s^-\}=\{1,4,6,7\}\setminus\{1,7\}=\{4,6\}$

Step 6. $s^+=\max \mathrm{S}=\max\{4,6\}=6$, $s^-=\min \mathrm{S}=\min\{4,6\}=4$

Step 7. $x_{r,s^-}(t+1)=x_{r,s^+}(t)$ and $x_{r,s^+}(t+1)=x_{r,s^-}(t)$, i.e. $x_{r,4}(t+1)=x_{r,6}(t)=1$ and $x_{r,6}(t+1)=x_{r,4}(t)=2$

Step 8. $\mathrm{S}=\mathrm{S}\setminus\{s^+,s^-\}=\{4,6,\}\setminus\{4,6\}=\{\varnothing\}$, i.e. $|\mathrm{S}|=0$ and hence stop.

The stopping criterion of DE could be a given number of consecutive iterations within which no improvement on solutions, a specified CPU time limit, or maximum number of iterations (fitness calculation), $t_{\max}$, is attained. Unless otherwise specified, in this paper we use the last one as the termination criteria, i.e. the algorithm terminates when a maximum number of fitness calculation is achieved.

Extractive summarization works by choosing a subset of the sentences in the original document. This process can be viewed as identifying the most salient sentences in a cluster that give the necessary and sufficient amount of information related to main content of the cluster (topic). In a cluster of related sentences, many of the sentences are expected to be somewhat similar to each other since they are all about the same topic. The approach, proposed in papers [13,19], is to assess the centrality of each sentence in a cluster and extract the most important ones to include in the summary. In centroid-based summarization, the sentences that contain more words from the centroid of the cluster are considered as central. Centrality of a sentence is often defined in terms of the centrality of the words that it contains. In this section we use other criterion to assess sentence salience, developed in [11] which is based on technique proposed in [47].

5. Experiments and Results

For evaluation the performance of our methods we used two document datasets DUC2001 and DUC2002 and corresponding 100-word summaries generated for each of documents. The DUC2001 and DUC2002 are an open benchmark datasets which contain 309 and 567 documents-summary pairs from Document Understanding Conference (http://duc.nist.gov). The datasets DUC2001 and DUC2002 are clustered into 30 and 59 topics, respectively.

At a preprocessing step, the stopwords in each sentence were removed using the stoplist provided in ftp://ftp.cs.cornell.edu/pub/smart/english.stop and the remaining words were stemmed using the Porter's scheme [48].

We use the ROUGE (Recall Oriented Understudy for Gisting Evaluation) toolkit [49,50], which was adopted by DUC for automatically summarization evaluation. It has been shown that ROUGE is very effective for measuring document summarization. It measures summary quality by counting overlapping units such as the N-gram, word sequences and word pairs between the candidate summary and the reference summary. The ROUGE-N measure compares N-grams of two summaries, and counts the number of matches. The measure is defined by Formula (31) [49–51]:

$$\text{ROUGE-N}=\frac{\sum_{S\in Summ_{ref}}\sum_{\text{N-gram}\in S} Count_{match}(\text{N-gram})}{\sum_{S\in Summ_{ref}}\sum_{\text{N-gram}\in S} Count(\text{N-gram})} \qquad (14)$$

where N stands for the length of the N-gram, $Count_{match}(\mathrm{N-gram})$ is the maximum number of N-grams co-occurring in candidate summary and a set of reference-summaries. $Count(\mathrm{N-gram})$ is the number of N-grams in the reference summaries. We show three of the ROUGE metrics in the experimental results: ROUGE-1, ROUGE-2 and ROUGE-SU4. The ROUGE-1 and ROUGE-2 scores are based on the overlap of unigrams and bigrams, respectively, between the candidate summary and the reference summary. The ROUGE-SU4 score is also based on the overlap of bigrams between summaries, but allows for gaps to occur between words (skip-bigram), with a maximum gap length of words, and includes unigram co-occurrence statistics as well.

