ISSN: 1945-3116(Print), 1945-3124(Online) Vol. 2, No. 4, November 2009

Journal of Software Engineering and Applications

Chief Editor : Dr. Ruben Prieto-Diaz

www.scirp.org/journal/jsea

Scientific Research

Journal Editorial Board

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

J. Software Engineering & Applications, **2009, 2**
Published Online November 2009 in SciRes(www.SciRP.org/journal/jsea)

CONTENTS

Volume 2 Number 4 **November 2009**

Journal of Software Engineering and Applications (JSEA)

Journal Information

SUBSCRIPTIONS

The *Journal of Software Engineering and Applications* (Online at Scientific Research Publishing, www.SciRP.org) is published monthly by Scientific Research Publishing, Inc., USA.

E-mail: jsea@scirp.org

Subscription rates: Volume 2 2009

Print: $50 per copy.

Electronic: free, available on www.SciRP.org.

To subscribe, please contact Journals Subscriptions Department, E-mail: jsea@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: jsea@scirp.org

Reprints (minimum quantity 100 copies)

Reprints Co-ordinator, Scientific Research Publishing, Inc., USA.

E-mail: jsea@scirp.org

COPYRIGHT

PRODUCTION INFORMATION

For manuscripts that have been accepted for publication, please contact:

E-mail: jsea@scirp.org

J. Software Engineering & Applications, 2009, 2: 221-236
doi:10.4236/jsea.2009.24030 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

Explanation vs Performance in Data Mining: A Case Study with Predicting Runaway Projects

Tim MENZIES[1], Osamu MIZUNO[2], Yasunari TAKAGI[3], Tohru KIKUNO[2]

[1]Lane Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, USA; [2]Graduate School of Information Science and Technology, Osaka University, Osaka, Japan; [3]Social Systems Business Group, OMRON Corporation, Shiokoji Horikawa, Japan.
Email: tim@menzies.us, {o-mizuno, kikuno}@ist.osaka-u.ac.jp, yasunari@omron.co.jp

Received May 6th, 2009; revised July 7th, 2009; accepted July 14th, 2009.

ABSTRACT

Often, the explanatory power of a learned model must be traded off against model performance. In the case of predicting runaway software projects, we show that the twin goals of high performance and good explanatory power are achievable after applying a variety of data mining techniques (discrimination, feature subset selection, rule covering algorithms). This result is a new high water mark in predicting runaway projects. Measured in terms of precision, this new model is as good as can be expected for our data. Other methods might out-perform our result (e.g. by generating a smaller, more explainable model) but no other method could out-perform the precision of our learned model.

***Keywords*:** *Explanation, Data Mining, Runaway*

1. Introduction

Every teacher knows that generating succinct explanations means skipping over tedious details. Such explanations can be quickly communicated, but can miss the details needed to apply that knowledge in a real world setting.

An analogous situation occurs with data miners. All data miners are *performance systems*; i.e. they can reach conclusions about a test case. However, only some data miners are *explanation systems* that offer a high-level description of how the learned model functions.

The ability to *explain how* a conclusion was reached is a very powerful tool for helping users to understand and accept the conclusions of a data miner. Despite this, sometimes explanatory power must be *decreased* in order to *increase* the efficacy of the predictor. For example, previously Abe, Muzono, Takagi, *et al.* used a Näive Bayes classifier to generate a predictor for runaway software projects [1–3]. That model *performs* well but, as shown below, cannot easily *explain* how it reaches its conclusions.

This paper repairs the *explainability* of that prior result. Using an iterative exploration of data mining techniques (cross-validation, different rule learners, discretization, feature subset selection), we found a particular combination of methods that yielded succinct explanations of how to predict for runaway software projects while out-performing the Näive Bayes classifier. In hold-out experiments, this new model exhibited perfect precision; i.e. precision = 1.0. Other methods *might* be able to out-perform this new result (e.g. by finding a more succinct and explainable model) but no other method could be more precise (since $0 \le precision \le 1$).

The rest of this paper is structured as follows. First, the software runaway problem is defined and the explanation problems of prior results are discussed. Next, the general problem of explaining a learned model is explored using a range of data miners. And examples from the software engineering literatures (in summary, the best *performing* models may be very poor at *explaining* how those models make their conclusions). A class of data miners called *rule learners* will then be introduced and applied to our data via various *treatments* (some combination of discretizer, feature selector, and learner). The subsequent discussion will review (a) related work; (b) the external validity of these results; as well as (c) general principles of building explainable models via data mining.

2. Runaway Software

Glass defines a "runaway software project" as "a project that goes out of control primarily because of the difficulty of building the software needed by the system" [4]. For Glass "out of control" means "schedule, cost, or functionality that was twice as bad as the original estimates".

Requirements					Estimation					Planning						Team Organi-zations			Management			
R1	R2	R3	R4	R5	E1	E2	E3	E4	E5	P1	P2	P3	P4	P5	P6	O1	O2	O3	M1	M2	M3	class
0	0	0	0	0	2	3	3	2	0	2	0	0	0	0	0	2	1	0	0	0	0	ok
0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	ok
0	0	0	0	3	0	0	2	3	0	0	0	0	0	2	0	0	0	0	0	0	0	ok
3	3	2	2	3	0	0	2	2	0	2	2	0	0	0	1	2	0	0	0	0	0	ok
0	0	0	0	2	0	0	0	0	0	0	2	0	2	2	0	0	0	0	0	2	0	ok
0	3	2	0	0	2	2	2	0	2	0	2	0	0	0	0	0	0	0	0	0	2	ok
0	0	2	3	2	0	0	0	0	0	0	2	0	3	0	0	0	0	0	0	0	0	ok
0	2	3	3	0	1	0	2	0	0	2	2	0	0	2	2	0	0	1	3	0	0	ok
0	2	0	2	3	0	0	0	0	0	2	2	0	2	2	0	0	0	0	0	0	2	ok
0	0	0	0	2	0	2	2	0	0	0	2	0	0	2	0	0	0	0	2	0	0	ok
0	3	3	2	0	0	0	3	3	0	0	0	0	0	0	0	0	0	2	0	0	0	ok
0	2	2	2	0	0	2	0	0	0	0	2	0	2	0	0	0	0	0	0	0	2	ok
0	2	0	2	0	0	0	0	0	0	2	3	3	0	2	2	2	2	0	2	2	1	ok
0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ok
0	2	2	2	2	0	2	2	0	0	0	0	0	0	0	0	3	2	0	3	0	0	ok
0	0	0	0	2	0	2	0	2	3	3	2	0	2	3	2	3	2	0	2	2	2	ok
0	0	0	0	0	0	2	0	0	0	2	2	2	3	2	2	0	0	0	2	2	0	ok
0	0	0	0	1	0	0	0	0	0	0	2	2	0	0	0	0	0	0	0	0	0	ok
0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ok
0	2	3	2	3	0	0	0	0	0	3	0	0	0	3	0	2	0	0	0	3	3	ok
0	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ok
3	2	3	3	2	2	1	3	2	1	0	2	2	2	0	1	3	1	2	2	2	0	ok
2	2	0	2	3	0	0	2	3	0	2	0	2	2	3	2	0	0	0	0	2	3	runaway
2	2	3	3	3	2	2	3	2	3	3	3	3	2	3	2	3	3	0	2	2	2	runaway
3	2	0	0	3	0	0	0	0	0	3	0	0	3	3	0	0	0	0	0	0	0	runaway
0	2	3	2	2	3	0	2	2	1	0	2	0	0	2	2	0	2	2	2	0	2	runaway
0	2	2	2	2	0	3	2	3	3	0	2	2	0	0	2	2	2	0	0	0	0	runaway
2	3	3	2	2	0	0	3	3	2	3	0	3	0	2	3	2	0	2	0	2	2	runaway
3	2	3	2	0	3	2	2	2	0	0	2	2	2	3	0	2	0	2	0	3	3	runaway
2	2	3	3	2	0	0	2	0	2	2	2	2	2	2	0	3	0	2	0	2	0	runaway
0	0	0	0	0	0	0	0	0	0	3	3	3	3	3	3	3	3	0	0	3	3	runaway
2	3	3	3	2	2	3	3	3	3	3	3	3	2	3	3	3	3	2	3	3	0	runaway

Figure 1. Data used in this study, collected using the methods. For an explanation of the columns features, see Figure 2 [1**]**

Requirements features relate to the understanding and commitment of the requirements among the project members
R1: Ambigious requirements
R2: Insufficient explanation of the requirements
R3: Misunderstanding of the requirements
R4: Lack of commitment regarding requirements between the customer and the project members;
R5: Frequent requirement changes

Estimation features relate to the technical methods for carrying out the estimation, and the commitment between project members and customers:
E1: Insufficient awareness of the importance of the estimation;
E2: Insufficient skills or knowledge of the estimation method;
E3: Insufficient estimation of the implicit requirements;
E4: Insufficient estimation of the technical issues;
E5: Lack of stake holders' commitment of estimation.

Planning features relate to the planning or scheduling activity and the commitment to the project plan among project members:
P1: Lack of management review for the project plan;
P2: Lack of assignment of responsibility;
P3: Lack of breakdown of the work products;
P4: Unspecified project review milestones;
P5: Insufficient planning of project monitoring and controlling;
P6: Lack of project members' commitment for the project plan.

Team organization features relate to the state of the projects; e.g. the fundamental skills or experience and morale of project members:
O1: Lack of skills and experience;
O2]: Insufficient allocation of resources;
O3]: Low morale.

Project management factors about management activities:
M1: Project manager lack of resource management throughout a project;
M2: Inadequate project monitoring and controlling;
M3: Lack of data needed to keep objective track of a project.

Figure 2. Explanation of the features seen in Figure 1

Many software projects suffer from runaways:

- In 2001, the Standish group reported that 53% of U.S. software projects ran over 189% of the original estimate [5]. This 189% is not the 200% required by Glass' definition, but it is close enough and large enough to be alarming.
- Figure 1 shows data from 31 real-world projects, 10 of which (32%) are classified as "runaway".

Figure 1 was collected by [1–3] as follows:

- Questions covering the various aspects of software development (see Figure 2) areas were delivered to development companies and collected one month later. These projects are actual industrial software development projects of embedded systems in the period 1996 to 1998.
- The questions were distributed to the project managers or project leaders of various target projects. The detail and purpose of the questionnaire was explained. Answers were coded strongly agree, Agree, Neither agree nor disagree, and Disagree as 3, 2, 1, and 0, respectively.
- All of these projects had completed their development. As a result, some of the projects could be classified as "runaways". Takagi *et al.* took care to ensure that all developers held a consensus view that some prior project had been a runaway. Also, to be classified as a runaway, the researchers used other objective measures such as cost and duration.

Using manual methods, Takagi *et al*. [1] found four features from Figure 3 (*e*3,*e*5,*p*3,*p*5) that seemed promising predictors for runaways. The coefficients of those terms (found via logistic regression) were combined as follows:

$$X(e3, e5, p3, p5) = -8.834e3 + 1.577e5 + 0.964p3 + 1.228p3 + 2.222p5$$

$$P(runaway \mid X) = \frac{e^X}{1+e^X} \quad (1)$$

Unlike prior results [4,6,7], this model is *operational*; it is possible to precisely characterize the strengths and weaknesses of its performance:

- For *high* and *low* values of $P(runaway|X)$, Equation 1 is a perfect predictor for runaways in Figure 1. No project with $P \le 0.03$ is a "runaway" and no project with $P \ge 0.81$ is "ok". This is the majority ($\frac{22}{33}$=67%) of the data in Figure 1.
- In the minority case ($\frac{11}{33}$), P is mid-range ($0.03<P(runaway|X)<0.81$) and Equation 1 yields incorrect predictions in $\frac{4}{11}$ rows.

While an important result, Equation 1 has several drawbacks:

- *Not automatic:* Equation 1 was created after a manual inspection of the data by a team of skilled mathematicians. Such a manual analysis is hard to reproduce or apply to a new data set. Subsequent work by Abe, Takagi, *et al*. [2] automated the method with a Näive Bayes classifier, but this compromised the *explainability* of the predictive model (see below).
- *Only explores one subset:* Takagi *et al*. did not compare the feature subset {*e*3,*e*5,*p*3,*p*5} with other feature subsets. Hence, while they showed that this subset was useful, they did not demonstrate that it was the *most* useful subset.
- *Ambiguous:* At low and high P values, Equation 1 sends a clear signal about what is, and is not, a potentially runaway project. However, at middle-range P values, Equation 1's conclusions are ambiguous and, hence, hard to explain.

3. The Explanation Problem

Learning explainable models is harder than it may appear. This section offers examples where learned models *perform* well, but *explain* themselves poorly.

3.1 Learning Latent Features

Numerous data mining methods check if the available features can be combined in useful ways. In this way, *latent* features within a data set can be discovered.

For example, principal components analysis (PCA) [8] has been widely applied to resolve problems with structural code measurements; e.g. [9]. PCA identifies the distinct orthogonal sources of variation in data sets, while mapping the raw features onto a set of uncorrelated features that represent essentially the same information contained in the original data. For example, the data shown in two dimensions of Figure 3 (left-hand-side) could be approximated in a single latent feature (right-hand-side).

Since PCA combines many features into fewer latent features, the structure of PCA-based models may be very simple. For example, previously [10], we have used PCA and a decision tree learner to find the following predictor for defective software modules:

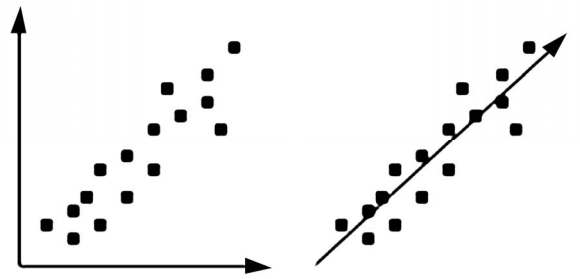


Figure 3. The two features in the left plot can be transferred to the right plot via one latent feature

if $domain_1 \leq 0.180$
then NoDefects
else if $domain_1 > 0.180$
then if $domain_1 \leq 0.371$ then NoDefects
else if $domain_1 > 0.371$ then Defects

Here, "$domain_1$" is one of the latent features found by PCA. This tree seems very simple, yet is very hard to explain to business clients users since "$domain_1$" is calculated using a very complex weighted sum (in this sum, $v(g)$,$ev(g)$,$iv(g)$ are McCabe or Halstead static code metrics [11,12] or variants on line counts):

$$\begin{aligned} domain_1 = {} & 0.241*loc+0.236*v(g) \\ & +0.222*ev(g)+0.236*iv(g)+0.241*n \\ & +0.238*\tilde{v}0.086*l+0.199*d \\ & +0.216*i+0.225*e+0.236*b+0.221*t \\ & +0.241*lOCode+0.179*lOComment \\ & +0.221*lOBlank+0.158*lOCodeAndComment \\ & +0.163*uniq_Op+0.234*uniq_Opnd \\ & +0.241*total_Op+0.241*total_Opnd \\ & +0.236*branchCount \end{aligned} \quad (2)$$

As we shall see below, other learners can yield effective models that are simpler to explain without using complex latent features.

3.2 Ensemble Learning

Data mining for SE means summarizing the complex behavior of a group of developers struggling to build intricate artifacts. Data mining over such complex multi-dimensional data often requires *fusing* together the results from multiple learners [13]. Such ensembles may *perform* well but, as we shall see, are hard to *explain*.

In *basic ensemble method* (BEM), l learners are run on various subsets of the available data. These learners use EQ $x\backslash s\backslash do5(j)(r\backslash,s)$ that returns the probability of the target classes s. BEM returns the mean probability:

$$\hat{x}_{BEM} = \frac{1}{l}\sum_{j=1}^{l} x_j(r,s) \quad (3)$$

The *linear generalized ensemble method* (GEM) returns a weighted sum of the conclusions of each learner x in the ensemble.

$$\hat{x}(a)_{GEM} = \sum_{j=1}^{l} a_j x_j(r,s) \quad (4)$$

where α_j is the normalized performance score of x_j on the training data (so learners that performed the worst, contribute the least).

For some data sets, the combination rule is non-linear and complex. For example, Toh *et al.* [13]'s variant of Equation 4 uses a Jacobian matrix for $\hat{x}$ with different coefficients for each feature $r_i \in r$ and target class $s_m \in s$. These coefficients are learned via multivariate polynomial regression. Toh *et al.* report that their resulting ensemble *performs* better than simpler schemes. However, it may be harder to *explain* the ensemble since that explanation must cover:

- The learning methods used to generate x_j;
- The combination rule that computes $\hat{x}$; and
- The regression method used to tune the coefficients used in the combination method.

Such an explanation is not required if the users are willing to accept the conclusions of the learner, without explanation. However, for data sets as small Figure 1, it seems reasonable to expect that a simple explanation of runaway projects should be possible. Also, if managers are to use the results of the learner as part of their deliberations, they need some succinct structures that they can reflect over.

3.3 Näive Bayes Classifiers

It is hardly surprising that complex latent features (e.g. Equation 2) or intricate combinations of multiple learners (e.g. Equation 4) are hard to explain. What is surprising is how hard it is to explain the results of even a single, supposedly simple, learner. For example, this section offers a complete description of how a Näive Bayes classifiers makes its conclusions. The reader is asked to consider how many users would understand this description (in our experience, we have yet to meet a single one).

A Näive Bayes classifier [14] is based on Bayes' Theorem. Informally, the theorem says *next=old*new* i.e. what we'll believe *next* comes from how *new* evidence effects *old* beliefs. More formally:

$$P(H|E) = \frac{P(H)}{P(E)} \prod_i P(E_i|H) \quad (5)$$

i.e. given fragments of evidence E_i and a prior probability for a class $P(H)$, the theorem lets us calculate a posterior probability $P(H|E)$.

When building predictors for runaways, the posterior probability of each hypothesis class ($H \in$ {"ok" or "runaway"}) is calculated, given the features extracted from a project such "ambiguous requirements" or "low morale" or any other of the features shown in Figure 2. The classification is the hypothesis H with the highest posterior $P(H|E)$.

Näive Bayes classifiers are called "näive" since they assume independence of each feature. While this assumption simplifies the implementation (frequency counts are required only for each feature), it is possible that correlated events are missed by this "näive" approach. Domingos and Pazzani show theoretically that the independence assumption is a problem in a vanishingly small percent of cases [15]. This explains the repeated empirical result that, on average, Näive Bayes classifiers perform as well as other seemingly more so

phisticated schemes (e.g. see Table 1 in [15]).

Equation 5 offers a simple method for handling missing values. Generating a posterior probability means of tuning a prior probability to new evidence. If that evidence is missing, then no tuning is needed. In this case Equation 5 sets $P(E_i|H)=1$ which, in effect, makes no change to $P(H)$.

When estimating the prior probability of hypothesis H, it is common practice [16] to use an *M-estimate* as follows. Given that the total number of hypothesis is C, the total number of training instances is I, and $N(H)$ is the frequency the hypothesis H within I, then

$$P(H) = \frac{N(H) + m}{I + m \cdot C} \tag{6}$$

Here m is a small non-zero constant (often, m=1). Three special cases of Equation 6 are:

- For high frequency hypothesis in large training sets, $N(H)$ and I are much larger than m and $m \cdot C$, so Equation 6 simplifies to $P(H)= \frac{N(H)}{I}$, as one might expect.
- For low frequency classes in large training sets, $N(H)$ is small, I is large, and the prior probability for a rare class is never less than $\frac{1}{I}$; i.e. the inverse of the number of instances. If this were not true, rare classes would never appear in predictions.
- For very small data sets, I is small and $N(H)$ is even smaller. In this case, Equation 6 approaches the inverse of the number of classes; i.e. $\frac{1}{C}$. This is a useful approximation when learning from very small data sets when all the data relating to a certain class has not yet been seen.