The optimization procedure used here is stochastic in nature. Hence, for each criterion function (F_1, F_2 and F) it has been run several times for different values of parameters $CR\in[0.3;0.8]$ and $MR\in[0.1;0.5]$. At experiments the size of population and the number of iteration we kept unchanged changing only parameters *CR* and *MR* with step 0.1. For both datasets we take the same number of iterations which is 1000. The population size

is 40 % of the total number of documents in the datasets. The parameters of the DE are reported in Table 1.

The first experiment compares our methods with other methods. We compare our proposed methods with both supervised and unsupervised methods. Among the supervised methods we choose *SVM* [4] and *CRF* [21]. *SVM* is one of the state-of the-art classifiers. *CRF* combines the merits of *HMM* (Hidden Markov Model) and *LR* (Logistic Regression). *HMM* extends Naive Bayes (*NB*) by considering the sequential information, while *LR* is a discriminative version of *NB*. The unsupervised methods we compare include *QCS* [16] and graph-based algorithm *HITS* [27] (Among the several options of graph-based algorithm [27] the method based on the authority score of HITS on the directed backward graph is the best. Therefore it is taken by us for comparison). Table 2 and 3 show the results of all the methods in terms ROUGE-1, ROUGE-2, and ROUGE-SU4 metrics on DUC2001 and DUC2002 datasets, respectively. As shown in Tables 2 and 3, on DUC2001 dataset, the values of ROUGE-1, ROUGE-2 and ROUGE-SU4 metrics of all the methods are better than on DUC2002 dataset. In the Tables 2 and 3 highlighted (***bold italic***) entries represent the best performing methods.

The numerical comparison of our methods with the methods *CRF*, *QCS*, *HITS* and *SVM* is shown in Tables 4 and 5. Here we use relative improvement $\frac{(\text{our method} - \text{other methods})}{\text{other methods}} \times 100$ for comparison. In spite of the fact that among our criterion functions the worst result is obtained by criterion function F_1 but it shows better result than the other methods.

Compared with the best method *QCS* on DUC2001 (DUC2002) dataset the criterion function F_1 improves the performance by 1.05% (0.57%), 0.37% (0.43%) and 0.59% (0.31%) in terms ROUGE-1, ROUGE-2 and ROUGE-SU4 metrics, respectively.

Table 1. Parameters of the DE

Dataset	***Population size,*** N	***Number of iteration,*** $t_{\max}$	***Crossover rate,*** CR	***Mutation rate,*** MR
DUC2001	155	1000	0.6	0.2
DUC2002	285	1000	0.6	0.2

Table 2. ROUGE scores for summarization methods on DUC2001 dataset

Methods	ROUGE-1	ROUGE-2	ROUGE-SU4
F	0.45836	0.19164	0.21574
F_1	0.45603	0.19046	0.21427
F_2	***0.45952***	***0.19338***	***0.21763***
CRF	0.44598	0.18564	0.20934
QCS	0.45129	0.18976	0.21302
HITS	0.43528	0.18317	0.20627
SVM	0.43132	0.18136	0.20372

Table 3. ROUGE scores for summarization methods on DUC2002 dataset

Methods	ROUGE-1	ROUGE-2	ROUGE-SU4
F	0.45119	0.18847	0.21184
F_1	0.44985	0.18896	0.21234
F_2	***0.45412***	***0.18982***	***0.21268***
CRF	0.44155	0.17974	0.20129
QCS	0.44865	0.18766	0.21119
HITS	0.42806	0.16792	0.18924
SVM	0.43405	0.17084	0.19036

Table 4. Comparison our methods with other methods on DUC2001 dataset

Methods	Metrics	F_2	F_1	F
		% of improvement		
CRF	ROUGE-1	3.04	2.25	2.78
	ROUGE-2	4.17	2.60	3.23
	ROUGE-SU4	3.96	2.36	3.06
QCS	ROUGE-1	1.82	1.05	1.57
	ROUGE-2	1.91	0.37	0.99
	ROUGE-SU4	2.16	0.59	1.28
HITS	ROUGE-1	5.57	4.77	5.30
	ROUGE-2	5.57	3.98	4.62
	ROUGE-SU4	5.51	3.88	4.59
SVM	ROUGE-1	6.54	5.73	6.27
	ROUGE-2	6.63	5.02	5.67
	ROUGE-SU4	6.83	5.18	5.90

Table 5. Comparison our methods with other methods on DUC2002 dataset

Methods	Metrics	F_2	F_1	F
		% of improvement		
CRF	ROUGE-1	2.85	2.18	1.88
	ROUGE-2	5.61	4.86	5.13
	ROUGE-SU4	5.66	5.24	5.49
QCS	ROUGE-1	1.22	0.57	0.27
	ROUGE-2	1.15	0.43	0.69
	ROUGE-SU4	0.71	0.31	0.54
HITS	ROUGE-1	6.09	5.40	5.09
	ROUGE-2	13.04	12.24	12.53
	ROUGE-SU4	12.39	11.94	12.21
SVM	ROUGE-1	4.62	3.95	3.64
	ROUGE-2	11.11	10.32	10.61
	ROUGE-SU4	11.73	11.28	11.55

6. Conclusions

We have presented an unsupervised approach to automatic document summarization. Our approach consists of two steps. First sentences are clustered, and then representative sentences are defined and extracted on each cluster. In our study we developed a modified discrete differential evolution algorithm to optimize the objective functions. When comparing our methods to several existing summarization methods on an open DUC2001 and DUC2001 datasets, we found that our methods can improve the summarization results significantly. The methods were evaluated using ROUGE-1, ROUGE-2 and ROUGE-SU4 metrics.

REFERENCES

[1] M. A. Fattah and F. Ren, "GA, MR, FFNN, PNN and GMM based models for automatic text summarization," Computer Speech and Language, Vol. 23, No. 1, pp.

126–144, 2009.

[2] U. Hahn and I. Mani, "The challenges of automatic summarization," IEEE Computer, Vol. 33, No. 11, pp. 29–36, 2000.

[3] I. Mani and M. T. Maybury, "Advances in automated text summarization," MIT Press, Cambridge, 442p, 1999.

[4] J-Y. Yeh, H-R. Ke, W-P. Yang, I-H. Meng, "Text summarization using a trainable summarizer and latent semantic analysis," Information Processing and Management, Vol. 41, No. 1, pp. 75–95, 2005.

[5] S. Ye, T.-S. Chua, M.-Y. Kan, and L. Qiu, "Document concept lattice for text understanding and summarization, Information Processing and Management, 2007, Vol. 43, No. 6, pp. 1643–1662.

[6] R. M. Alguliev and R. M. Aliguliyev, "Effective summarization method of text documents," Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05), France, pp. 264–271, 19–22 September 2005.

[7] R. M. Alguliev and R. M. Alyguliev, "Automatic text documents summarization through sentences clustering," Journal of Automation and Information Sciences, Vol. 40, No. 9, pp. 53–63, 2008.

[8] R. M. Alguliev, R. M. Aliguliyev, and A. M. Bagirov, "Global optimization in the summarization of text documents," Automatic Control and Computer Sciences, Vol. 39, No. 6, pp. 42–47, 2005.

[9] R. M. Aliguliyev, "A novel partitioning-based clustering method and generic document summarization," Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT'06 Workshops) (WI-IATW'06), Hong Kong, China, pp. 626–629, 18–22 December 2006.

[10] R. M. Aliguliyev, "A new sentence similarity measure and sentence based extractive technique for automatic text summarization," Expert Systems with Applications, Vol. 36, No. 4, pp. 7764–7772, 2009.

[11] R. M. Aliguliyev, "Clustering techniques and discrete particle swarm optimization algorithm for multi-document summarization, Computational Intelligence (accepted).

[12] Y. Gong and X. Liu, "Generic text summarization using relevance measure and latent semantic analysis," in: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, USA, pp. 19–25, 9–12 September, 2001.