The prior probability calculated in Equation 6 is a useful lower bound for $P(E_i|H)$. If some value v is seen $N(f=v|H)$ times in feature f 's observations for hypothesis H, then

$$P(E_i|H)= \frac{N(f=v|H)+l \quad P(H)}{N(H)+l} \tag{7}$$

Here, l is the *L-estimate* and is set to a small constant (Yang &Webb [16] recommend l=2). Two special cases of are:

- A common situation is when there are many examples of an hypothesis and numerous observations have been made for a particular value. In that situation, $N(H)$ and $N(f=v|H)$ are large and Equation 7 approaches $\frac{N(f=v|H)}{N(H)}$, as one might expect.
- In the case of very little evidence for a rare hypothesis, $N(f=v|H)$ and $N(H)$ are small and Equation 7 approaches $\frac{l \cdot P(H)}{l}$; i.e. the default frequency of an observation in a hypothesis is a fraction of the probability of that hypothesis. This is a useful approximation when very little data is available.

For numeric features it is common practice for Näive Bayes classifiers to use the Gaussian probability density function [17]:

$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)2}{2\sigma^2}} \tag{8}$$

where $\{\mu,\sigma\}$ are the feature's {mean, standard deviation}, respectively. To be precise, the probability of a continuous feature having exactly the value x is zero, but the probability that it lies within a small region, say $x \pm \varepsilon/2$, is $\varepsilon \times g(x)$. Since ε is a constant that weighs across all possibilities, it cancels out and needs not be computed.

Näive Bayes classifiers are frustrating tools in the data mining arsenal. They exhibit excellent performance, but offer few clues about the structure of their models. The means and standard deviations for Figure 1 are shown in Figure 44. Note that this figure is an incomplete characterization of Figure 1. For example, row 1 of Figure 4 suggests that $r1$ ("ambiguous requirements") for "ok" is a Gaussian distribution with a mean of 0.27 and a standard deviation of 0.86. A visual inspection of column one values for "ok" projects in Figure 1 shows that this is not true: $r1$ is usually zero except in two cases where it takes the value of three.

One method of handling non-Gaussians like $P(r1=X|ok)$ is Johns and Langley's *kernel estimation* technique [18]. This technique approximates a continuous distribution sampled by n observations $\{ob_1, ob_2, ..., ob_n\}$ as the sum of multiple Gaussians with means of multiple Gaussians with means $\{ob_1, ob_2, ..., ob_n\}$ and standard deviation

	P(ok) = 0.68		P(runaway) = 0.32	
feature	mean	sd	mean	sd
r1	0.2727	0.8624	1.35	1.0500
r2	0.9545	1.0650	1.65	0.8078
r3	0.9545	1.1571	1.95	1.3500
r4	0.8864	1.0759	1.65	1.0500
r5	1.4091	1.2670	1.90	1.0440
e1	0.3182	0.6998	1.00	1.2649
e2	0.7273	1.0082	1.00	1.2649
e3	0.8182	1.0824	1.65	1.0500
e4	0.5455	0.9642	1.65	1.2460
e5	0.2727	0.7497	1.40	1.2806
p1	0.6818	0.9833	1.80	1.3077
p2	0.9545	0.8516	1.50	1.1619
p3	0.3409	0.7744	1.80	1.1225
p4	0.6818	0.9833	1.35	1.0500
p5	0.7500	0.9857	2.25	1.0062
p6	0.4545	0.7820	1.70	1.1874
o1	0.6818	1.0824	1.65	1.2460
o2	0.3636	0.7100	1.30	1.3454
o3	0.2273	0.5979	1.00	1.0000
m1	0.6136	0.9762	0.60	0.9950
m2	0.4773	0.8323	1.50	1.1619
m3	0.5455	0.9404	1.50	1.2845

Figure 4. Means and standard deviations from Figure 1

$\sigma = \frac{1}{\sqrt{n}}$ In this approach, to create a highly skew distribution like $P(r1=X|ok)$, multiple Gaussians would be added together at $r1=0$. Conclusions are made by asking all the Gaussians which class they believe is most likely.

3.4 Näive Bayes and Software Engineering

NäiveBayes classifiers are widely used in the SE literature for several reasons. NäiveBayes classifiers summarize the training data in one frequency table per class. Hence, they consume very little memory and can quickly modify their knowledge by incrementing the frequency count of feature ranges seen in new training examples. Also, many studies (e.g. [15,19,20]) report that Näive Bayes exhibit excellent performance compared to other learners.

For example, recently Menzies, Greenwald & Frank [21] have built predictors for software detectors using a Näive Bayes classifier and two explanation systems- the OneR rule learner and the J4.8 decision tree learner. In that study, the learner with the *worst* explanation power (Näive Bayes) had the *best* performance, by far. For the data sets explored by Menzies, Greenwald & Frank, the median advantage of Näive Bayes, the C4.5 decision tree learner [22], and the OneR rule learner [23] over the other learners was 52.4%, 0%,-16.7%, respectively (see Figure 5). On analysis, Menzies, Greenwald & Frank concluded that Näive Bayes worked so well because of the the product calculation of Equation 5. They reasoned as follows. Many static code features have similar information content. Hence, minor changes in how the training data was sampled yielded different "best" features for predicting defects. The best predictions come from mathematical methods like Näive Bayes that accumulate the signal from many code features (using Equation 5's product rule). Decision tree learners like C4.5 and rule learners like OneR, on the other hand, do not perform well in this domain since they assume hard and fast boundaries between what is defective and what is not.

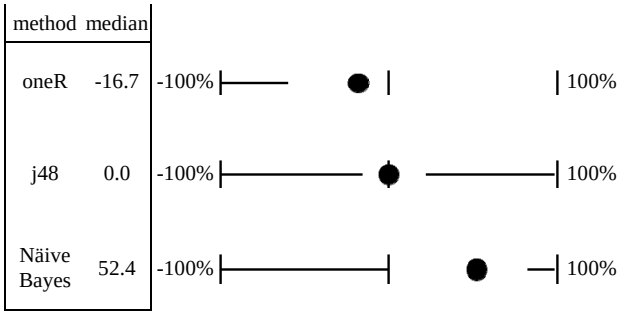


Figure 5. Quartile charts from Menzies, Greenwald & Frank [21]. The charts show the *differences* when learners were applied to the same the training and test data

Performance was measured using *recall*; i.e. the percent of the defective modules found by the learners. The the upper and lower quartiles are marked with black lines. The median is marked with a black dot. Vertical bars are added to mark (i) the zero point and (ii) the minimum possible value and (iii) the maximum possible value. The median performance of Näive Bayes was much higher than the other methods.

In summary, when mining software engineering data, there are many reasons to start with a Näive Bayes classifier. Abe, Muzono, Takagi, *et al*. [2] used such classifiers to extend their prior work on runaway software projects [1,3]. However, this classifier was only a *performance system*. not an *explanation system*, so it could not offer insights into, say, how to best change a software project in order to avoid runaways. As shown above, Näive Bayes classifiers do not generate such succinct generalizations. This is a problem since what developers really want to know is what should be done to avoid runaway status.

3.5 Discussion of the Explanation Problem

As the mathematics gets more elaborate, it becomes harder to explain a Näive Bayes classifier to a typical business user:

- Many users are not trained mathematicians. Hence, they may be confused by Equation 5, Equation 6, Equation 7 and Equation 8.
- Presenting the internal statistics (e.g. Figure 4) is uninformative, at least for the business users we have worked with.
- The problem is compounded if the data is non-Gaussian (like Figure 1) since this requires explaining kernel estimation.
- Worse, a standard Näive Bayes classifier (with our without kernel estimation) can not answer business-level questions such as "what minimal changes should be make to most decrease the odds of runaway projects?"

To be fair, Näive Baye's explanation problems are seen in other kinds of data miners:

- The problems with PCA and ensemble-based learners were discussed above.
- Tree learners such as C4.5 [22] or CART [24] execute in local top-down search, with no memory between different branches. Hence, the same concept can be needlessly repeated many times within the tree. Such trees can be cumbersome, needlessly large, and difficult to understand.
- Clustering algorithms [25] and nearest neighbor methods [26,27] do not condense their working memory into succinct descriptions. Rather, inferences on new information are made by a query over all the old information.
- Simulated annealers [28] learn constraints to an input space that results in higher values in the output space. However, there is no generalization or summarization in a simulated annealer such as which subset of the input space is most important to control.
- Neural networks store their knowledge as weights distributed across a network. Concepts have no centralized

location so it is impossible to inspect, say, all the information about one idea at one location in a network [29].

The problem of explaining the performance of these learners to end-users has been explored extensively in the literature (see the review in [30]). Often, some *post-processor* is used to convert an opaque model into a more understandable form:

- Towell and Shavlik generate refined rules from the internal data structures of a neural network [29].
- Quinlan implemented a post-processor to C4.5 called C45 rules that generates succinct rules from cumbersome decision tree branches via (a) a greedy pruning algorithm followed by (b) duplicate removal then (c) exploring subsets of the rules relating to the same class [22].
- TARZAN was another post-processor to C4.5 that searched for the smallest number of decisions in decision tree branches that (a) pruned the most branches to undesired outcomes while (b) retaining branches leading to desired outcomes [31].

4. Learning Methods

4.1 Rule Learners

Rather than patch an opaque learner with a post-processor, it may be better to build learners than directly generate succinct high-level descriptions of a domain. For example, RIPPER [32] is one of the fastest *rule learners* known in the literature. The generated rules are of the form *condition* ⟶ *conclusion*:

$$\underbrace{Feature_1 = Value_1 \wedge Feature_2 = Value_2 \wedge \ldots}_{condition} \rightarrow \underbrace{Class}_{conclusion}$$

The rules generated by RIPPER perform as well as C45rules, yet are much smaller and easier to read [32].

Rule learners like RIPPER and PRISM [33] generate small, easier to understand, symbolic representations of the patterns in a data set. PRISM is a less sophisticated learner than RIPPER and is not widely used. It was initially added to this study to generate a lower bound on the possible performance. However, as we shall see, it proved surprisingly effective.

1. Find the majority class C
2. Create a *R* with an empty condition that predicts for class *C*.
3. Until *R* is perfect (or there are no more features) do
   (a) For each feature *F* not mentioned in *R*
      - For each value $v \in F$, consider adding $F=v$ to the condition of *R*
   (b) Select *F* and *v* to maximize $\frac{p}{t}$ where *t* is total number of examples of class *C* and *p* is the number of examples of class *C* selected by $F=v$. Break ties by choosing the condition with the largest *p*.
   (c) Add $F=v$ to *R*
4. Print *R*
5. Remove the examples covered by *R*.
6. If there are examples left, loop back to (1)

Figure 6. PRISM pseudo-code

Like RIPPER, PRISM is a *covering* algorithm that runs over the data in multiple passes. As shown in the pseudo-code of Figure 6, PRISM learns one rule at each pass for the *majority class* (e.g. in Figure 6, at pass 1, the majority class is *ok*). All the examples that satisfy the condition are marked as *covered* and removed from the data set. PRISM then recurses on the remaining data.

The output of PRISM is an ordered *decision list* of rules where $rule_j$ is only tested if all conditions in $rule_{i<j}$ fail. PRISM returns the conclusion of the first rule with a satisfied condition.

One way to visualize a covering algorithm is to imagine the data as a table on a piece of paper. If there exists a clear pattern between the features and the class, define that pattern as a rule and cross out all the rows covered by that rule. As covering recursively explores the remaining data, it keeps splitting the data into:

- What is easiest to explain, and
- Any remaining ambiguity that requires a more detailed analysis.

PRISM is a näive covering algorithm and has problems with *residuals* and *over-fitting*. If there are rows with similar patterns and similar frequencies occur in different classes, then:

- These *residual* rows are the *last* to be removed for each class;
- So the *same* rule can be generated for *different* classes.

In *over-fitting*, a learner fixates on spurious signals that do not predict for the target class. PRISM's over-fitting arises from part 3.a of Figure 6 where the algorithm loops through all features. If some feature is poorly measured, it might be noisy (contains spurious signals). Ideally, a rule learner knows how to skip over noisy features.

RIPPER addresses residuals and over-fitting problem three techniques: *pruning*, *description length* and *rule-set optimization* for a full description of these techniques, see [34]. In summary:

- *Pruning:* After building a *rule*, RIPPER performs a back-select to see what parts of a *condition* can be deleted, without degrading the performance of the rule. Similarly, after building a *set of rules*, RIPPER performs a back-select to see what *rules* can be deleted, without degrading the performance of the rule set. These back-selects remove features/rules that add little to the overall performance. For example, back pruning could remove the residual rules.
- *Description length:* The learned rules are built while minimizing their *description length*. This is an information theoretic measure computed from the size of the learned rules, as well as the rule errors. If a rule set is over-fitted, the error rate increases, the description length grows, and RIPPER applies a rule set pruning operator.
- *Rule set optimization* tries replacing rules strawman alternatives (i.e. rules grown very quickly by some näive method).

4.2 Performance Measures

Our results are presented in terms of the following performance measures. Suppose we have some historical log, like Figure 1 that can comment on the correct classification of each row. By comparing the historical log with the output of the learner, we can define several measures of success. Let {*A*,*B*,*C*,*D*} denote the true negatives, false negatives, false positives, and true positives (respectively) found by a binary detector (binary detectors work on data sets with two classes, like Figure 1). *A*,*B*,*C*,*D* can be combined in many ways. For example, accuracy (or *acc*) is the percentage of true positives (*D*) and negatives (*A*) found by the detector.

$$acc=accuracy=(A+D)/(A+B+C+D) \quad (9)$$

Also, recall (or *pd*) comments on how much of the target was found.

$$pd=recall=D/(B+D) \quad (10)$$

Precision (or *prec*) comments on how many of the instances that triggered the detector actually containing the target concept.

$$prec=precision=D/(D+C) \quad (11)$$

The *f*-measure is the harmonic mean of precision and recall. It has the property that if *either* precision or recall is low, then the *f*-measure is decreased. The *f* measure is useful for dual assessments that include *both* precision and recall.

$$f-measure=\frac{2\cdot prec\cdot pd}{prec+pd} \quad (12)$$

All these measures fall in the range $0\leq \{pd,prec,f,acc\} \leq 1$. Also, the *larger* these values, the better the model.

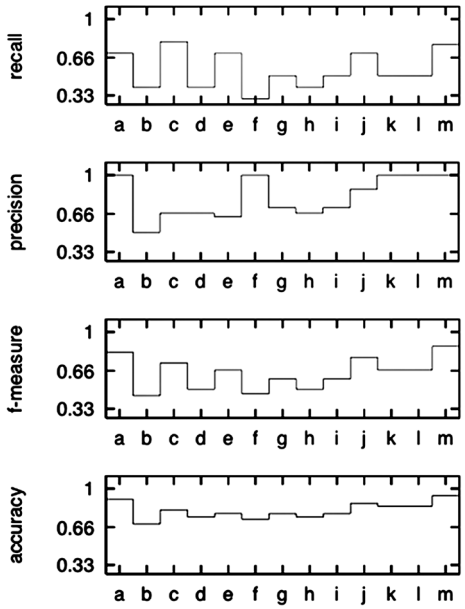
4.3 Experiments with the Learning Methods

Various combinations of the learning method described above were applied to Figure 1. The results are shown in Figure 7. In all 13 *treatments* where applied to Figure 1. Each *treatment* is some combination of a data filter, a learner, and a assessment method. This section discusses how each treatment was designed using results from the proceeding treatments.

Before moving on, we call attention to the *accuracy* results of Figure 7. Observe how *accuracy* can be a remarkably insensitive performance measure; i.e. it remained roughly constant, despite large changes in recall and precision. This result has been seen in many other data sets [21,35]. Hence, accuracy is deprecated by this paper.

4.3.1 Cross-Validation

Treatment *a* is a simple application of RIPPER to Figure 1. The learned theory was applied back on the training data used to generate it; i.e. all of Figure 1. As shown in Figure 7, this produced one of the largest *f*-measures seen in this study.



	bins	learner	#features	#tests
a	n/a	ripper	22	1
b	n/a	ripper	22	10
c	n/a	nb	22	10
d	3	ripper	22	10
e	3	nb	22	10
f	3	prism	22	10
g	3	ripper	1 (r1)	10
h	3	ripper	2 (r1 + p5)	10
i	3	bayes	1 (r1)	10
j	3	bayes	2 (r1 + p5)	10
k	3	prism	1 (r1)	10
l	3	prism	2 (r1 + p6)	10
m	3	prism	3 (r1 + p6 + o3)	10

Figure 7. Results from this study. The four plots, shown at top, come from the 13 treatments shown at bottom

Treatment *a* assessed a learned model using the data that generated it. Such a *self-test* can lead to an over-estimate of the value of that model. Cross-validation, on the other hand, assesses a learned model using data *not* used to generate it. The data is divided into, say, 10 buckets. Each bucket is set aside as a test set and a model is learned from the remaining data. This learned model is then assessed using the test set. Such cross-validation studies are the preferred evaluation method when the goal is to produce predictors intended to predict future events [17].

In treatment *b*, a cross-validation experiment was applied to the data. The treatment *b* results shows how badly treatment *a* overestimated the performance: changing the training data by as little as 10% nearly halved the precision and recall. Clearly, the conclusions from the self-test from this data set are *brittle*; i.e. unduly

altered by minor changes in the training data.

Treatment *c* illustrates the explanation vs performance trade-off discussed in the introduction. As mentioned above, the output from rule learners can be far easier to explain than the output of treatment *c*; i.e. a Näive Bayes classifier (with kernel estimation) running on data sets with non-Gaussian distributions like Figure 1. So, if optimizing for *explainability*, an analyst might favor rule learners over Bayes classifiers. On the other hand, Figure 7 shows treatment *c* out-performing treatment *b*, especially in terms of recall. So, if optimizing for *performance* an analyst might favor a Bayes classifier.

Note that treatment *c* uses the method favored by the previous high water mark in this research [2]. In the sequel, we show how this study found data mining methods that significantly out-perform that prior work.

4.3.2 Discretization

Treatments *d*,*e* and *f* explore *discretization*. Discretization clumps together observations taken over a continuous range into a small number of regions. Humans often discretize real world data. For example, parents often share tips for "toddlers"; i.e. humans found between the breaks of *age*=1 and *age*=3. Many researchers report that discretization improves the performance of a learner since it gives a learner a smaller space to reason about, with more examples in each part of the space [16,20], [36,37].

Discretization can generally be described as a process of assigning data attribute instances to bins or buckets that they fit in according to their value or some other score. The general concept for discretization as a binning process is dividing up each instance of an attribute to be discretized into a number distinct buckets or bins. The number of bins is most often a user-defined, arbitrary value; however, some methods use more advanced techniques to determine an ideal number of bins to use for the values while others use the user-defined value as a starting point and expand or contract the number of bins that are actually used based upon the number of data instances being placed in the bins. Each bin or bucket is assigned a range of the attribute values to contain, and discretization occurs when the values that fall within a particular bucket or bin are replaced by identifier for the bucket into which they fall.

After Gama and Pinto [38], we say that discretization is the process of converting a continuous range into a histogram with *k* break points $b_1 \ldots b_k$ where $\forall i < j : b_i \le b_j$. The histogram divides a continuous range into bins (one for each break) and many observations from the range may fall between two break points b_i and b_{i+1} at frequency counts c_i.

Simple discretizers are *unsupervised* methods that build their histograms without exploiting information about the target class; e.g.

- *equal width*: $\forall i, j : (b_i - b_{i-1}) = (b_j - b_{j-1})$;
- *equal frequency*: $\forall i, j : (c_i = c_j)$. For Näive Bayes classifiers working on *n* instances, Yang & Webb [16] advocate equal frequency with $c_i = c_j = \sqrt{n}$.

For example, Figure 1 holds 32 instances so a *b*=3 equal frequency discretion hopes to place $\frac{32}{3} \approx 10$ values into each part of the histogram. However, Figure 1 does not have ten instances for each feature value so, as shown in Figure 8, a skewed histogram is generated.

More sophisticated discretizers are *supervised* methods that build their histograms using knowledge of the target class. Specifically, the continuous range is explored looking for a break that is a *cliff*; i.e. a point where the class frequencies are most different above and below the cliff. Once a top-level *cliff* is found, this method usually recurses into each region above and below the *cliff* to find the next best sub-cliff, sub-sub-cliff, and so on.

For example, the Fayyad & Irani [37] supervised discretizer assumes that the best cliff is the one that most divides target classes. In terms of information theory, this can be measured using *entropy*; i.e. the number of bits required to encode the class distribution. If the classes in a sample of *n* instances occur at frequencies counts $c_1, c_2, \ldots$, then the entropy of that sample is

$$Ent(c_1, c_2 \ldots) = -\frac{c_1}{n} \cdot log_2\left(\frac{c_1}{n}\right) - \frac{c_2}{n} \cdot log_2\left(\frac{c_2}{n}\right) - \ldots$$

If a break divides *n* numbers into two regions of size n_1, n_2, then the best cliff is the one that minimizes the sum of the entropy below and above the cliff; i.e. $\frac{n_1}{n} \cdot Ent_1 + \frac{n_2}{n} \cdot Ent_2$.

Various discretizers were explored, with disappointing results:

- Yang & Webb's rule ($c_i = \sqrt{n} = \sqrt{33} \approx 6$) was not useful here since our data has less than 6 distinct values per feature.
- Fayyad&Irani's method reduced most features to a single bin; i.e. it found no information gain in any parts

feature	range	frequency	
o3	0	24	■■■■■■■■■■■■■■■■■■■■■■■■
	1	1	■
	2,3	7	■■■■■■■
p6	0	19	■■■■■■■■■■■■■■■■■■■
	1,2	10	■■■■■■■■■■
	3	3	■■■
r1	0,1	23	■■■■■■■■■■■■■■■■■■■■■■■
	2	5	■■■■■
	3	4	■■■■

Figure 8. Some 3*bin* results from Figure 1

of our ranges.

- Best results were seen with a simple *3bin* equal frequency scheme (i.e. $|b|=3$) in Treatment *f* where PRISM achieved precisions as high as the RIPPER self-test (treatment *a*). However, the same experiment saw the worst recall.
- The same 3*bin* scheme offered little help to RIPPER or Näive Bayes (see treatments *d*,*e*).

Since the precision results were the most promising seen to date, 3*bin* was retained for the rest of our experiments. Other methods were then employed to achieve the benefits of 3*bin* (high precision) without its associated costs (low recall).

4.3.3 Feature Subset Selection

The remaining treatments (*g*,*h*,*i*,*j*,*k*,*l*,*m*) explore how different *feature subsets* change the performance of the learning. A repeated result in the data mining community is that simpler models with equivalent or higher performance can be built via *feature subset selection* algorithms that intelligently prune useless features [19]. Features may be pruned for several reasons:

- They may be noisy; i.e. contain spurious signals unrelated to the target class;
- They may be uninformative; e.g. contain mostly one value, or no repeating values;
- They may be correlated to other variables- in which case, they can be pruned since their signal is also present in other variables.

The reduced feature set has many advantages:

- Miller has shown that models generally containing fewer variables have less variance in their outputs [39].
- The smaller the model, the fewer are the demands on interfaces (sensors and actuators) to the external environment. Hence, systems designed around small models are easier to use (less to do) and cheaper to build.
- In terms of this article, the most important aspect of learning from a reduced features set is that it produces smaller models. Such smaller models are easier to explain (or audit).

One such feature subset selector is Kohavi & Johns' WRAPPER algorithm [40]. Starting with the empty set, WRAPPER adds some combinations of features and asks some target learner to build a model using just those features. WRAPPER then grows the set of selected features and checks if a better model comes from learning over the larger set of features.

If we applied WRAPPER to our three learners (RIPPER, PRISM, Näive Bayes), then WRAPPER's search through the 22 features of Figure 1 could require $3 \cdot 2^{22}=12{,}582{,}912$ calls to a learner. In practice, a heuristic search drastically reduced this search space. WRAPPER stops when there are no more features to select, or there has been no significant improvement in the learned model for the last five additions (in which case, those last five additions are deleted). Technically speaking, this is a hill-climbing forward select search with a "stale" parameter set to 5. For data sets as small as Figure 1, WRAPPER terminates in an under a minute (but for large data sets, other feature selectors would be required-see [19] for a survey).

	feature	PRISM	Näive Bayes	RIPPER	average
group #1 : usually selected	r1	10	10	6	8.7
	o3	7			7
	p5		8	4	6
	p6	8	1		4.5
group #2: sometimes selected	m3		3		3
	r2			2	2
	p2		2	1	1.5
	e1	1	2	1	1.3
	o2	1	2	1	1.3
	e2			1	1
group #3: rarely selected	e3			1	1
	m2	1			1
	o1	1			1
	p1		1	1	1
	p3			1	1
	p4		1		1
	r3	1			1
	r4				
group #4: never selected	r5				
	e4				
	e5				
	m1				

Figure 9. Number of times WRAPPER selected features in ten experiments on 90% samples of the data

Figure 9 shows the results of running 10 WRAPPER experiments on Figure 1 (discretized via 3*bin*) for our three learners. In each experiment, 10% of Figure 1 (selected at random) was ignored:

- Group #1 shows the features that, on average, were selected in the majority of ten runs (on average, 6 times or more).
- Group #2 shows the features that were selected 2 to 5 times.
- Group #3 shows the features that were selected only once.
- Group #4 shows the features that were never selected.

There are only three features in Group #1 suggesting that many of the Figure 1 features could be ignored. This has implications for the cost of data collection and the explaining runaway projects:

Data collection could be constrained to just Group #1, and perhaps *p*6 (which PRISM selected eight times). Such a constrained data collection program would be cheaper to conduct, especially over a large organization.

- Figure 10 shows a rule predicting runaway projects found by PRISM using just the features recommend by WRAPPER (*r*1, *p*6, *o*3) on 3*bin* discretized data. The figure shows that just using the top-ranked features of

1: If o3 = 1	then ok
2: If r1 = 0,1 and p6 = 1	then ok
3: If r1 = 3 and p6 = 1,2	then ok
4: If r1 = 0,1 and p6 = 1,2 and o3 = 0	then ok
5: If r1 = 1,2	then runaway
6: If p6 = 3	then runaway
7: If r1 = 3 and p6 = 0	then runaway
8: If r1 = 0,1 and p6 = 1,2 and o3 = 2,3	then runaway
9: If r1 = 0,1 and p6 = 1,2 and o3 = 0	then runaway

Figure 10. Rules generated by treatment *m*

Figure 9 yields a very succinct, easy to explain model.

Treatments *g,h,...m* show the results of applying the top-ranked features to the discretized data. For each learner, if WRAPPER usually selected *N* features, then that learner was tested in a 10-way cross-validation using the top ranked feature, the second-top ranked features, and so on up to using *N* features.

4.3.4 Best Results

The best results were obtained in treatment *m*. That treatment applied PRISM using the three features usually selected by WRAPPER+PRISM: *r*1, *o3*, *p6*. This resulted in Figure 10.

Figure 8 showed *r*1 {0,1}, *p*6 {1,2}, *o*3=0 is a frequent pattern in our data. Hence, after a covering algorithm removes all other more interesting structures, the residual rows can contain this frequent pattern. This, in turn, means that identical rules could be generated for different classes; e.g. rules 4&9 of Figure 10 (this is the *residual rule* problem discussed above).

It is important to read these rules top to bottom since a rule fires *only* if *all* the rules above it *fail*. In practice, this means that the residual rule 9 is never used (it is blocked by rule 4).

A 10-way cross-validation study showed that this rule generation method yields an average precision, recall, and *f*-measure across the 10-way of 1, 0.85, and 0.92 (respectively). This result is actually much better than it appears. To achieve average precisions and recalls of 1 and 0.85 in such a 10-way is something of an accomplishment. In a 10-way cross-validation on the 33 records of Figure 1, the test set is of size three or four. In such a small test set, a single outlier project can have a large and detrimental result on the collected statistics.

4.3.5 User Studies

To test the *explainability* of Figure 10, we ran a session with eight software engineers managing large software verification projects.

Pseudocode for Näive Bayes (with kernel estimation) and PRISM (Figure 6) was introduced. PRISM was summarized this way: "each rule handles some examples, which are then removed, and the algorithm repeats on the remaining data."

Within an hour, the engineers were hand-simulating PRISM. Using a pen and ruler, all the rows of Figure 1 that matched rule #1 (in Figure 10) were identified and crossed off. The rows that matched rule #2 were identified, then crossed off. The engineers stopped after simulating PRISM's activities on two or three rules, making comments like "I see what is going on- the learner is finding and handling the most obvious next thing." Significantly, none of the engineers tried to apply Näive Bayes; i.e. m-estimates, l-estimates, the approximation, and the Gaussians of kernel estimation.

In summary, the simplicity of PRISM the rules of Figure 10 allowed them to be explained to one focus group, all within a one hour session.

5. Discussion

5.1 Related Work

This research aims at producing a precise, explainable, operational definition of a runaway project. Other work in this area is less precise and not operational.

For example, in 1997, Glass [4] had informally sampled several high-profile software disasters and found the following features to be predictive for runaways:

- Project objectives not fully specified (in 51% of the sample);
- Bad planning and estimating (48%);
- Technology new to the organization (45%);
- Inadequate/no project management methodology (42%);
- Insufficient senior staff on the team (42%);
- Poor performance by suppliers of hardware/software (42%)
- Other-performance (efficiency) problems (42%)

Glass did not offer a clear operational method for combining their features into an effective predictor. Other work carefully documented the software risk problem, but did not offer automatic tool support:

- Jiang *et al.* [6] studied 40 features collected from questionnaires posted to personnel with recent experience with an IS project. Their study is an exemplary example of software engineering research: after clearly defined six hypotheses about software risk, they identify those hypotheses not supported by their data.
- Ropponen & Lyytinen [7] studied self-reported data from 83 project managers and 1,110 projects to find 26 software risk components: six scheduling and timing risks; four system functionality risks; three subcontracting risks; four requirements management risks; four resource usage and performance risks; and five personnel management risks.

Both reports have the same limitations: their conclusions contain a somewhat ill-defined and manual procedure for managers to explore the above risks. For example, both reports list risks and their weighted contribution to total risk. However, no combination rule is offered on how to best combine evidence of *multiple* risks.

Another aspect that sets this work apart from other

studies is *reproducibility*. Neither the Jiang *et al*. nor Ropponen & Lyytinen [7] studies are reproducible since they did not made their data available to other researchers. Reproducibility is an important methodological principle in other disciplines since it allows a community to confirm, refute, or even improve prior results. In our view, in the field of software engineering, there are all too few examples of reproduced, and extended, results[*]. This current report began when the second and third authors published their data [1] and defined a research challenge: how to better explain the results of their learning to developers [2]. We would strongly encourage software engineering researchers to share data, define challenges, and to take the time to rework the results of others.

5.2 External Validity

This study has produced:

1). A recommended *feature subset* for predicting runaways (*r*1,*p*6,*o*3);

2). A recommended *model* that combines those features (Figure 10); and

3). A recommended *method* for generating that subset and that model:

- 3*bin* discretization;
- a WRAPPER around PRISM;
- 10-way cross-validation using PRISM on the subsets found by WRAPPER.

It is good practice to question the external validity of these recommendations.

WRAPPER selected different features than the manual method that produced Equation 1. That is, the recommended feature subset learned by our recommended method is different to that found by our earlier work. This raises a concern about external validity: why do our conclusions keep changing?

We endorse the conclusions of this study over our prior work [1] for two reasons. Firstly, this study explored far more feature subsets that before:

- Equation 1 was generated after a manual analysis of a few features.
- Figure 10 was generated after an automatic search through thousands of subsets.

Secondly, the results of this study perform better than our prior results:

- Equation 1 offers ambiguous conclusions in the range ($0.03<P(runaway|X)<0.81$).
- Figure 10 offers categorical conclusions about the runaway status of a project. Further, it does so with perfect precision.

A more serious validity threat comes from the data used in this study. Any inductive process suffers from a *sampling bias*; i.e. the conclusions of the study are a function of the data used in that study. In that regard, we have evidence that our results are stable across small to medium-sized changes to our project sample. In a 10-way cross-validation experiment, 10% of the data (in our case, 3 to 4 records) is set aside and the model is learned from the remaining information. Our learned model had an average precision of 1.0 in a 10-way; i.e. the precision of our model remained perfect, despite a 10% change in the training data.

Also, Figure 1 does not show all the data available to this study. Some of the data available to this research group is proprietary and cannot be generally released. In order to check the external validity of our methods, these ten extra records were not analyzed until *after* we reached the above conclusions regarding the recommended data mining method for this data. When our recommended method was applied to Figure 1, plus the extra ten records, WRAPPER still found the features shown in Figure 9. Further, the performance of the rule set learned from the extended data had the same properties as Figure 10; i.e.

- It out-performed NäiveBayes;
- It exhibited perfect precision (precision=1.0) over the 10-way cross-validation.

In summary, despite the data set size changing by a small to medium amount (-10% to +33%), there is:

- No instability in the recommended features;
- No instability in the performance of the recommended model;
- No instability in the recommended method.

5.3 Method Selection for Quirky Data

Several times we found that certain widely regarded methods (RIPPER; discretization using Fayyad&Irani; discretization with Yang & Webb's $\sqrt{n}$ rule) did not yield the best results for this data set. The reason for this is simple: software engineering data sets are often small:

- Figure 1 is one table with only 22*33 cells;
- Elsewhere we have published results on even smaller data sets [41,42].

It is hard to know *apriori* what are the quirks of small software engineering data sets. Hence, we recommend trying many methods, even supposedly out-dated ones. For example, in this study, a very simple rule-learner (PRISM) produced the best performance while being most understandable to our users.

More generally, Fayyad [43] argues persuasively that data mining should be viewed as a *small part* of the knowledge and data discovery (KDD) cycle shown in Figure 11. For example, in this report we used discretization and feature subset-selection for *pre-processing* and *selection* steps shown in Figure 11. Also, we looped through the KDD cycle 13 times: each time, the results from the previous round informed our work for the next round.

[*]Exception: see the reports of the PROMISE workshop http://promisedata.org/repository/papers.html

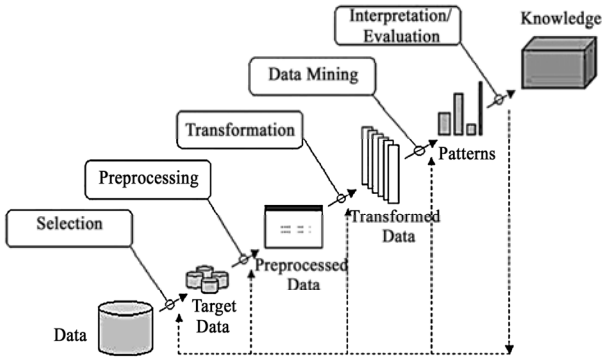


Figure 11. The KDD (Knowledge Discovery in Databases) cycle, adapted from [43]

5.4 Data Mining Methods

Based on this work, and certain standard texts in the data mining field [17,43], we offer the following advice to other researchers data mining on SE data.

It is important to understand the *goals* of the data mining task. If the learned model only needs to *perform*, and not *explain* then any data mining method might do ranging from

- Näive Bayes classifiers
- To clustering algorithms, decision tree learners, neural nets, etc
- Or, as explored in Equation 4, ensembles of the above.

The simplest of the above is Näive Bayes. Such classifiers scale to very large data sets and, in many domains, have performed very well [15,19,20]. Also, in at least one SE domain [21], they far out-performed other methods.

However, if the goal is to generate an *explainable* theory, then:

- Many business users do not have the background required to understand mathematical-based learners. For such users, the rule learners (e.g. RIPPER) may be most useful since they produce succinct summaries of the data.
- It is useful to reduce the range of number variables with discretization. Once reduced, the learned model can be simpler since it only needs to comment on a few discrete ranges rather than the entire number line.
- It is also useful to reduce the number of features with feature subset selection. A repeated result in the literature [19,39,40] is that the majority of the features can be pruned away and the resulting model is either simpler, performs better or both. For example, in this case study, the best performance and the most succinct/explainable model were found using just 3/22 of the available data.

As to the choice of feature subset selector:

- Hall and Holmes [19] compare WRAPPER to several other variable pruning methods including the principal component analysis (PCA) method used by Ropponen & Lyytinen and Munson [9] (amongst others). Feature selection methods can be grouped according to (a) whether or not they make special use of the target variable in the data set such as "runaway"; (b) whether or not pruning uses the target learner. PCA does not make special use of the target variable. Also, unlike other pruning methods, WRAPPER *does* use the target learner as part of its analysis. Hall and Holmes found that PCA was one of the worst performing methods (perhaps because it ignored the target variable) while WRAPPER was the best (since it can exploit its special knowledge of the target learner).
- For large data sets, WRAPPER can be too slow. When WRAPPER is not possible, see the conclusion of the Hall & Holmes study [19] for recommendations on two other feature subset selection methods.
- If the data set is small enough (e.g. Figure 1), use WRAPPER around a rule learner. WRAPPER is the slowest feature subset selector but it is the only one that can tune itself to the target learner.

Regarding performance measures, we have two recommendations:

- Comparing the *f*-measures in treatment *a* and *b* of Figure 7, it is clear that self-tests can over-estimate the value of a learned model. Hold-out sets are the recommended way to assess a learned model.
- Accuracy is a widely used measure for assessing a learned theory. Figure 7 shows that it can be remarkably uninformative. In that figure, large changes in precision and recall make very little impact on the accuracy. Hence, we strongly recommend against the use of accuracy.

The above issues are widely discussed in the data mining literature (e.g. [17,43–45]). Nevertheless, our reading of the literature is that multiple traversals of the KDD cyclic application using a range of techniques (e.g. different learners, discretizers, and feature subset selectors) is quite rare. Often researchers take one learner, apply it once, then report the conclusion. Also, despite many positive empirical studies, feature selection is rarely seen in software engineering (exceptions: [21,46]). Further, it is still standard practice for software engineers to present their data mining results in terms of accuracy of non-hold-out experiments (e.g. [47]). We hope our results encourage a change in that standard practice.

6. Conclusions

Intuitively, it seems reasonable that optimizing for performance can compromise explainability. Software engineering data can be complex, noisy, or confusing. Such complex data may require complex and arcane learning strategies; e.g. the defect data sets studied by Menzies,

Greenwald, and Frank. Complex and arcane learning strategies will be hard to explain. That is, good performance in a learned model may imply poor explanatory power, especially for real world software engineering data.

This paper is a counter-argument to such pessimism. We show that at least for predicting runaway software projects, certain standard data mining methods resulted in models with both:

- High performance: i.e. precision=1.0; and
- Good explainability: i.e. small rule sets, understandable by our users;

This result is a new high water mark in predicting runaway projects. This new predictor out-performs prior results in several ways:

- Our results are fully reproducible: the data for our analysis comes from Figure 7; the software used is freely available*.
- Prior work by other researchers [4–7] has carefully documented the influence of features on software risk, but did not offer an operational model (by "operational", we mean that the model can generate performance statistics like Figure 7).
- As to our own prior results, the logistic regression method [1] required some manual intervention on the part of the analyst. In contrast to that, the techniques described here are automatic. Also, due to ambiguities in the middle *P* ranges of Equation 1, or the inner complexities of our Näive Bayes classifier [2], our prior mathematical results were much harder to explain than the new rules of Figure 10.
- Comparing treatment *c* and treatment *m* in Figure 7, we see that our new data mining method (treatment *m*: 3*bin*, WRAPPER, PRISM) has similar recall but much higher precision than our old data mining method (treatment *c*: NäiveBayes [2]).
- Measured in terms of precision, this new model is as good as can ever be expected for our data. Other combination data mining methods could out-perform our result (e.g. by generating a smaller, more explainable model with higher recall) but no other method could be more precise (since precision's maximum value is 1.0).
- Prior results conducted a manual exploration of a few subsets of the features [1]. Here, we employed a feature subset selector that explored thousands of feature subsets. Hence, we have far more confidence that the following factors are most useful in recognizing runaways: ambiguous requirements; low morale; lack of project members' commitment to the project plan.