[13] D. R. Radev, H. Jing, M. Stys, and D. Tam, "Centroid-based summarization of multiple documents," Information Processing and Management, Vol. 40, No. 6, pp. 919–938, 2004.

[14] G. Salton, A. Singhal, M. Mitra, and C. Buckley, "Automatic text structuring and summarization," Information Processing and Management, Vol. 33, No. 2, pp. 193–207, 1997.

[15] K. M. Svore, L. Vanderwende, and C. J. C. Burges, "Enhancing single-document summarization by combining RankNet and third-party sources," in: Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL'07), Prague, Czech Republic, pp. 448–457, 28–30 June 2007.

[16] D. M. Dunlavy, D. P. O'Leary, J. M. Conroy, and J. D. Schlesinger, "QCS: A system for querying, clustering and summarizing documents," Information Processing and Management, Vol. 43, No. 6, pp. 1588–1605, 2007.

[17] K. S. Jones, "Automatic summarizing: the state of the art," Information Processing and Management, Vol. 43, No. 6, pp. 1449–1481, 2007.

[18] D. Zajic, B. J. Dorr, J. Lin, and R. Schwartz, "Multi-candidate reduction: sentence compression as a tool for document summarization tasks," Information Processing and Management, Vol. 43, No. 6, pp. 1549–1570, 2007.

[19] G. Erkan and D. R. Radev, "Lexrank: Graph-based lexical centrality as salience in text summarization," Journal of Artificial Intelligence Research, Vol. 22, pp. 457–479, 2004.

[20] D. Radev, E. Hovy, and K. McKeown, "Introduction to the special issue on summarization," Computational Linguistics, Vol. 28, No. 4, pp. 399–408, 2002.

[21] D. Shen, J.-T.Sun, H. Li, Q. Yang, and Z. Chen, "Document summarization using conditional random fields," Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07), Hyderabad, India, pp. 2862–2867, January 6–12, 2007.

[22] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: probabilistic models for segmenting and labeling sequence data," Proceedings of the 18th International Conference on Machine Learning, pp. 282–289, 28 June–01 July 2001.

[23] X. Wan, "A novel document similarity measure based on earth mover's distance," Information Sciences, Vol. 177, No. 18, pp. 3718–3730, 2007.

[24] S. Fisher and B. Roark, "Query-focused summarization by supervised sentence ranking and skewed word distributions," Proceedings of the Document Understanding Workshop (DUC'06), New York, USA, 8p, 8–9 June 2006.

[25] P. Fung and G. Ngai, "One story, one flow: Hidden Markov story models for multilingual multidocument summarization," ACM Transaction on Speech and Language Processing, Vol. 3, No. 2, pp. 1–16, 2006.

[26] D. M. McDonald and H. Chen, "Summary in context: Searching versus browsing," ACM Transactions on Information Systems, Vol. 24, No. 1, pp. 111–141, 2006.

[27] R. Mihalcea and P. Tarau, "A language independent algorithm for single and multiple document summarizations," Proceedings of the Second International Joint Conference Natural Language Processing (IJCNLP'05), Korea, pp. 602–607, 11–13 October 2005.

[28] J. Li, L. Sun, C. Kit, and J. Webster, "A query-focused multi-document summarizer based on lexical chains,"

Proceedings of the Document Understanding Conference (DUC'07), New York, USA, 4p, 26–27 April 2007.

[29] X. Wan, "Using only cross-document relationships for both generic and topic-focused multi-document summarizations, Information Retrieval, Vol. 11, No. 1, pp. 25–49, 2008.

[30] R. Mihalcea and H. Ceylan, "Explorations in automatic book summarization, Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL'07), Prague, Czech Republic, pp. 380–389, 28–30 June 2007.

[31] Y. Guo and G. Stylios, "An intelligent summarization system based on cognitive psychology," Information Sciences, Vol. 174, No.1–2, pp. 1–36, 2005.