REFERENCES

[1] Y. Takagi, O. Mizuno, and T. Kikuno, "An empirical approach to characterizing risky software projects based on logistic regression analysis," Empirical Software Engineering, Vol. 10, No. 4, pp. 495–515, 2005.

[2] S. Abe, O. Mizuno, T. Kikuno, N. Kikuchi, and M. Hirayama, "Estimation of project success using bayesian classifier," in ICSE 2006, pp. 600–603, 2006.

[3] O. Mizuno, T. Kikuno, Y. Takagi, and K. Sakamoto, "Characterization of risky projects based on project managers evaluation," in ICSE 2000, 2000.

[4] R. Glass, "Software runaways: Lessons learned from massive software project failures," Pearson Education, 1997.

[5] "The Standish Group Report: Chaos 2001," 2001, http://standishgroup.com/sample research/PDFpages/ extreme chaos.pdf.

[6] J. Jiang, G. Klein, H. Chen, and L. Lin, "Reducing user-related risks during and prior to system development," International Journal of Project Management, Vol. 20, No. 7, pp. 507–515, October 2002.

[7] J. Ropponen and K. Lyytinen, "Components of software development risk: how to address them? A project manager survey," IEEE Transactions on Software Engineering, pp. 98–112, Feburary 2000.

[8] W. Dillon and M. Goldstein, "Multivariate analysis: Methods and applications." Wiley-Interscience, 1984.

[9] J. C. Munson and T. M. Khoshgoftaar, "The use of software complexity metrics in software reliability modeling," in Proceedings of the International Symposium on Software Reliability Engineering, Austin, TX, May 1991.

[10] G. Boetticher, T. Menzies, and T. Ostrand, "The PROMISE Repository of Empirical Software Engineering Data," 2007, http://promisedata.org/repository.

[11] T. McCabe, "A complexity measure," IEEE Transactions on Software Engineering, Vol. 2, No. 4, pp. 308–320, December 1976.

[12] M. Halstead, "Elements of software science," Elsevier, 1977.

[13] K. Toh, W. Yau, and X. Jiang, "A reduced multivariate polynomial model for multimodal biometrics and classifiers fusion," IEEE Transactions on Circuits and Systems for Video Technology, pp. 224–233, February 2004.

[14] R. Duda, P. Hart, and N. Nilsson, "Subjective bayesian methods for rule-based inference systems," in Technical Report 124, Artificial Intelligence Center, SRI International, 1976.

[15] P. Domingos and M. J. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," Machine Learning, Vol. 29, No. 2-3, pp. 103–130, 1997. http:// citeseer.ist.psu.edu/domingos97 optimality. html

[16] Y. Yang and G. Webb, "Weighted proportional k-interval discretization for naive-bayes classifiers," in Proceedings of the 7th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2003), 2003,

http://www.csse.monash.edu/_webb/Files/YangWe-bb03.pdf.

[17] I. H. Witten and E. Frank, Data mining. 2nd edition. Los Altos, Morgan Kaufmann, US, 2005.

[18] G. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," in Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence Montreal, Quebec: Morgan Kaufmann, 1995, pp. 338–345, http://citeseer.ist.psu.edu/john95 estimating.html.

[19] M. Hall and G. Holmes, "Benchmarking attribute selection techniques for discrete class data mining," IEEE Transactions On Knowledge And Data Engineering, Vol. 15, No. 6, pp. 1437–1447, 2003, http://www.cs.waikato.ac.nz/_mhall/HallHolmesTKDE.pdf.

[20] J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and unsupervised discretization of continuous features," in International Conference on Machine Learning, pp. 194–202, 1995, http://www.cs.pdx.edu/_timm/dm/dougherty95supervised.pdf.

[21] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," IEEE Transactions on Software Engineering, January 2007, http://menzies.us/pdf/06learnPredict.pdf.

[22] R. Quinlan, C4.5: Programs for Machine Learning. Morgan Kaufman, 1992.

[23] R. Holte, "Very simple classification rules perform well on most commonly used datasets," Machine Learning, Vol. 11, pp. 63, 1993.

[24] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "Classification and regression trees," Wadsworth International, Monterey, CA, Tech. Rep., 1984.

[25] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297, 1967.

[26] T. M. Cover and P. E. Hart, "Nearest neighbour pattern classification," IEEE Transactions on Information Theory, pp. 21–27, January 1967.

[27] A. Beygelzimer, S. Kakade, and J. Langford, "Cover trees for nearest neighbor," in ICML'06, 2006, http://hunch.net/_jl/projects/cover tree/cover tree.html.

[28] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," Science, No. 4598, Vol. 220, pp. 671–680, 1983, http://citeseer.nj.nec.com/kirkpatrick83opt-imization.html

[29] G. G. Towell and J. W. Shavlik, "Extracting refined rules from knowledge-based neural networks," Machine Learning, Vol. 13, pp. 71–101, 1993, http: //citeseer.ist.psu.edu/towell92extracting.html

[30] B. Taylor and M. Darrah, "Rule extraction as a formal method for the verification and validation of neural networks," in IJCNN '05: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, Vol. 5, pp. 2915–2920, 2005.

[31] T. Menzies and E. Sinsel, "Practical large scale what-if queries: Case studies with software risk assessment," in Proceedings ASE 2000, 2000, http://menzies.us/pdf/00ase.pdf.

[32] W. Cohen, "Fast effective rule induction," in ICML'95, 1995, pp. 115–123, http://www.cs.cmu.edu/_wcohen/postscript/ml-95-ripper.ps.

[33] J. Cendrowska, "Prism: An algorithm for inducing modular rules," International Journal of Man-Machine Studies, Vol. 27, No. 4, pp. 349–370, 1987.

[34] T. Dietterich, "Machine learning research: Four current directions," AI Magazine, Vol. 18, No. 4, pp. 97–136, 1997.

[35] T. Menzies and J. S. D. Stefano, "How good is your blind spot sampling policy?" in 2004 IEEE Conference on High Assurance Software Engineering, 2003, http://menzies.us/pdf/03blind.pdf.

[36] J. Lu, Y. Yang, and G. Webb, "Incremental discretization for naive-bayes classifier," in Lecture Notes in Computer Science 4093: Proceedings of the Second International Conference on Advanced Data Mining and Applications (ADMA 2006), pp. 223–238, 2006, http://www.csse.monash.edu/_webb/Files/LuYangWebb06.pdf.

[37] U. M. Fayyad and I. H. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, pp. 1022–1027, 1993.

[38] J. Gama and C. Pinto, "Discretization from data streams: Applications to histograms and data mining," in SAC '06: Proceedings of the 2006 ACM symposium on Applied computing. New York, NY, USA: ACM Press, pp. 662–667, 2006. http://www.liacc.up.pt/_jgama/ IWKDDS/Papers/p6.pdf.

[39] A. Miller, Subset Selection in Regression (second edition). Chapman & Hall, 2002.

[40] R. Kohavi and G. H. John, "Wrappers for feature subset selection," Artificial Intelligence, Vol. 97, No. 1-2, pp. 273–324, 1997, http://citeseer.nj.nec.com/ kohavi96wrappers.html

[41] T. Menzies and J. D. Stefano, "More success and failure factors in software reuse," IEEE Transactions on Software Engineering, May 2003, http://men- zies.us/pdf/02sereuse.pdf.

[42] T. Menzies, Z. Chen, J. Hihn, and K. Lum, "Selecting best practices for effort estimation," IEEE Transactions on Software Engineering, November 2006, http://menzies.us/pdf/06coseekmo.pdf.

[43] U. Fayyad, "Data mining and knowledge discovery in databases: Implications for scientific databases," in Proceedings on Ninth International Conference on Scientific and Statistical Database Management, pp. 2–11, 1997.

[44] F. Provost, T. Fawcett, and R. Kohavi, "The case against accuracy estimation for comparing induction

algorithms," in Proc. 15th International Conf. on Machine Learning. Morgan Kaufmann, San Francisco, CA, pp. 445–453, 1998, http://citeseer.nj.nec.com/ provost98case.html.

[45] R. Bouckaert, "Choosing between two learning algorithms based on calibrated tests," in ICML'03, 2003, http://www.cs.pdx.edu/_timm/dm/10x 10way.

[46] C. Kirsopp and M. Shepperd, "Case and feature subset selection in case-based software project effort prediction," in Proc. of 22nd SGAI International Conference on Knowledge-Based Systems and Applied Artificial Intelligence, Cambridge, UK, 2002.

[47] N. Nagappan and T. Ball, "Static analysis tools as early indicators of pre-release defect density," in ICSE 2005, St. Louis, 2005.

J. Software Engineering & Applications, 2009, 2: 237-247
doi:10.4236/jsea.2009.24031 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

A New Interactive Method to Solve Multiobjective Linear Programming Problems

Mahmood REZAEI SADRABADI[1], Seyed Jafar SADJADI[2]

[1]Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, Netherlands; [2]Department of Industrial Engineering, Iran University of Science & Technology, Tehran, Iran.
Email: m.rezaei.sadrabadi@student.tue.nl

Received May 20th, 2009; revised June 19th, 2009; accepted June 29th, 2009.

ABSTRACT

Multiobjective Programming (MOP) has become famous among many researchers due to more practical and realistic applications. A lot of methods have been proposed especially during the past four decades. In this paper, we develop a new algorithm based on a new approach to solve MOP by starting from a utopian point, which is usually infeasible, and moving towards the feasible region via stepwise movements and a simple continuous interaction with decision maker. We consider the case where all objective functions and constraints are linear. The implementation of the proposed algorithm is demonstrated by two numerical examples.

Keywords*: *Multiobjective Linear Programming, Multiobjective Decision Making, Interactive Methods

1. Introduction

During the past four decades, many methods and algorithms have been developed to solve Multiobjective Programming (MOP), in which some objectives are conflicting and the utility function of the Decision Maker (DM) is imprecise in nature. MOP is believed to be one of the fastest growing areas in management science and operations research, in that many decision making problems can be formulated in this domain. For some engineering applications of MOP problems the interested reader is referred to [1,2]. Decision making problems with several conflicting objectives are common in practice. Hence, for such problems, a single objective function is not sufficient to seek the real desired solution. Because of this limitation, an MOP method is needed to solve many real world optimization problems [3].

Although different solution procedures have been introduced, the interactive approaches are generally believed to be the most promising ones, in which the preferred information of the DM is progressively articulated during the solution process and is incorporated into it [4]. The purpose of MOP in the mathematical programming framework is to optimize *r* different objective functions, subject to a set of systematic constraints. A mathematical formulation of an MOP is also known as the vector maximization (or minimization) problem. Generally, MOP can be divided into four different categories.

The first and the oldest group of MOP need not to get any information from DM during the process of finding an efficient solution. These types of algorithms rely solely on the pre-assumptions about DM's preferences. In this category, L-P Metric methods are noticeable, algorithms whose objectives are minimization of deviations of the objective functions from the ideal solution. Since different objectives have different scales, they must be normalized before the process of minimization of deviations starts. Therefore, a new problem is minimized which has no scale [5].

The second group of MOP includes gathering cardinal or ordinal preferred information before the solving process initiates. In the method of utility function [6], for example, we determine DM's utility as a function of objective functions and then we maximize the overall function under the initial constraints. The other method in this group, which is extensively used by many researchers, is Goal Programming (GP) [7] in which DM determines the least (the most) acceptable level of Max (Min) functions. Since attaining these values might lead to an infeasible point, the constraints are allowed to exceed, but we try to minimize these weighted deviations.

The third group of MOP provides a set of efficient solutions in which DM has the opportunity to choose his preferred solution among the efficient ones. Although finding an efficient solution in MOP is not difficult, but finding all efficient solutions to render DM is not a trivial task. Many papers have discussed this important issue [8–11]. The set of all efficient feasible solutions in a Multiobjective Linear Programming (MOLP) can be represented by convex combination of efficient extreme

points and efficient extreme rays in the feasible region. Therefore, the set of efficient extreme points and efficient extreme rays can be regarded as the solution set for an MOLP problem. Ida [8] develops an algorithm to find the structure of efficient solutions in an MOLP using all efficient extreme points and extreme rays. Pourkarimi et al. [9] represent the structure of efficient solutions as maximal efficient faces. Youness and Emam [11] investigate the relationships among some efficient solutions in the objective space and then obtain all efficient solutions of the MOLP and their structure in the constraint space. Steuer and Piercy [10] use regression analysis to estimate the number of efficient extreme points in MOLP. Even though the final solution among the efficient solutions is usually chosen by DM, but Gass and Roy [12] propose a mathematical method for ranking the set of efficient extreme solutions in an MOLP. The idea is to enclose the given efficient solutions within an annulus of minimum width, where the width is determined by a hypersphere that minimizes the maximum deviation of the points from the surface of the hypersphere.

Finally, the last group of MOP problems provides solutions based on a continuous interaction with DM and tries to reach the preferred solution at the end of the algorithm. Based on this sound idea, there are many developed methods categorized in this group [13–21]. Different procedures may be better suited for different types of decision makers, for different types of decision situations, or for different stages in the decision making process [22]. Homburg [16], for instance, proposed a hierarchical procedure which consists of two levels, a top-level and a base-level. The main idea is that the top-level only provides general preference information from DM. Taking this information into account, the base-level then determines a compromise solution via interaction with DM by using an interactive procedure. As another example, Tchebycheff metric based approaches have become popular in this category for sampling the set of efficient solutions in a continuous interaction with DM to narrow his choices down to a single most preferred efficient solution. The interaction with DM proceeds by generating smaller subsets of the efficient set until a final solution is located [19]. The proposed method by Engau [14] decomposes the original MOP problem into a collection of smaller-sized subproblems to facilitate the evaluation of tradeoffs and the articulation of preferences. A priori preferences on objective tradeoffs are integrated into this process, and DM is supported by an interactive procedure to coordinate any remaining tradeoffs.

There are many advantages on using interactive methods such as:

- There is no need to get any information from DM before the solving process initiates,
- The solving process helps DM learn more about the nature of the problem,
- Only minor preferred information are needed during the solving process,
- Since DM continuously contributes via analyst to the problem, he is more likely to accept the final solution,
- There are fewer restricting assumptions involved in these types of problems in comparison with other groups of MOP methods.

However, there are some drawbacks associated with these types of algorithms that the most important ones are as follows:

- The accuracy of the final solution depends entirely upon DM's precise answers. In other words, if DM does not carefully interact with the analyst, the outcome(s) of the final solution may be undesirable,
- There is no guarantee to reach a desirable solution after a finite number of iterations,
- DM needs to make more effort during the process of these algorithms in comparison with other groups.

During the past decades, many researchers have tried to review or to discuss the strengths, the weaknesses, and the comparative studies on the existing methods. Borges and Antunes [23] deal with the sensitivity analysis of the weights in MOLP. Buchanan [24] has an excellent paper that reviews and comments on ten famous methods including [25–27]. Each description is followed by a detailed analysis of the method which consists of comments about the underlying approach, technical aspects and practical considerations. All methods are compared in terms of some important features such as applicability, convergence, and difficulty of questions. Also, Buchanan and Daellenbach [24] describe a laboratory experiment which compares the performance from the user's point of view of four different methods for MOP problems. Reeves and Gonzalez [22] compare the computational performance and the quality of the solutions generated by two similar and yet contrasting interactive procedures. Sun [4] investigates the solution quality in interactive MOP. They include value functions used, weights assigned to the objective functions in the value functions, the size of the efficient set, and the number of objective functions. The feasibility and existence of the ideal and nadir points are also discussed. The work of Vanderpooten [28] is a very good reference to review the main concepts in interactive procedures. After briefly introducing the interactive procedures, a general technical framework for the understanding of existing methods is presented.

The main goals of the mentioned papers are to introduce some criteria to measure the efficiency of various algorithms and to introduce the characteristics of a good method. According to Reeves and Franz [29], the main characteristics of a proper interactive algorithm can be numerated as follows:

1) Minimum amount of information be required from DM,

2) The nature of decision making be simple,

3) If DM provides his answers improperly in some interactions, he can have the opportunity to compensate it in the following interactions,

4) The number of iterations to reach the final solution be reasonable,

5) DM be familiar with the nature of judgments he is asked for,

6) The algorithm be suitable for solving large scale problems.

In this paper, we propose a new algorithm which is mainly in the group of interactive methods. However, we also need to get some information from DM before problem solving initiates; therefore, this algorithm is neither a pure interactive method nor a pure method in the second category. In addition, the proposed algorithm is based upon a novel approach to the problem, starting from an infeasible utopian point and moving towards the feasible region and then the final efficient point. The remaining of this paper is organized as follows. Section 2 provides some of the necessary definitions we need to use in this paper. In section 3, the problem statement and the proposed algorithm are explained. Two numerical examples are demonstrated in section 4 to illustrate the proposed algorithm. Finally, conclusion remarks appear in section 5 to summarize the contribution of the paper.

2. Definitions

Consider an MOLP problem defined as follows,

$$\begin{aligned} &\max\{Z = f_k(X) = C_k^T X; k = 1,2,...,r\} \\ &s.t. \\ &M = \{X \in R^n \mid A_i X \le b_i; X \ge 0; i = 1,2,...,m\} \end{aligned} \quad (1)$$

where,

$f_k(X)$: is the kth objective function,

C_k: is the vector of coefficients in the kth objective function,

X: is an n-dimensional vector of decision variables,

A_i: is the ith row of technological coefficients,

b_i: is the RHS of the ith constraint, and

M: is the feasible region.

A solution $\overline{X} \in M$ is *efficient* if and only if there does not exist another $X \in M$ such that $f_k(X) \ge f_k(\overline{X})$ for all $k = 1,2,...,r$ and $f_k(X) > f_k(\overline{X})$ for at least one k. Then, the vector,

$$\overline{Z} = \{f_k(\overline{X}); k = 1,2,...,r\} \quad (2)$$

is called a *non-dominated criterion vector*. All efficient solutions in M form the efficient set E. Although some interactive algorithms search the entire feasible region M, the majority of them are designed to search only the efficient set E. The vector,

$$Z^* = \{f_k(X^*) \mid f_k(X^*) = \max f_k(X); k = 1,2,...,r\} \quad (3)$$

is called the *ideal point* or the *ideal criterion vector*. It should be mentioned that the ideal criterion vector, and so the *ideal solution* X^*, does not usually exist. The vector,

$$Z^{**} = \{f_k(X^{**}) \mid f_k(X^{**}) \ge \max f_k(X); k = 1,2,...,r\} \quad (4)$$

is called a *utopian vector* or a *utopian point*. Unlike the ideal criterion vector, there exist many utopian vectors. Nevertheless, their corresponding X^{**}'s are most likely infeasible.

3. Problem Statement

The majority of methods proposed in the domain of interactive procedures search the feasible region M or the efficient set E through interaction with DM in order to attain the final solution. Here, we develop a new algorithm to solve MOLP problems by starting from a utopian point X^{**} (which is usually infeasible) and moving towards the feasible region M and then the efficient set E via stepwise movements and a plain continuous interaction with DM in order to be in line with his preferences. Since there are many utopian points outside the M, we choose the closest X^{**} to M as the start point, by considering the least sum of weighted deviations from the constraints.

3.1 The Proposed Algorithm

The proposed algorithm attains an efficient solution of an MOLP through the following steps:

Algorithm:

Step 1. Ask DM to determine a_k, the maximum acceptable reduction in the amount of f_k in any interaction. Also, ask him to determine w_i, a penalty for deviation of each unit from the ith constraint. In the next step, we find a utopian point allowing some deviations from the constraints $x_j \ge 0$, in that the utopian point maybe a point with some negative x_j's. However, we also consider a big penalty, w', for each unit of such deviations.

Step 2. Maximize each $f_k(X)$ with consideration of the feasible set M as follows,

$$\begin{aligned} &\max f_k(X) = C_k^T X \\ &s.t. \\ &M = \{X \in R^n \mid A_i X \le b_i; X \ge 0; i = 1,2,...,m\} \end{aligned} \quad (5)$$

Step 3. Let $f_k(X^*)$ be the optimal solution for

each $f_k(X); k=1,2,...,r$. Solve the following GP problem,

$$\begin{aligned}&\min\{\sum_{i=1}^{m} w_i d_i + w'\sum_{j=1}^{n} d'_j \mid f_k(X) \geq f_k(X^*);\\ &A_i X \leq b_i + d_i; x_j \geq -d'_j; d \geq 0;\\ &i=1,2,...,m; j=1,2,...,n; k=1,2,...,r\}\end{aligned} \quad (6)$$

where, d_i represents the deviation from the *i*th constraint. In this step, we allow our solution to go outside the feasible region. Suppose *X* is the solution for (6). Set $X^0 = X$ and go to step 4.

Step 4. Let θ_{ik} be the angle between f_i and f_k. Calculate $\sin\theta_{ik}$ as follows,

$$\sin\theta_{ik} = \sqrt{1 - \frac{C_i.C_k}{|C_i|.|C_k|}} \quad (7)$$

Now, we can determine a small step δ by which we move towards the feasible region in each iteration as,

$$\delta = \min\{\frac{a_i}{|C_i|\sin\theta_{ik}}; i,k=1,2,...,r; i \neq k\} \quad (8)$$

Step 5. Consider constraints $f_k(X^0) \geq f_k(X^*)$ which remain active. Now ask DM to see which active objective has the least desirability. Let *l* be the index for the f_l which has the least desirability.

Step 6. Solve the following optimization problem in which we take a step δ from X^0 towards the feasible region while we keep the amount of f_l,

$$\begin{aligned}&\min\{\sum_{i=1}^{m} w_i d_i + w'\sum_{j=1}^{n} d'_j \mid f_l(X) \geq f_l(X^0);\\ &A_i X \leq b_i + d_i; \mid X - X^0 \mid = \delta; x_j \geq -d'_j; d \geq 0;\\ &i=1,2,...,m; j=1,2,...,n\}\end{aligned} \quad (9)$$

where, |.| is the 2-norm. In this step there is no change in the value of f_l but we usually expect that the other objective functions get worse, but not necessarily. In other words, we might encounter a situation in which the values of some active or inactive f_k get better.

Step 7. If $\sum_{i=1}^{m} w_i d_i + w'\sum_{j=1}^{n} d'_j = 0$ then go to step 8, otherwise set $X^0 = X$, calculate the new values of $f_k(X^0)$, and go to step 5.

Step 8. $\sum_{i=1}^{m} w_i d_i + w'\sum_{j=1}^{n} d'_j = 0$ implies that we are inside the feasible region, but most likely not on the boundary. Therefore, we take a smaller step to be stopped on the boundary by solving,

$$\begin{aligned}&\min\{\mid X - X^0 \mid \mid f_k(X) \geq f_k(X^0); A_i X \leq b_i; x_j \geq 0;\\ &i=1,2,...,m; j=1,2,...,n; k=1,2,...,r\}\end{aligned} \quad (10)$$

There is no guarantee that the solution of step 8 is a non-dominated one. So, we move on the boundary to reach a non-dominated solution. Set $X^0 = X$, $S = \{1,2,...,r\}$, and go to step 9.

Step 9. Ask DM to see which objective in *S* has the least desirability. Let *l* be the index for the f_l which has the least desirability. Solve the following optimization problem in which we take a step at most with the amount of δ from X^0 on the boundary of the feasible region while we keep the amounts of $f_k; k=1,2,...,r$,

$$\begin{aligned}&\max\{f_l(X) \mid f_k(X) \geq f_k(X^0); A_i X \leq b_i; \mid X - X^0 \mid \leq \delta;\\ &x_j \geq 0; i=1,2,...,m; j=1,2,...,n; k=1,2,...,r\}\end{aligned} \quad (11)$$

Step 10. If $f_l(X) > f_l(X^0)$ then set $S = \{1,2,...,r\}$ and go to step 9, otherwise set $S = S - l$ and go to step 11.

Step 11. If $S = \phi$ then choose *X* as the final efficient solution, otherwise set $X^0 = X$ and go to step 9.

End.

It should be noted that steps 1-8 help us to reach to the feasible region *M* by starting from the closest utopian point in line with DM's preferences, whereas steps 9-11 guarantee that the final solution is an efficient one, i.e., the final solution is in *E*.

3.2 Some Lemmas to Determine δ

Here, we show how to choose δ in Step 4 of the proposed algorithm with the following three lemmas.

Lemma 1: Any step δ along gradient vector C_k will result a decrease (or increase) of $\delta \mid C_k \mid$ in f_k.

Proof: Let α_{kj} be the angle between C_k and axis x_j. Therefore,

$$\begin{aligned}\cos\alpha_{kj} &= \frac{C_k.x_j}{|C_k||x_j|} = \frac{(c_{k1},...,c_{kj},...,c_{kn}).(0,...,1,...,0)}{|C_k|\times 1}\\ &= \frac{c_{kj}}{|C_k|}\end{aligned} \quad (12)$$

where, x_j is the *j*th unique vector in an n-dimensional space. The angle between C_k and x_j helps us to compute the projection of C_k over the axis x_j, i.e.,

if we take a step δ along vector C_k, the amount of change in each element of x_j is $\delta\cos\alpha_{kj}$ or $\delta\cos(\pi-\alpha_{kj})$ depending on the direction we choose. Figure 1 depicts the gradient vector C_k and its projection in a 2-dimensional space.

Therefore,

$$\Delta x_j = \delta\cos\alpha_{kj} = \delta\frac{c_{kj}}{|C_k|} \tag{13}$$

or

$$\Delta x_j = \delta\cos(\pi-\alpha_{kj}) = -\delta\cos\alpha_{kj} = -\delta\frac{c_{kj}}{|C_k|} \tag{14}$$

Therefore, we can compute the change in the amount of f_k as follows,

$$|\Delta f_k| = \sum_{j=1}^{n} c_{kj}.\Delta x_j = \sum_{j=1}^{n} c_{kj}.\delta\frac{c_{kj}}{|C_k|} = \frac{\delta}{|C_k|}\sum_{j=1}^{n} c_{kj}^2 = \frac{\delta}{|C_k|}.|C_k|^2 = \delta.|C_k| \tag{15}$$

We now present a generalized form of Lemma (1).

Lemma 2: Any step δ along C_l which makes the angle θ_{lk} with C_k will result a decrease (increase) of $\delta|C_k|\cos\theta_{lk}$ in f_k.

Proof: It is clear that taking a step δ along C_l which makes the angle θ_{lk} with C_k is the same as taking a step $\delta\cos\theta_{lk}$ along C_k. Using the results of Lemma(1) yields,

$$|\Delta f_k| = \delta\cos\theta_{lk}\,|C_k| \tag{16}$$

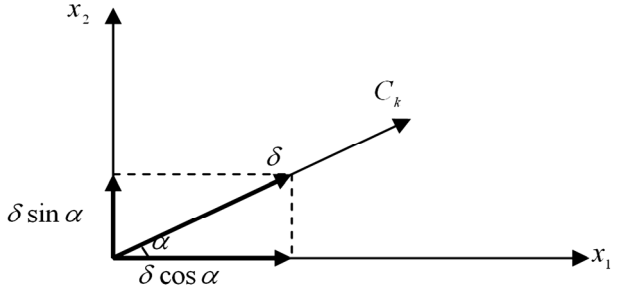


Figure 1. The gradient vector C_k and its projection

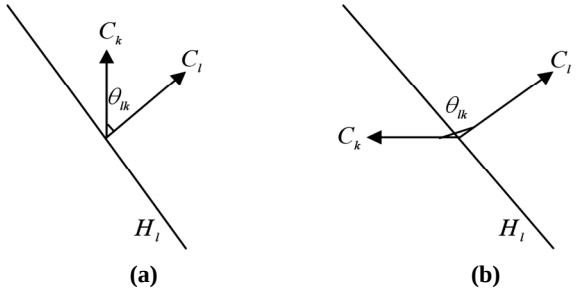


Figure 2. Demonstration of taking a step δ on H_l in a 2-dimensional space

Lemma 3: Let H_l be a hyperplane which is orthogonal to C_l and C_l makes the angle θ_{lk} with C_k. Any step δ on the hyperplane H_l in any direction will result a decrease (increase) of $\delta\sin\theta_{lk}\,|C_k|$ in f_k.

Proof: We prove this lemma in two steps. In the first step, let $0 \le \theta_{lk} \le \pi/2$, then taking any step δ on H_l in any direction is the same as taking a step δ in the direction whose angle with C_l is $\pi/2$ and therefore makes the angle $\pi/2 \pm \theta_{lk}$ with C_k. Figure 2(a) demonstrates the situation in a 2-dimensional space.

According to lemma (2), taking any step δ along the direction which makes the angle $\pi/2+\theta_{lk}$ or $\pi/2-\theta_{lk}$ with C_k will result a change with the amount of $\delta\cos(\pi/2+\theta_{lk})\,|C_k|$ or $\delta\cos(\pi/2-\theta_{lk})\,|C_k|$ in f_k. Since $0 \le \theta_{lk} \le \pi/2$, we have,

$$\delta\cos(\pi/2+\theta_{lk})\,|C_k| = -\delta\sin\theta_{lk}\,|C_k| \tag{17}$$

or

$$\delta\cos(\pi/2-\theta_{lk})\,|C_k| = \delta\sin\theta_{lk}\,|C_k| \tag{18}$$

Finally, we have,

$$|\Delta f_k| = \delta\sin\theta_{lk}\,|C_k| \tag{19}$$

Now, in the second step, suppose $\pi/2 \le \theta_{lk} \le \pi$. Taking any step δ on H_l in any direction is the same as taking a step δ in the direction whose angle with C_k is $\theta_{lk}-\pi/2$ or $3\pi/2-\theta_{lk}$. Figure 2(b) demonstrates the situation in a 2-dimensional space. Using similar argument used in the first step yields,

$$\delta\cos(\theta_{lk}-\pi/2)\,|C_k| = \delta\sin\theta_{lk}\,|C_k| \tag{20}$$

or

$$\delta\cos(3\pi/2-\theta_{lk})\,|C_k| = -\delta\sin\theta_{lk}\,|C_k| \tag{21}$$

Finally, we have,

$$|\Delta f_k| = \delta\sin\theta_{lk}\,|C_k| \tag{22}$$

Now, we are ready to determine the amount of δ properly. Suppose DM determines that he wouldn't expect any reduction more than a_k in the amount of f_k in any interaction. When we perform step (4) in the algorithm, actually we keep f_l unchanged. In order to

achieve this goal, we have to take step δ on H_l. According to lemma (3), the step leads to an increase (decrease) $\delta \sin\theta_{lk} \mid C_k \mid$ in f_k. There is no problem in our approach in case f_k increases. However, we must ensure that the step δ would not worsen f_k more than a_k, which suggest to keep the following condition,

$$\delta \sin\theta_{lk} \mid C_k \mid \leq a_k; k = 1,...,r; k \neq l \tag{23}$$

or

$$\delta \leq \frac{a_k}{\mid C_k \mid \sin\theta_{lk}}; k = 1,...,r; k \neq l \tag{24}$$

Holding (19) in all interactions throughout the algorithm guarantees that there would be no reduction in any $f_k; k \neq l$ more than a_k. Since DM is entitled to keep the amount of any f_k, the following condition must hold in order to obtain an appropriate δ,

$$\delta \leq \frac{a_k}{\mid C_k \mid \sin\theta_{lk}}; k,l = 1,...,r; k \neq l \tag{25}$$

Finally, we are about to determine the best amount of δ with consideration of DM's intentions and concurrently reaching to the feasible solution by doing as fewer interactions as possible. Thus, we have,

$$\delta = \min\{\frac{a_k}{\mid C_k \mid \sin\theta_{lk}}; k,l = 1,...,r; k \neq l\} \tag{26}$$

4. Numerical Examples

In this section we demonstrate implementation of the proposed method using two numerical examples.

4.1 Example 1

Consider the following MOLP problem with two objective functions,

$$\begin{aligned} &Maxz_1 = x_1 + 6x_2 \\ &Maxz_2 = 5x_1 + 2x_2 \\ &s.t. \\ &-x_1 + 4x_2 \leq 20 \\ &7x_1 + 9x_2 \leq 63 \\ &22x_1 + 15x_2 \leq 165 \\ &x_1 \leq 6.5 \\ &x_1, x_2 \geq 0 \end{aligned} \tag{27}$$

We first ask DM to specify his sensitivity about the constraints and the objectives. As we already defined, w_i's are the penalties associated with the constraints and a_k's are the permitted amounts of reduction on the objective functions in each iteration. For the sake of simplicity suppose that all constraints have equal sensitivity, i.e., $w_i = 1; i = 1,...,4$. Next, we have to determine the acceptable amount of reduction on the objectives z_1 and z_2. For this example, suppose DM specifies 2 and 3 for a_1 and a_2, respectively. The appropriate value for δ can be determined as the following,

$$C_1 = (1,6) \Rightarrow \mid C_1 \mid = \sqrt{1^2 + 6^2} = \sqrt{37}$$

$$C_2 = (5,2) \Rightarrow \mid C_2 \mid = \sqrt{5^2 + 2^2} = \sqrt{29}$$

$$\cos\theta_{12} = \frac{C_1.C_2}{\mid C_1 \mid . \mid C_2 \mid} = \frac{(1,6).(5,2)}{\sqrt{37}.\sqrt{29}} = 0.52$$

$$\sin\theta_{12} = \sqrt{1-(0.52)^2} = 0.85$$

$$\delta = \min\{\frac{a_1}{\mid C_1 \mid \sin\theta_{12}}, \frac{a_2}{\mid C_2 \mid \sin\theta_{21}}\}$$

$$= \min\{\frac{2}{\sqrt{37}(0.85)}, \frac{3}{\sqrt{29}(0.85)}\} = 0.38$$

Then, we must find z_1^* and z_2^*. Solving two distinct LP problems with consideration of z_1 and z_2 yields $(x_1, x_2) = (1.95,5.49)$ with $z_1^* = 34.86$ and $(x_1, x_2) = (6.50,1.47)$ with $z_2^* = 35.43$, respectively. In the next step, we obtain the utopian point in which both objectives are satisfied at least with their optimal values, while we reach to a common point. Hence, we have,

$$\begin{aligned} &MinD = d_1 + d_2 + d_3 + d_4 + 1000(d_5 + d_6) \\ &s.t. \\ &-x_1 + 4x_2 \leq 20 + d_1 \\ &7x_1 + 9x_2 \leq 63 + d_2 \\ &22x_1 + 15x_2 \leq 165 + d_3 \\ &x_1 \leq 6.5 + d_4 \\ &x_1 + 6x_2 \geq 34.86 \\ &5x_1 + 2x_2 \geq 35.43 \\ &x_1 \geq -d_5 \\ &x_2 \geq -d_6 \\ &x_1, x_2 : free\ in\ sign \\ &d_i \geq 0; i = 1,...,6 \end{aligned} \tag{28}$$

The optimal solution for (28) is $(x_1^{**}, x_2^{**}) = (5.10,4.96)$ with $(z_1^{**}, z_2^{**}) = (34.86,35.43)$ and $D^{**} = 39.02$. In the next step, the DM is asked to select the objective which has the least desirability for him. Suppose in the first interaction the DM adopts z_2. Therefore, we must solve the following problem,

$$
\begin{aligned}
&MinD = d_1 + d_2 + d_3 + d_4 + 1000(d_5 + d_6)\\
&s.t.\\
&-x_1 + 4x_2 \le 20 + d_1\\
&7x_1 + 9x_2 \le 63 + d_2\\
&22x_1 + 15x_2 \le 165 + d_3\\
&x_1 \le 6.5 + d_4\\
&x_1 \ge -d_5\\
&x_2 \ge -d_6\\
&5x_1 + 2x_2 \ge 35.43\\
&\sqrt{(x_1 - 5.10)^2 + (x_2 - 4.96)^2} = 0.38\\
&x_1, x_2 : \textit{free in sign}\\
&d_i \ge 0; i = 1,...,6
\end{aligned}
\tag{29}
$$

The optimal solution for (29) is $(x_1, x_2) = (5.24,4.61)$ with $(z_2, z_2) = (32.89,35.43)$ and $D = 34.65$. Table 1 summarizes the results of implementation of the proposed algorithm during the next iterations.

As one can observe, we have reached to the feasible region after 8 iterations. The final step by which we reach to the feasible region is from $(x_1, x_2) =$ (4.04, 4.10) to $(x_1, x_2) = (4.18,3.75)$ with feasible amounts $(z_1, z_2) = (26.65,28.38)$. So, in order to reach to the feasible region by a smaller step we solve,

Table 1. The detailed information for implementation of the proposed method for example 1

Iteration	Objective	x_1	x_2	D	z_1	z_2
0	max z_1	1.95	5.49	0	34.86	20.73
0	max z_2	6.5	1.47	0	15.32	35.43
0	utopian	5.1	4.96	39.02	34.86	35.43
1	keep z_2	5.24	4.61	34.65	32.89	35.43
2	keep z_2	5.38	4.25	30.27	30.88	35.43
3	keep z_1	5.01	4.32	20.9	30.88	33.69
4	keep z_2	5.17	3.91	15.87	28.63	33.69
5	keep z_1	4.8	3.97	6.5	28.63	31.94
6	keep z_1	4.42	4.04	4.28	28.63	30.18
7	keep z_1	4.04	4.1	2.14	28.63	28.38
8	keep z_2	4.18	3.75	0	26.65	28.38
9	min delta	4.17	3.75	0	26.69	28.38
10	max z_1	4.17	3.75	0	26.69	28.38
11	max z_2	4.17	3.75	0	26.69	28.38

$$
\begin{aligned}
&MinD = \sqrt{(x_1 - 4.04)^2 + (x_2 - 4.10)^2}\\
&s.t.\\
&x_1 + 6x_2 \ge 26.65\\
&5x_1 + 2x_2 \ge 28.38\\
&-x_1 + 4x_2 \le 20\\
&7x_1 + 9x_2 \le 63\\
&22x_1 + 15x_2 \le 165\\
&x_1 \le 6.5\\
&x_j \ge 0; j = 1,2
\end{aligned}
\tag{30}
$$

Problem (30) leads to $(x_1, x_2) = (4.17,3.75)$, with $(z_1, z_2) = (26.69,28.38)$ and $\delta = 0.37$, which is the first feasible point on the boundary of the feasible region. Then, the DM is asked to determine the objective function which has the least desirability. Suppose he adopts z_1, so we solve,

$$
\begin{aligned}
&Maxz_1 = x_1 + 6x_2\\
&s.t.\\
&\sqrt{(x_1 - 4.17)^2 + (x_2 - 3.75)^2} \le 0.38\\
&5x_1 + 2x_2 \ge 28.38\\
&-x_1 + 4x_2 \le 20\\
&7x_1 + 9x_2 \le 63\\
&22x_1 + 15x_2 \le 165\\
&x_1 \le 6.5\\
&x_j \ge 0; j = 1,2
\end{aligned}
\tag{31}
$$

Problem (31) leads to $(x_1, x_2) = (4.17,3.75)$ with $(z_1, z_2) = (26.69,28.38)$. As one can see, z_1 cannot be improved by moving from $(x_1, x_2) = (4.17,3.75)$. So, we have $S = \{2\}$ and z_2 is chosen to get improved. We solve,

$$
\begin{aligned}
&Maxz_2 = 5x_1 + 2x_2\\
&s.t.\\
&\sqrt{(x_1 - 4.17)^2 + (x_2 - 3.75)^2} \le 0.38\\
&x_1 + 6x_2 \ge 26.69\\
&-x_1 + 4x_2 \le 20\\
&7x_1 + 9x_2 \le 63\\
&22x_1 + 15x_2 \le 165\\
&x_1 \le 6.5\\
&x_j \ge 0; j = 1,2
\end{aligned}
\tag{32}
$$

Problem (32) leads to $(x_1, x_2) = (4.17,3.75)$ with $(z_1, z_2) = (26.69,28.38)$. As one can see, z_2 cannot be improved by moving from $(x_1, x_2) = (4.17,3.75)$. So,

$S=\phi$ and $(x_1,x_2)=(4.17,3.75)$ with $(z_1,z_2)=(26.69,28.38)$ is the final efficient feasible solution.