[32] J. Grabmeier and A. Rudolph, "Techniques of cluster algorithms in data mining," Data Mining and Knowledge Discovery, Vol. 6, No. 4, pp. 303–360, 2002.

[33] J. Han and M. Kamber, "Data mining: concepts and technique (2nd Edition), Morgan Kaufman, San Francisco, 800p, 2006.

[34] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," ACM Computing Surveys, Vol. 31, No. 3, pp. 264–323, 1999.

[35] M. G. H. Omran, A. P. Engelbrecht, and A. Salman, "An overview of clustering methods," Intelligent Data Analysis, Vol. 11, No. 6, pp. 583–605, 2007.

[36] K. M. Hammouda and M. S. Kamel, "Efficient phrase-based document indexing for web document clustering," IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 10, pp. 1279–1296, 2004.

[37] T. Li, "A unified view on clustering binary data," Machine Learning, Vol. 62, No. 3, pp. 199–215, 2006.

[38] Y. Li, C. Luo and S. M. Chung, "Text clustering with feature selection by using statistical data," IEEE Transactions on Knowledge and Data Engineering, Vol. 20, No. 20, pp. 641–652, 2008.

[39] Y. Li, D. McLean, Z. A. Bandar, J. D. O'Shea, K. Crockett, "Sentence similarity based on semantic nets and corpus statistics," IEEE Transactions on Knowledge and Data Engineering, Vol. 18, No. 8, pp. 1138–1150, 2006.

[40] X. Liu, Y. Zhou, and R. Zheng, "Sentence similarity based on dynamic time warping," Proceedings of the First International Conference on Semantic Computing (ICSC' 07), Irvine, USA, pp. 250–256, 17–19 September 2007.

[41] X. Wan, J. Yang, and J. Xiao, "Manifold-ranking based topic-focused multi-document summarization, Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07), Hyderabad, India, pp. 2903–2908, January 6–12, 2007.

[42] D. Bollegala, Y. Matsuo, and M. Ishizuka, "Measuring semantic similarity between words using web search engines," Proceedings of 16th World Wide Web Conference (WWW16), Alberta, Canada, pp. 757–766, May 8–12, 2007.

[43] R. L. Cilibrasi and P. M. B. Vitanyi, "The google similarity distance," IEEE Transaction on Knowledge and Data Engineering, Vol. 19, No. 3, pp. 370–383, 2007.

[44] Y. Zhao and G. Karypis, "Empirical and theoretical comparisons of selected criterion functions for document clustering," Machine Learning, Vol. 55, No. 3, pp. 311–331, 2004.

[45] S. Das, A. Abraham, and A. Konar, "Automatic clustering using an improved differential evolution algorithm," IEEE Transaction on Systems, Man, and Cybernetics – Part A: Systems and Humans, Vol. 38, No. 1, pp. 218–237, 2008.

[46] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," Journal of Global Optimization, Vol. 11, No. 4, pp. 341–359, 1997.

[47] M. Pavan and M. Pelillo, "Dominant sets and pairwise clustering," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 29, No. 1, pp. 167–172, 2007.

[48] M. Porter, "An algorithm for suffix stripping," Program, Vol. 14, No. 3, pp. 130–137, 1980.

[49] C.-Y. Lin, "ROUGE: A package for automatic evaluation summaries," Proceedings of the Workshop on Text Summarization Branches Out, Barcelona, Spain, pp. 74–81, 25–26 July 2004.

[50] C.-Y. Lin and E. H. Hovy, "Automatic evaluation of summaries using n-gram co-occurrence statistics," Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL'03), Edmonton, Canada, Vol. 1, pp. 71–78, 27 May–1 June 2003.

[51] H. Nanba and M. Okumura, "An automatic method for summary evaluation using multiple evaluation results by a manual method," Proceedings of the COLING/ACL on Main Conference Poster Sessions, Sydney, Australia, pp. 603–610, 17–18 July 2006.


9772150819004 02