4.2 Example 2

Consider the following MOLP problem with three objective functions,

$$\begin{aligned}
&Maxz_1 = 10x_1 + 80x_2 + 25x_3 + 16x_4\\
&Maxz_2 = 6x_1 + 7x_2 + 25x_3 + 8x_4\\
&Maxz_3 = 8x_1 - 5x_2 + 12x_3 + 4x_4\\
&s.t.\\
&-6x_1 + 7x_2 + 5x_3 + 3x_4 \le 142\\
&2x_1 + 7x_2 + 25x_3 + 9x_4 \le 320\\
&20x_1 + 13x_2 + 40x_3 + 16x_4 \le 800\\
&3x_1 - 10x_2 + x_3 - 24x_4 \le 15\\
&16x_1 + 5x_2 - 2x_3 + 80x_4 \ge 228\\
&x_1,..., x_4 \ge 0
\end{aligned} \tag{33}$$

Suppose that the values 12, 5, 45, 2, and 6 are specified by the DM for w_1, w_2, w_3, w_4, and w_5, respectively and we consider $w'=1000$. Also, 300, 50, and 30 are determined as the acceptable amount of reduction for z_1, z_2, and z_3. The optimal value for δ is determined as follows,

$$C_1=(10,80,25,15)\Rightarrow|C_1|=\sqrt{7350}$$

$$C_2=(6,7,25,8)\Rightarrow|C_2|=\sqrt{774}$$

$$C_3=(8,-5,12,4)\Rightarrow|C_2|=\sqrt{249}$$

$$\cos\theta_{12}=\frac{C_1.C_2}{|C_1|.|C_2|}=\frac{(10,80,25,15).(6,7,25,8)}{\sqrt{7350}.\sqrt{774}}=0.57$$

$$\Rightarrow\sin\theta_{12}=\sqrt{1-(0.57)^2}=0.82$$

$$\cos\theta_{13}=\frac{C_1.C_3}{|C_1|.|C_3|}=\frac{(10,80,25,15).(8,-5,12,4)}{\sqrt{7350}.\sqrt{249}}=0.03$$

$$\Rightarrow\sin\theta_{13}=\sqrt{1-(0.03)^2}=1$$

$$\cos\theta_{23}=\frac{C_2.C_3}{|C_2|.|C_3|}=\frac{(6,7,25,8).(8,-5,12,4)}{\sqrt{774}.\sqrt{249}}=0.79$$

$$\Rightarrow\sin\theta_{23}=\sqrt{1-(0.79)^2}=0.62$$

$$\delta=\min\{\frac{a_1}{|C_1|\sin\theta_{12}},\frac{a_1}{|C_1|\sin\theta_{13}},\frac{a_2}{|C_2|\sin\theta_{21}},\frac{a_2}{|C_2|\sin\theta_{23}},\frac{a_3}{|C_3|\sin\theta_{31}},\frac{a_3}{|C_3|\sin\theta_{32}}\}=\min\{4.27,3.50,2.19,2.90,1.90,3.07\}=1.90$$

Now, z_1^*, z_2^*, and z_3^* must be found. Solving three LP problems with consideration of z_1, z_2, and z_3 separately yields $(x_1,x_2,x_3,x_4)=(17.22,35.05,0,0)$ with $z_1^*=2975.87$, $(x_1,x_2,x_3,x_4)=(16.66,4.52,10.20,0)$ with $z_2^*=386.64$, and $(x_1,x_2,x_3,x_4)=(36.82,0,0,3.98)$ with $z_3^*=310.45$, respectively. Then, we obtain the utopian point in which three objectives are satisfied at least with their optimal values while we reach to a common point. Therefore, we have,

$$\begin{aligned}
&MinD = 12d_1 + 5d_2 + 45d_3 + 2d_4 + 6d_5 +\\
&1000(d_6 + d_7 + d_8 + d_9)\\
&s.t.\\
&-6x_1 + 7x_2 + 5x_3 + 3x_4 \le 142 + d_1\\
&2x_1 + 7x_2 + 25x_3 + 9x_4 \le 320 + d_2\\
&20x_1 + 13x_2 + 40x_3 + 16x_4 \le 800 + d_3\\
&3x_1 - 10x_2 + x_3 - 24x_4 \le 15 + d_4\\
&16x_1 + 5x_2 - 2x_3 + 80x_4 \ge 228 - d_5\\
&10x_1 + 80x_2 + 25x_3 + 16x_4 \ge 2975.87\\
&6x_1 + 7x_2 + 25x_3 + 8x_4 \ge 386.64\\
&8x_1 - 5x_2 + 12x_3 + 4x_4 \ge 310.45\\
&x_1 \ge -d_6\\
&x_2 \ge -d_7\\
&x_3 \ge -d_8\\
&x_4 \ge -d_9\\
&x_1,..., x_4 : \textit{free in sign}\\
&d_i \ge 0; i = 1,...,9
\end{aligned} \tag{34}$$

The optimal solution is $(x_1^{**},x_2^{**},x_3^{**},x_4^{**})=(57.56,30,0,0)$ with $(z_1^{**},z_2^{**},z_3^{**})=(2975.60,555.36,310.48)$ and $D^{**}=33380.43$. In the next step, the DM is asked to select the objective which has the least desirability for him. Since the constraint associated with z_2 is not active, the DM is allowed to select one of the objectives z_1 or z_3 to keep its value. Suppose in the first iteration the DM adopts z_3. Therefore, the following problem should be solved,

$$\begin{aligned}
&MinD = 12d_1 + 5d_2 + 45d_3 + 2d_4 + 6d_5 +\\
&1000(d_6 + d_7 + d_8 + d_9)\\
&s.t.\\
&-6x_1 + 7x_2 + 5x_3 + 3x_4 \le 142 + d_1\\
&2x_1 + 7x_2 + 25x_3 + 9x_4 \le 320 + d_2\\
&20x_1 + 13x_2 + 40x_3 + 16x_4 \le 800 + d_3\\
&3x_1 - 10x_2 + x_3 - 24x_4 \le 15 + d_4\\
&16x_1 + 5x_2 - 2x_3 + 80x_4 \ge 228 - d_5\\
&x_1 \ge -d_6\\
&x_2 \ge -d_7\\
&x_3 \ge -d_8\\
&x_4 \ge -d_9\\
&8x_1 - 5x_2 + 12x_3 + 4x_4 \ge 310.48\\
&\sqrt{(x_1-57.56)^2+(x_2-30)^2+(x_3-0)^2+(x_4-0)^2}\\
&=1.9\\
&x_1,..., x_4 : \textit{free in sign}\\
&d_i \ge 0; i = 1,...,9
\end{aligned} \tag{35}$$

Table 2. The detailed information for implementation of the proposed method for example 2

Iteration	Objective	x_1	x_2	x_3	x_4	D	z_1	z_2	z_3
0	max z_1	17.22	35.05	0	0	0	2976.2	348.67	-37.49
0	max z_2	16.66	4.52	10.2	0	0	783.2	386.6	233.08
0	max z_3	36.82	0	0	3.98	0	431.88	252.76	310.48
0	utopian	57.56	30	0	0	33380.43	2975.6	555.36	310.48
1	keep z_3	56.74	28.29	-0.17	0	31484.82	2826.35	534.22	310.48
2	keep z_3	55.92	26.58	-0.33	0	29613.44	2677.35	513.33	310.48
3	keep z_1	54.4	27.09	-1.35	0	27726.82	2677.35	482.28	283.55
4	keep z_3	53.58	25.38	-1.52	0	25860.25	2528.2	461.14	283.55
5	keep z_3	52.76	23.67	-1.68	0	23988.88	2379.2	440.25	283.55
6	keep z_3	51.94	21.96	-1.85	0	22118.36	2229.95	419.11	283.55
7	keep z_3	51.12	20.25	-2.02	0	20244.01	2080.7	397.97	283.55
8	keep z_1	49.6	20.76	-3.04	0	18351.77	2080.7	366.92	256.52
9	keep z_1	48.08	21.27	-4.06	0	16465.06	2080.7	335.87	229.57
10	keep z_3	47.26	19.56	-4.23	0	14599.55	1931.65	314.73	229.57
11	keep z_3	46.44	17.85	-4.39	0	12728.18	1782.65	293.84	229.57
12	keep z_3	45.62	16.14	-4.56	0	10857.66	1633.4	272.7	229.57
13	keep z_1	44.1	16.65	-5.58	0	8965.42	1633.4	241.65	202.59
14	keep z_1	42.58	17.16	-6.6	0	7078.71	1633.4	210.6	175.64
15	keep z_3	41.12	16.21	-5.83	0	5830.92	1562.25	214.44	177.95
16	keep z_3	39.75	15.32	-4.86	0	4855.56	1501.6	224.24	183.08
17	keep z_3	38.38	14.43	-3.88	0	3882.43	1441.2	234.29	188.33
18	keep z_2	37.01	13.54	-2.91	0	2906.67	1380.55	244.09	193.46
19	keep z_2	35.64	12.65	-1.93	0	1933.53	1320.15	254.14	198.71
20	keep z_1	33.95	12.59	-1.07	0	1066.54	1320.15	265.08	195.81
21	keep z_1	32.26	12.53	-0.2	0	203.27	1320.15	276.27	193.03
22	keep z_1	30.45	12.59	0	0.52	0	1320.15	274.99	182.73
23	min delta	31.86	12.52	0	0	0	1320.2	278.8	192.28
24	max z_1	31.86	12.52	0	0	0	1320.2	278.8	192.28
25	max z_2	31.85	12.52	0	0	0	1320.2	278.8	192.28
26	max z_3	31.86	12.52	0	0	0	1320.2	278.8	192.28

The optimal solution for (35) is $(x_1, x_2, x_3, x_4) = (56.74, 28.29, -0.17, 0)$ with $(z_1, z_2, z_3) = (2826.35, 534.22, 310.43)$ and $D = 3148482$.

Table 2 summarizes the results of implementation of the proposed algorithm for example 2. Note that the constraint associated with z_2 is not active till iteration 8. Therefore, he is allowed to choose z_2 as the objective whose desirability is the least amount from iteration 8.

According to Table 2, we reach to the feasible region in iteration 22. So, solving the following problem helps us to attain the boundary of the feasible region,

$$
\begin{aligned}
&Min D = \sqrt{(x_1 - 32.26)^2 + (x_2 - 12.53)^2 + (x_3 + 0.2)^2 + (x_4 - 0)^2} \\
&s.t. \\
&10x_1 + 80x_2 + 25x_3 + 16x_4 \geq 1320.02 \\
&6x_1 + 7x_2 + 25x_3 + 8x_4 \geq 274.99 \\
&8x_1 - 5x_2 + 12x_3 + 4x_4 \geq 182.73 \\
&-6x_1 + 7x_2 + 5x_3 + 3x_4 \leq 142 \\
&2x_1 + 7x_2 + 25x_3 + 9x_4 \leq 320 \\
&20x_1 + 13x_2 + 40x_3 + 16x_4 \leq 800 \\
&3x_1 - 10x_2 + x_3 - 24x_4 \leq 15 \\
&16x_1 + 5x_2 - 2x_3 + 80x_4 \geq 228 \\
&x_j \geq 0;\ j = 1,...,4
\end{aligned}
\tag{36}
$$

The optimal solution for (36) is $(x_1, x_2, x_3, x_4) = (31.86,12.52,0,0)$ with $(z_1, z_2, z_3) = (1320.2,278.8,192.28)$ and $\delta = 0.44$.

Suppose the DM adopts z_1 as the objective to get improved. Hence,

$$\begin{aligned}
&Max z_1 = 10x_1 + 80x_2 + 25x_3 + 16x_4 \\
&s.t. \\
&\sqrt{(x_1 - 32.26)^2 + (x_2 - 12.53)^2 + (x_3 + 0.2)^2 + (x_4 - 0)^2} \le 1.9 \\
&6x_1 + 7x_2 + 25x_3 + 8x_4 \ge 274.99 \\
&8x_1 - 5x_2 + 12x_3 + 4x_4 \ge 182.73 \\
&-6x_1 + 7x_2 + 5x_3 + 3x_4 \le 142 \\
&2x_1 + 7x_2 + 25x_3 + 9x_4 \le 320 \\
&20x_1 + 13x_2 + 40x_3 + 16x_4 \le 800 \\
&3x_1 - 10x_2 + x_3 - 24x_4 \le 15 \\
&16x_1 + 5x_2 - 2x_3 + 80x_4 \ge 228 \\
&x_j \ge 0; j = 1,...,4
\end{aligned} \tag{39}$$

The optimal solution for (39) is $(x_1, x_2, x_3, x_4) = (31.86,12.52,0,0)$ with $(z_1, z_2, z_3) = (1320.2,278.8,192.28)$. Since z_1 does not change, we have $S = \{2,3\}$. Then, z_2 is adopted by the DM to get improved, which leads to,

$$\begin{aligned}
&Max z_2 = 6x_1 + 7x_2 + 25x_3 + 8x_4 \\
&s.t. \\
&\sqrt{(x_1 - 32.26)^2 + (x_2 - 12.53)^2 + (x_3 + 0.2)^2 + (x_4 - 0)^2} \le 1.9 \\
&10x_1 + 80x_2 + 25x_3 + 16x_4 \ge 1320.2 \\
&8x_1 - 5x_2 + 12x_3 + 4x_4 \ge 182.73 \\
&-6x_1 + 7x_2 + 5x_3 + 3x_4 \le 142 \\
&2x_1 + 7x_2 + 25x_3 + 9x_4 \le 320 \\
&20x_1 + 13x_2 + 40x_3 + 16x_4 \le 800 \\
&3x_1 - 10x_2 + x_3 - 24x_4 \le 15 \\
&16x_1 + 5x_2 - 2x_3 + 80x_4 \ge 228 \\
&x_j \ge 0; j = 1,...,4
\end{aligned} \tag{40}$$

The optimal solution for (40) is $(x_1, x_2, x_3, x_4) = (31.86,12.52,0,0)$ with $(z_1, z_2, z_3) = (1320.2,278.8,192.28)$. Obviously, z_2 remains unchanged; so, $S = \{3\}$. The only remaining objective is z_3 and we have,

$$\begin{aligned}
&Max z_3 = 8x_1 - 5x_2 + 12x_3 + 4x_4 \\
&s.t. \\
&\sqrt{(x_1 - 32.26)^2 + (x_2 - 12.53)^2 + (x_3 + 0.2)^2 + (x_4 - 0)^2} \le 1.9 \\
&10x_1 + 80x_2 + 25x_3 + 16x_4 \ge 1320.2 \\
&6x_1 + 7x_2 + 25x_3 + 8x_4 \ge 278.8 \\
&-6x_1 + 7x_2 + 5x_3 + 3x_4 \le 142 \\
&2x_1 + 7x_2 + 25x_3 + 9x_4 \le 320 \\
&20x_1 + 13x_2 + 40x_3 + 16x_4 \le 800 \\
&3x_1 - 10x_2 + x_3 - 24x_4 \le 15 \\
&16x_1 + 5x_2 - 2x_3 + 80x_4 \ge 228 \\
&x_j \ge 0; j = 1,...,4
\end{aligned} \tag{41}$$

The optimal solution for (41) is $(x_1, x_2, x_3, x_4) = (31.86,12.52,0,0)$ with $(z_1, z_2, z_3) = (1320.2,278.8,192.28)$. Since similar to z_1 and z_2, the amount of z_3 remains unchanged, we have $S = \phi$. Therefore, the final efficient feasible solution is $(x_1, x_2, x_3, x_4) = (31.86,12.52,0,0)$ with $(z_1, z_2, z_3) = (1320.2,278.8,192.28)$.

5. Conclusions

We have proposed a new interactive algorithm to solve MOLP in which we need some initial information about DM's preferences. Unlike the majority of interactive methods, the proposed method starts from the utopian point, which is usually infeasible, and moves towards the feasible region and the efficient set. Based on the results of some proved lemmas, we are able to specify the amount of steps towards the feasible region. Our method satisfies most of the characteristics that a good interactive method needs, such as simplicity of the nature of judgments for DM, having the opportunity to compensate improper decisions in previous interactions, and handling his nonlinear utility. Implementation of the proposed method has been demonstrated by using two numerical examples.

REFERENCES

[1] F. B. Abdelaziz, "Multiple objective programming and goal programming: New trends and applications," European Journal of Operational Research, Vol. 177, pp. 1520–1522, 2007.

[2] M. M. Wiecek, "Multiple criteria decision making for engineering," Omega, Vol. 36, pp. 337–339, 2008.

[3] J. Kim and S. K. Kim, "A CHIM-based interactive Tchebycheff procedure for multiple objective decision making," Computers & Operations Research, Vol. 33, pp. 1557–1574, 2006.

[4] M. Sun, "Some issues in measuring and reporting solution quality of interactive multiple objective programming procedures," European Journal of Operational Research, Vol. 162, pp. 468–483, 2005.

[5] M. Zeleny, "Multiple criteria decision making," MC Graw-Hill, New York, 1982.

[6] R. Kenney and H. Raiffa, "Decisions with multiple objectives: Preferences and value trade-offs," J. Wiley, New York, 1976.

[7] C. Romero, "Handbook of critical issues in goal programming," Pergamon Press, Oxford, 1991.

[8] M. Ida, "Efficient solution generation for multiple objective linear programming based on extreme ray generation method," European Journal of Operational Research, Vol. 160, pp. 242–251, 2005.

[9] L. Pourkarimi, M. A. Yaghoobi and M. Mashinchi, "Determining maximal efficient faces in multiobjective linear programming problem," Journal of Mathematical Analy-

sis and Applications, Vol. 354, pp. 234–248, 2009.

[10] R. E. Steuer and C. A. Piercy, "A regression study of the number of efficient extreme points in multiple objective linear programming," European Journal of Operational Research, Vol. 162, pp. 484–496, 2005.

[11] E. A. Youness and T. Emam, "Characterization of efficient solutions for multi-objective optimization problems involving semi-strong and generalized semi-strong e-convexity," Acta Mathematica Scientia, Vol. 28B(1), pp. 7–16, 2008.

[12] S. I. Gass and P. G. Roy, "The compromise hypersphere for multiobjective linear programming," European Journal of Operational Research, Vol. 144, pp. 459–479, 2003.

[13] J. Chen and S. Lin, "An interactive neural network-based approach for solving multiple criteria decision making problems," Decision Support Systems, Vol. 36, pp. 137–146, 2003.

[14] A. Engau, "Tradeoff-based decomposition and decision-making in multiobjective programming," European Journal of Operational Research, Vol. 199, pp. 883–891, 2009.

[15] L. R. Gardiner and R. E. Steuer, "Unified interactive multiple objective programming," European Journal of Operational Research, Vol. 74, pp. 391–406, 1994.

[16] C. Homburg, "Hierarchical multi-objective decision making," European Journal of Operational Research, Vol. 105, pp. 155–161, 1998.

[17] C. L. Hwang and A. S. M. Masud, "Multiple objective decision making methods and applications," Springer-Verlag, Amsterdam, 1979.

[18] B. Malakooti and J. E. Alwani, "Extremist vs. centrist decision behavior: Quasi-convex utility functions for interactive multi-objective linear programming problems," Computers & Operations Research, Vol. 29, pp. 2003–2021, 2002.

[19] G. R. Reeves and K. R. MacLeod, "Some experiments in Tchebycheff-based approaches for interactive multiple objective decision making," Computers & Operations Research, Vol. 26, pp. 1311–1321, 1999.

[20] R. E. Steuer, J. Silverman, and A. W. Whisman, "A combined Tchebycheff/aspiration criterion vector interactive multi-objective programming procedure," Computers & Operations Research, Vol. 43, pp. 641–648, 1995.

[21] M. Sun, A. Stam, and R. E. Steuer, "Interactive multiple objective programming using Tchebycheff programs and artificial neural networks," Computers & Operations Research, Vol. 27, pp. 601–620, 2000.

[22] G. R. Reeves and J. J. Gonzalez, "A comparison of two interactive MCDM procedures," European Journal of Operational Research, Vol. 41, pp. 203–209, 1989.

[23] A. R. P. Borges and C. H. Antunes, "A visual interactive tolerance approach to sensitivity analysis in MOLP," European Journal of Operational Research, Vol. 142, pp. 357–381, 2002.

[24] J. T. Buchanan and H. G. Daellenbach, "A comparative evaluation of interactive solution methods for multiple objective decision models," European Journal of Operational Research, Vol. 29, pp. 353–359, 1987.

[25] A. A. Geoffrion, J. S. Dyer, and A. Feinberg, "An interactive approach for multi-criterion optimization with an application to the operation of an academic department," Management Science, Vol. 19, pp. 357–368, 1972.

[26] R. E. Steuer and E. U. Choo, "An interactive weighted Tchebycheff procedure for multiple objective programming," Mathematical Programming, Vol. 26, pp. 326–344, 1983.

[27] S. Zionts and J. Wallenius, "An interactive multiple objective linear programming method for a class of underlying nonlinear utility functions," Management Science, Vol. 29, pp. 519–529, 1983.

[28] D. Vanderpooten, "The interactive approach in MCDA: A technical framework and some basic conceptions," Mathematical and Computer Modelling, Vol. 12, pp. 1213–1220, 1989.

[29] G. R. Reeves and L. Franz, "A simplified interactive multiple objective linear programming procedure," Computers and Operations Research, Vol. 12, pp. 589–601, 1985.

J. Software Engineering & Applications, **2009, 2: 248-258**
doi:10.4236/jsea.2009.24032 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

Scientific Research

An Aspect-Oriented Approach for Use Case Based Modeling of Software Product Lines

Stéphane S. SOMÉ, Pauline ANTHONYSAMY

SITE, University of Ottawa, Ottawa, Canada.
Email: ssome@site.uottawa.ca

Received May 29th, 2009; revised June 22nd, 2009; accepted June 27th, 2009.

ABSTRACT

Software Product Line Development advocates software reuse by modeling common and variable artefacts separately across members of a family of products. Aspect-Oriented Software Development aims at separation of concerns with "aspects" to increase modularity, reusability, maintainability and ease of evolution. In this paper, we apply an aspect-oriented use case modeling approach to product line system modeling. A use case specification captures stakeholders concerns as interactions between a system and its actors. We adapt our previous work with the introduction of a "variability" relationship for the expression of variabilities. This relationship is used to model variable and common behaviours across a family of products as use cases. A variability composition mechanism enables building of executable behaviour models for each member of a product line family by integrating common elements with the applicable variable elements.

***Keywords*:** *Product Lines, Use Cases, Aspects, Requirements Modeling*

1. Introduction

The importance of a *Software Product Line* (*SPL*) emerged from the field of software reuse when developers realized that reusing development artefacts such as requirements, designs, and components across different members of a product family significantly reduces cost, effort and time. According to Clements *et al*. [1], a software product line is defined as "*a set of software intensive systems sharing common, managed set of features that satisfy specific needs of a particular market segment and that are developed from a common set of core assets in a prescribed way*". However, effectiveness of a software product line does not solely depend on reuse capability but also on how *commonalities* and *variabilities* of a product line are managed and modeled from the requirements phase to the implementation phase.

Use cases are widely used to model functional requirements in traditional as well as product line systems. A use case specification captures stakeholders concerns as interactions between a system and its actors. Various extensions to traditional use case modeling have been proposed for the expression of variability and commonality. For instance, Ecklund *et al*. [2] proposed *change case* to specify anticipated changes that may impact a software product line. Change cases provide an "*impact link*" that creates traceability to use cases whose implementations might be affected. In [3], Jacobson *et al*. proposed *variation points* and *abstract use cases* to model variabilities and commonalities with the UML "*extend*" and "*generalization*" relations. Whereas, Gomma [4], introduced UML stereotypes "*kernel*", "*alternatives*" and "*optional*" to distinguish common and variable use case specifications in software product lines. Similarly, John and Muthig [5] proposed stereotype "*variant*" and the marking of sections of use case diagrams as optional to model variabilities. As for use case descriptions, they advocate using XML tags *'variant' '/variant'* to mark optional and alternative steps (and scenarios).

In our previous work [6], we proposed an approach to support use cased based requirements engineering. This approach is supported by a tool called *Use Case Editor, (UCEd)* [7]. UCEd is a use case modeling tool that takes a set of related use cases written in a restricted natural language and automatically generate executable State Charts that integrates the partial behaviours defined by these use cases. A domain model is used for syntactical analysis of use cases and as a basis for state model generation from the use cases. We then extended our approach to support modelling aspects in use case specifications [8]. We introduced an *"aspect"* relation for crosscutting requirements and derived a composition mechanism for the generation of a global behaviour

model integrating use cases with crosscutting concerns. In this paper, we apply this *aspect-oriented use case modeling* approach to product line systems specification. A number of recent works have demonstrated that applying *Aspect-Oriented Software Development* (*AOSD*) to SPL provides an improved mechanism to encapsulate and model variabilities and commonalities throughout the entire software lifecycle [9−11]. We model variabilities and crosscutting commonalities as use cases and link them with a "*variability*" relation. The *"variability"* relation is a *specialization* of the *"aspect"* relation. The approach allows variabilities and commonalities to be better encapsulated and modularized. An aspect composition mechanism enables building of executable behaviour models for each member of a product line family by integrating common elements with the applicable variable elements.

The remainder of this paper is organized as follows. Section 2 presents some background material on use case modeling and presents our approach on modeling the concerns with "*aspect*" relation. In section 3, we present our argument on modeling variability and commonalities in a product line by adapting the approach presented in section 2 and describe variability compositions in terms of Petri net formalism. We then position our work relatively to close works in Section 4 and finally, section 5 concludes the paper and discusses some future works.

2. Use Case Modeling

A use case is defined as "*the specification of a set of actions performed by a system (or subsystem), which yields an observable result that is, typically, of value for one or more actors or other stakeholders of the system.*" [12]. In this section, we briefly review the UML use case relationships and our *"aspect"* relationship introduced in [8] for crosscutting concerns.

2.1 UML Use Case Modeling

A UML use case model includes use cases, actors and relationships. There are three types of relationships between use cases – *include*, *extend* and *generalization*. An include relationship $uc_{base} \times uc_{inc}$ represents the inclusion of use case uc_{inc} as a sub-process of use case uc_{base} (*base use case*). An extend relationship $uc_{ext} \times econd \times epoints \times uc_{base}$ represents an extension of a base use case, uc_{base} by an extension use case, uc_{ext}. Behaviors described in the extension use case are included at specific places in the base use case called *extension points*, *epoints*. Each extension is realized under a specific condition known as, *econd*. Whereas, a generalization relationship defines an inheritance relation between an abstract use case and a more specific use cases. Figure 1 shows a UML use case diagram for a Microwave System (single system).

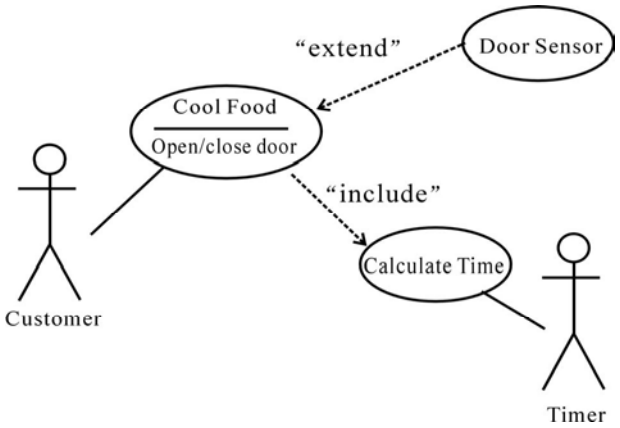


Figure 1. UML use case diagram for a microwave oven

Use case diagrams represent abstract overview of a system. Each use case is specified in the form of description of interactions (as natural language text) between a user and the system. In order to support automated synthesis of state models from use cases, we formalized use case description by defining an abstract syntax, a concrete syntax based on natural language and by providing Petri nets based semantics to use cases [13].

Figure 2 shows an example of a use case. A semi-formal natural language* is used for *operations* and *conditions* in use case steps. The UCEd tool uses a domain model where domain entities including operations are defined to validate the use cases [6]. The domain model serves as a high-level class model that captures domain concepts and their relationships. Use case execution semantics are expressed using the Petri nets formalism [14] and an algorithm described in [13] generates Petri nets from use cases as an intermediate model for UML State Charts.

2.2 Aspect-Oriented Use Case Modeling

Aspects-oriented software development aims at providing software developers techniques and tools to better manage crosscutting concerns. A crosscutting concern is typically scattered among several other concerns. Crosscutting concerns needs to be identified and effectively handled from the beginning of the development lifecycle (i.e. requirements engineering). Jacobson and Ng [15] noticed a close relation between use cases and aspects as each use case typically crosscuts a set of components and usually involves crosscutting concerns such as synchronization, accuracy, access control and more. Other approaches that were proposed for aspect-oriented modeling of use cases include [16,17]. In our *aspect-oriented use case modeling* approach, crosscutting concerns defined as advice use cases are linked to affected concerns using an *"aspect"* relationship [8]. Differently to the UML relationships, the *"aspect"* relationship cardinality

*Operations are specified as active sentences and conditions are in the form of predicative sentences. Further reference to the semi-formal natural language can be found in [6, 13].

is *one to many*. We defined *AspectJ* [18] constructs in use case terms and adopted a *symmetric* model where all concerns (including crosscutting concerns) may be extended as opposed to AspectJ *asymmetric* model.

Formally, an "*aspect*" relation defined as $uc_{adv} \times acond \times apcuts \times baseUCs$ links the advice use case, uc_{adv} to the set of base use cases *baseUCs* according to pointcut expressions *apcuts* when condition *acond* is fulfilled. As mentioned previously, the cardinality of "*aspect*" relation is *one to many* since crosscutting concerns typically influences several use cases. The set of an *"aspect"* relationship target use cases is identified using parameterization based on use case description elements such as name, title, primary actors, goals, post-conditions etc. Advice use cases that capture crosscutting requirements are defined in the same form as normal use cases. However, we do not require that advice use cases strictly adhere to use cases *well-formness* rules as stated in the UML specification [12]. An advice use cases may only have some of a use case sections. We also allow advice use cases to be initiated by the system and to describe incomplete interactions. The linking of advice use cases with affected base use cases is based on syntactical matching of *joinpoints* (potential occurrence of a crosscutting concern in a base use case) and *pointcut expressions*. Any use case description element i.e. *steps, operations, alternatives, extension points,* etc is a possible joinpoint. Pointcuts are parameterized pattern-based expressions that match joinpoints.

Additionally, a pointcut specifies *how* advice use cases are weaved at the joinpoints. Parameterization is essential in pointcut expressions since the number of target joinpoints can be large, complex and unpredictable. For instance a pointcut specified as "step 1, 2" refers to step one and two of the use case and "step *" refers to all the steps in the use case. Similarly, pointcut "operations Microwave System *" refers to all operations of entity "Microwave System" and operations "*opens*" matches all operations that contains the word "open" as part of the operation name. Three types of advice weaving are traditionally defined in AOP: *before*, *after* and *around*. We consider the same composition types with an additional type for concurrent composition (*concurrent*). Below is a brief description of each composition type:

- ***before***: crosscutting requirements are applied before a joinpoint
- ***after***: crosscutting requirements are applied after a joinpoint
- ***around***: crosscutting requirements are applied instead of a particular joinpoint (*wrapping*)
- ***concurrent***: crosscutting requirements are applied concurrently with a joinpoint (*in parallel*)

Figure 3 shows UCEd representation of the Microwave Oven use case model with <<*aspect*>> relations. The model includes one advice use case "Safety". The target use cases of the <<*aspect*>> relation from use case "Safety" are specified as a wildcard "*". "Safety" use case is therefore applied to the all other use cases in the model but itself (an advice use case is always excluded from its set of related use cases to avoid infinite inclusion). The "Safety" advice use case is weaved based on operation pointcut "***after** operations * **presses Start** **" (defined as a children of the relation).

Finally, we defined aspect composition at the Petri net level [8]. Behaviours defined in advice use cases are weaved with the affected base use cases according to pointcut expressions. A global behaviour model is obtained by integrating all crosscutting concerns and base use cases. Resulting Petri net models may be transformed

Title: Cook Food
Primary Actor: User
Goal: Allows User to cook, heat food
Precondition: Microwave oven is idle
Post-condition: Microwave oven has cooked the food
Steps:
1.The User opens the oven door
2.The User puts the food in the oven
3.The User closes the oven door
4.The User presses the Cooking Time button
5.The Microwave System prompts the User for the cooking time
6.The User enters a cooking time on the numeric keypad and presses Start button
7.The Microwave System starts cooking the food
8.The Microwave System continually cooks and displays the cooking time remaining
9.The Timer elapses and notifies the Microwave System
10.The Microwave System stops cooking the food and displays the end message
11.The User opens the oven door
12.The User removes the food from the oven
13.The User closes the door
14.The Microwave System clears the display
Alternatives:
6.a.The Cooking time is zero
6.a.1.The Microwave System notifies the User of an error

Figure 2. Details of use cases "Cook Food"

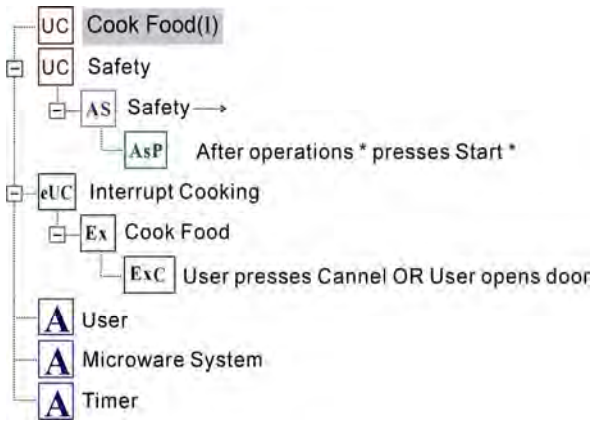


Figure 3. UCEd representation use cases with "*aspect*" relations

to UML State Charts and used as prototypes for requirements validation by simulation in UCEd [13].

3. Adapting Aspect-Oriented Use Case Modeling to Product Lines

A product line is a set of products that share a common set of characteristics and yet differ from each other based on a set of variabilities. *Product Line Engineering* (*PLE*) is about exploiting commonalities across product line systems while managing variabilities in order to improve reusability (of software artifacts such as requirements, models, components etc.), reduce time to market, cost and improve product quality. Commonalities are features that are common to a set of products and variabilities are features that products may optionally have in a product family. Variabilities influence software systems in the similar manner as crosscutting concerns [9–11,19]. For instance, variability "*x*" may be implemented several times and over different products across a product family. According to [19], aspects and variability are orthogonal concepts, which are independent of the core system and can be combined with it when needed.

In this section, we adapt our aspect-oriented use case modeling approach [8] to variabilities and commonalities. We introduce a *"variability"* relation to model variabilities and crosscutting commonalities in a product line system. The *"variability"* relation is a *specialization* of the *"aspect"* relation. It has similar characteristics as *"aspect"* relation but with some differences. Similar to crosscutting concerns, variabilities and crosscutting commonalities are specified as advice use cases and weaved into base use cases according to the different types of pointcut expressions. Our objectives in adapting aspect-oriented use case modeling to product lines specification include better encapsulation and modularity of variabilities and commonalities. Aspect-orientation allows variabilities and crosscutting commonalities to be modeled separately thus, improves readability as well as system evolution. By implementing the approach as part of the UCEd toolset, we also aim at providing traceability between products and features and take advantage of UCEd simulation capabilities for product design and validation.

We illustrate our approach with a product-line version of the microwave system [4].The product line consists of microwave ovens that come with features ranging from basic to advanced features. A basic microwave oven system has input buttons for selecting Cooking Time, Start, and Cancel, as well as a numeric keypad. It also has a display to show remaining cooking time. Additionally, the oven has a microwave-heating element for cooking food, a weight sensor to detect if there is an object in the oven. Optional features for more-advanced microwaves include a beeper to indicate when cooking are done, a light that is switched on when the door is open and when food is being cooked, and a turntable.

3.1 "Variability" Relation

The *"variability"* relation is a *"specialization"* of *"aspect"* relation for modeling variabilities and crosscutting commonalities in a product line system. The *"variability"* relation links variabilities and crosscutting commonalities to affected base use cases and specifies pointcuts. More formally, a "*variability*" relationship $uc_{var} \times plcond \times vpcuts \times baseUCs$ specifies that variabilities defined as an advice use case uc_{var} are weaved to the set of base use cases *baseUCs* according to pointcut expressions *vpcuts* when condition *plcond* is fulfilled. Condition, *plcond* specifies whether a given member of the product line provides the functionality described by the advice use case uc_{var}. Thus, the functionality is provided when *plcond* is *true*; whereas if the condition is *false*, then functionality is not provided. Similar to the *"aspect"* relation, the cardinality of the *"variability"* relation is *one to many*. This is a reflection of the fact that variabilities and crosscutting commonalities affect several use cases in a product line model. The *one-to-many* cardinality allows several base use case to be conveniently linked to a single variability. The set of target use cases is identified using parameterization in a similar way as the *"aspect"* relation. For instance, the targets of an *"variability"* relation specified as "*cook* are all use cases in the use case model which names contains the word "cook", i.e. "Cook Food" and "Interrupt Cooking" in the use case model in Figure 1. Parameterization is not limited to use case names but may also be used for description elements such as "*primary actors*", "*goals*", "*pre-conditions*" and "*post-conditions*". For instance, "Primary Actor User" matches all use cases with "User" as the primary actor. The parameterization allows changes to use case models and product evolution independently of variabilities, e.g. when a new variability option is added to a product family, changes to the existing model can be avoided. Furthermore, UCEd "tree" representation (refer to Figure 7) helps model variabilities and commonalities within a software product family in a clearer manner.

3.2 Composition Mechanism

Similar to crosscutting concerns, we define composition mechanism for product lines at the Petri net level, where advice use cases (variabilities) are weaved with affected base use cases based on pointcut expressions. Differently to composition mechanism for crosscutting concerns, where a global behavior model from use cases integrating all independently defined concerns is generated, we generate a Petri net for a particular product within a product line.

A Petri net is a triple [P, T, F] with: P a finite set of *places*, T a finite set of *transitions* and $F \subseteq (P \times T) \cup (T \times P)$

a *flow relation*. Places are represented in the graphical description of a Petri net, as circles, transitions as boxes and the flow relation corresponds to arrows. Figure 4 shows how advice use cases (variabilities) are weaved according to the different types of pointcuts. Use cases are mapped to Petri nets such that each reference to use case description elements (*operations, step, etc)* corresponds to a *transition* in the Petri Net model [13]. To ease our discussion, we model the composition mechanism based on pointcut expressions formed with operation joinpoints (*op*). In Figure 4, a_1, a_2, a_3 …are sequence of events in the advice use cases (variabilities) and transition corresponding to the operation joinpoint, *op* in the Petri net derived from the base use case is colored black. Notice that differently to the general situation in [8], "*variability*" conditions *(plcond)* are not included in the composition for variabilities. Only relevant advice sequences are weaved at the corresponding operations joinpoints. Similar composition mechanisms are used for other types of joinpoints (i.e. *steps, alternatives, extension points*); with transition *op* replaced by the corresponding elements.

Several advice use cases (variabilities) can refer to the same joinpoint. In that case, the advice use cases are sequentially ordered based on the type of pointcuts. *Before* advices are applied first and then *after* advices. The behaviour described in an *around* advices substitutes the behaviour described by a specific operation or step in the base use case based on the "*variability*" relations condition. Lastly, *concurrent* advices are executed in parallel with joinpoints and *around* advices. Several pointcuts of the same type are applied in a *non-deterministic* manner. For instance, in a situation where several before pointcuts refer to a same joinpoint, the corresponding advice use cases would be randomly weaved one after the other and before the joinpoint. Although, this is not an optimal resolution technique, UCEd allows the modeler to validate and simulate resulting Petri nets to uncover any inconsistencies that may occur and correct them accordingly.

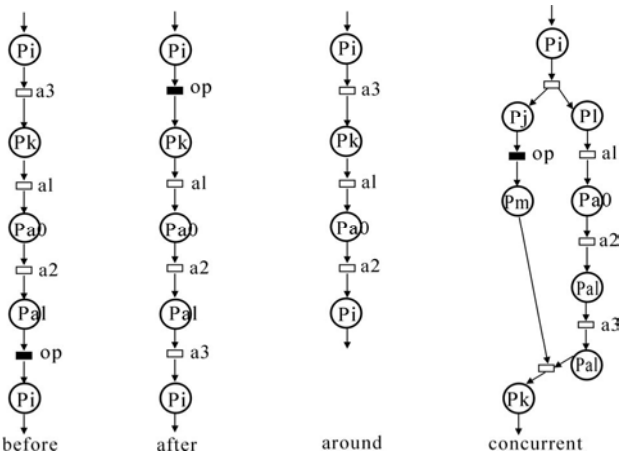


Figure 4. Weaving based on different types of operation, *op*

3.3 Modeling Variability in UCEd

Use case "Cook Food" in Figure 2 captures a common functionality of products in the microwave oven product line. We use feature modeling [20] to represent the high-level view of the Microwave Oven product family. *Optional* and *alternative* features describe variabilities in a product line and determine the characteristics of a given member of the product family. Optional features are variabilities that are required by some but not all members of a product family and alternative features are variabilities that are in different versions and are required by different members of the product line. These alternative variabilities are usually mutually exclusive. Figure 5 shows variabilities of the microwave oven product line as a feature model. Each microwave oven may include mandatory alternative and optional variabilities. The mandatory alternative variabilities are a *weight sensor* that is either Boolean or analog, a *display unit* that is

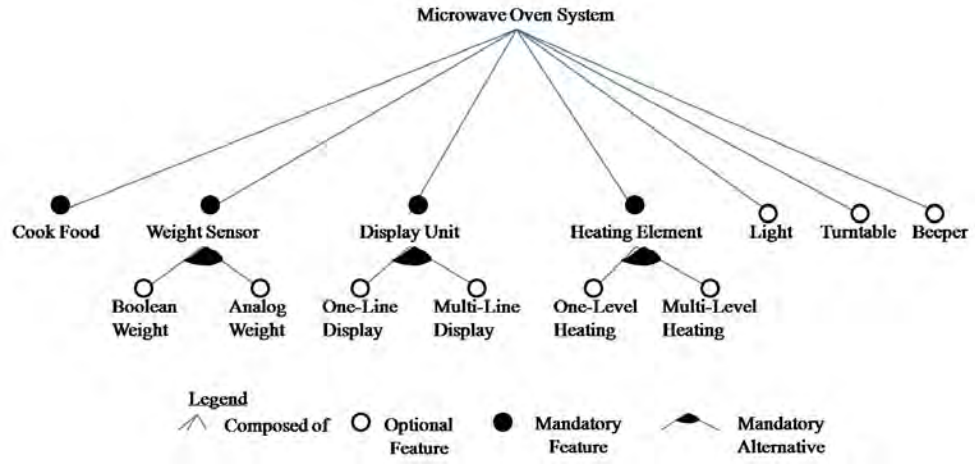


Figure 5. Feature model for microwave oven product line

one-line or multi-line, a one-level or multi-level *heating element*. The optional variabilities are a light, a turntable and a beeper.

We model variabilities as advice use cases and link them to commonalities with "*variability*" relations. As crosscutting concerns, variabilities are attached with conditions and pointcut expressions.

The attached condition specifies whether a given member of the product line provides the functionality described by the particular advice use case. Figure 7 shows UCEd representation of the commonalities and variabilities of the Microwave product line system. Use case "Cook Food" represents commonalities while, variabilities are attached with the "*variability*" relation and are linked to all other use cases using wildcards '*'. Each variable option is attached with a "*variability*" relation condition stating that the option is selected and a pointcut expression that specifies where the variable option is weaved. For instance, consider variability "One-Line Display" in Figure 7. The variability may affect all the other use cases in the model when condition "One Line Display option is selected" holds and it is weaved *before* step 1 of affected use cases. There can be multiple pointcut expressions attached to a single "*variability*" relation. For instance, in Figure 6 the "Light" option is weaved according to pointcuts "*concurrent operations User opens **" and "*concurrent operations Microwave System starts **".

UCEd uses a "tree" representation for use case models such that properties attached to a relation appears as children of that relation. This results in a representation where, variations in the product line are clearly distinguishable and identifiable. UCEd allows all variabilities in a product family to be modeled at the same time. Specific variabilities can be selected to form distinct members of the product family during composition.

Figure 6. UCEd use case integration tool

3.4 Product Generation

In order to generate a Petri net for a particular product within a product line, we need to specify which set of options apply. UCEd includes an integration tool to enable the selection of relevant features. Figure 6, illustrates the UCEd use case integration tool. Use cases listed in the integration tool are populated based on the *"variability"* relationship. Variable options that are selected are weaved with the common options and the corresponding Petri nets are generated. For instance, Table 1 shows the different feature sets for microwave oven models A and B.

Figure 6 shows the feature set selection for Model A in UCEd. The model consists of a Boolean weight sensor, one-line display panel, one-level heating element and a beeper as an optional feature. Figure 10 shows the advices use cases (variabilities) for model A and B.

Figure 8 shows the Petri net obtained from the composition of the features in Model A while Figure 9 shows the Petri net obtained from the composition of the features in Model B.

Notice that <<*variability*>> conditions are not included in the generated Petri nets. Disabled variabilities are simply ignored from the resulting model. This allows the derivation of Petri nets specific to each member of a product family.

Table 2 shows how the different composition types are used for variabilities. The weaving of variability behaviors occurs *after, before, around* or *concurrent* to use case description elements such as operations, steps, extension points etc. *Optional* and *alternative* variabilities are weaved with *after*, *before*, or *concurrent* composition types. For instance, in Figure 7, the "Boolean Weight" variability is weaved *after operations User puts* * and the "One-Line Display" option is weaved *before step 1* of use case "Cook Food". While, the *around* type can be used to substitute behaviors in commonalities with new behaviors. For instance, suppose that step 8 in use case "Cook Food" is expressed as follows: "*The Microwave System continually cooks and displays the cooking time remaining with a one-line display panel*". An *around* composition type can be used to replace the one-line display panel to with multi-line display panel. We consider the concurrent composition as an essential composition type for use case modeling and to model variabilities and commonalities within a software product family (refer section 3.2) since it allows several operations (or use cases) to execute in parallel.

Table 1. Feature set for microwave ovens Model A and B

Model A	Model B
Boolean Weight	Boolean Weight
One-Line Display	Multi-Line Display
One-Level Heating	One-Level Heating
Beeper	Turntable
	Beeper

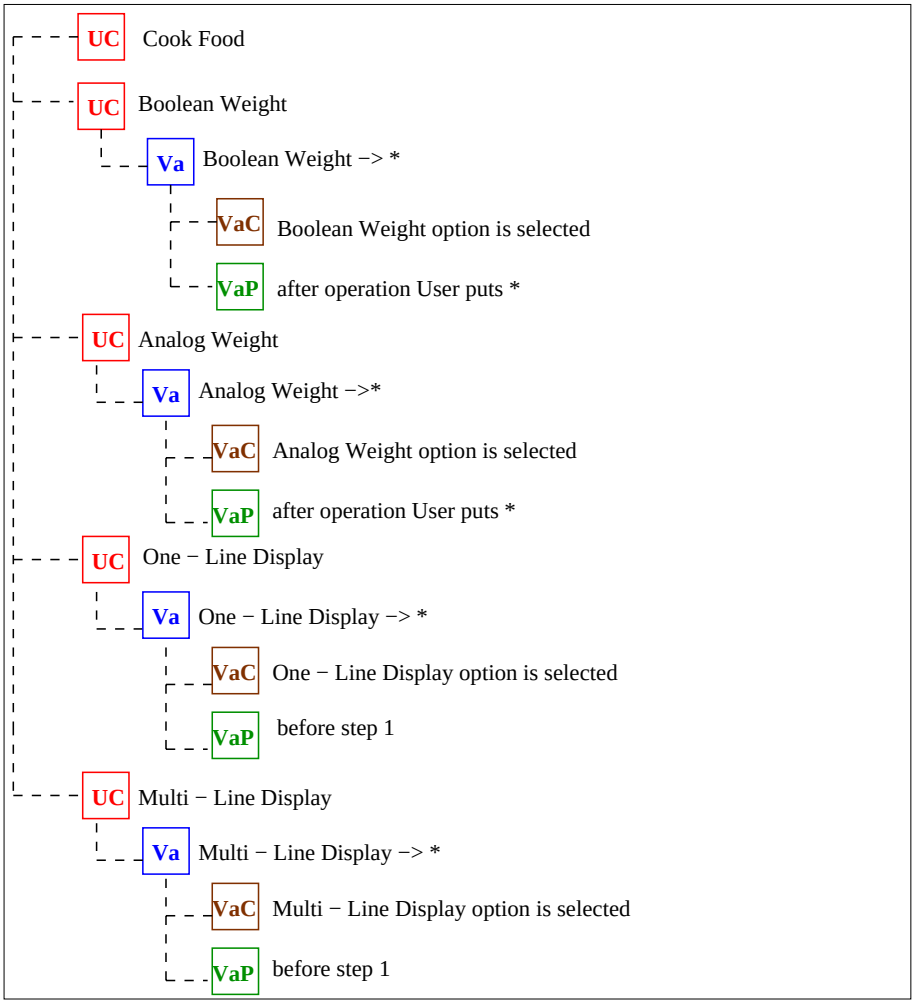


Figure 7. UCEd representation of variabilities with "*variability*" relations

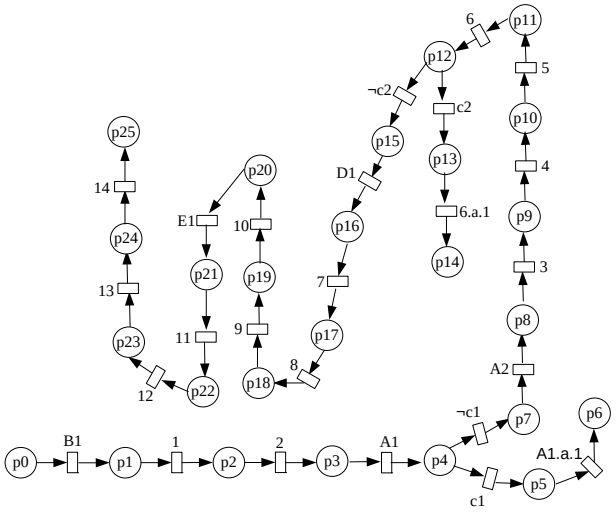


Figure 8. Petri net model for microwave oven Model A. We use corresponding step numbers as labels for transitions. c1 is condition "No item is present" and c2 is condition "Cooking time is zero"

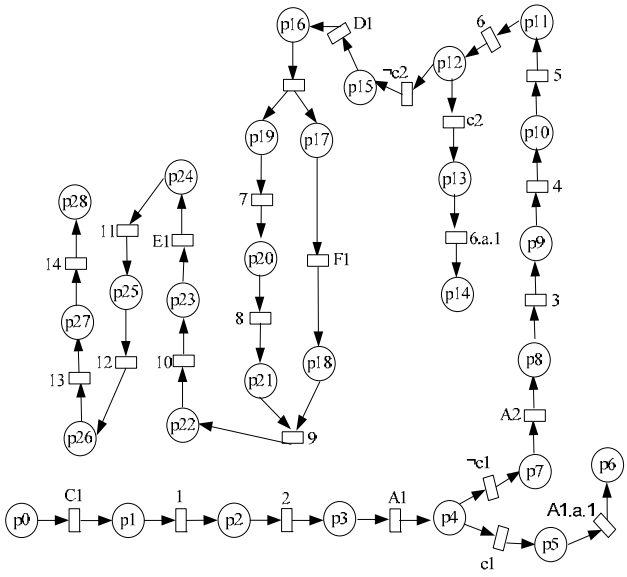


Figure 9. Petri net model for microwave oven Model B

Title: Boolean Weight Goal: To check that item is present in the oven Steps: A1.The Microwave System checks for items in the oven A2.IF an item is present THEN the Weight Sensor indicates to the Microwave System that an item is present Alternatives: A1.a.No item is present A1.a.1.The Weight Sensor indicates to the Microwave System that no item is present
Title: One-Line Display Goal: To display messages and cooking time Steps: B1.The Microwave System enables one-line display panel
Title: Multi-Line Display Goal: To display messages and cooking time Steps: C1.The Microwave System enables multi-line display panel
Title: One-Level Heating Goal: To heat food Steps: D1.The Microwave System enables one-level heating capability
Title: Beeper Goal: To indicate user when cooking stops Steps: E1.The Microwave System activates the beeper when cooking stops
Title: Turntable Goal: To rotate food while cooking in progress Steps: F1.The Microwave System rotates for the duration of cooking

Figure 10. Advice use cases "Boolean Weight", "One-Line Display", "Multi-Line Display", "One-Level Heating", "Beeper" and "Turntable"

Table 2. Composition types

	Compositions
After	Weave behavior after a use case description element
Before	Weave behavior before a use case description element
Concurrent	Weave variable behavior in parallel with a use case description element
Around	Substitute a use case description element with variable behavior

Some inconsistencies that may result from the combination of variabilities and commonalities can be identified using Petri nets analysis techniques integrated to UCEd [8]. UCEd also implements an algorithm for State Charts generation from Petri nets [13].

Figure 11 illustrates the UCEd generated State Chart for Model A with the weaved variabilities in bold.

UCEd allows simulation of the resulting State Charts as prototypes. It is thus possible to validate specific product member's characteristics within a product family. Furthermore, test cases can be derived for early validations of specific member of a product family.

4. Related Work

In [4], Gomma presents a UML-based software design

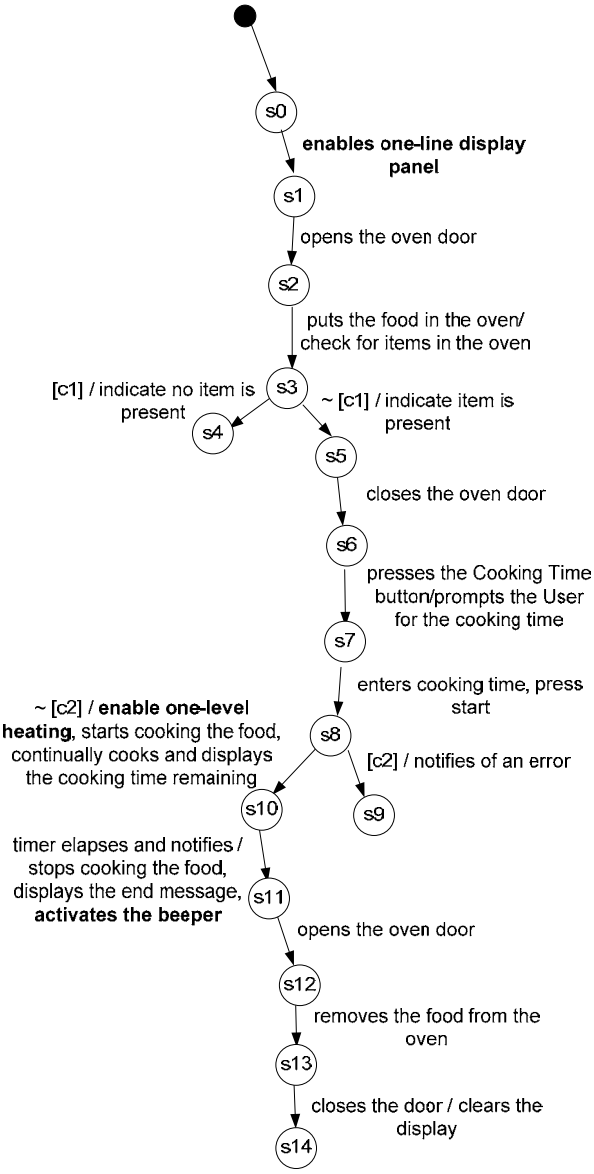


Figure 11. State chart for microwave oven Model A

method for software product lines called PLUS (Product Line UML-Based Software Engineering). The PLUS method extends the UML-based modeling approaches that are usually used for single systems to address software product lines. The objective of PLUS is to explicitly model commonalities and variabilities in a software product line. The method includes three stages: software product line requirements modeling, software product line analysis modeling and software product line design modeling. Firstly, during requirements modeling, kernel, optional and alternative use cases are developed to define the functional requirements of a system. The approach introduces UML stereotypes *"kernel"*, *"alternatives"* and *"optional"* to distinguish common and variable use case specifications in software product lines. Kernel use

cases describe functionalities that are common to all members of a software product line, whereas optional and alternative use cases describe variable functionalities which are specific to only certain members of a product line family. The approach also uses variation points to describe the location in a use case where a change can take place. Variation points are defined in a variation point section within a use case. Each identifies a line number where variability can be introduced or a conditional *"extend"* or *"include"* relationship. A feature model is then developed to capture the product line commonalities and variabilities and to describe how they relate to the use case models. During the analysis modeling, while static models are developed for defining kernel, optional and variant classes and their relationships, dynamic models which include state charts and interaction models are also developed. The state charts define the state dependent aspect of a product line and the interaction models describe the dynamic interaction between the objects that participate in each kernel, optional and alternative use case. Finally, during the design modeling stage, the component-based software architecture for the product line is developed. A similarity between our approach and the PLUS method is that use cases serve to capture requirements for product line systems. However, there are several differences between our work and the PLUS method. Our scope is limited to requirement specification while PLUS aims to address the whole development cycle. Another distinction between our work and PLUS concerns the way variation points are specified within variable use cases. Variation points are explicitly expressed in [4] and traditional use case relationships ("*extend"* and "include") are used to link variabilities with commonalities. This creates a strong dependency between variable and kernel (base) use cases which limit flexibility and modularity. A similar approach, where variation points are explicitly specified in base use cases and *"extend"* or "*generalization*" relations are used to link variabilities, is used in [3]. By modeling variabilities as *aspects* and by using parameterized pattern based pointcuts, we avoid the dependency problem and allow the independent modeling of variabilities and commonalities. Unlike [3,4], we also propose a composition mechanism that allows the derivation of an executable behaviour model for each member of a product line family automatically integrating common elements with the applicable variable elements.

John and Muthig [5] describe how commonality and variabilities can be integrated and described in use case diagrams and textual use case descriptions. The approach allows modeling of variabilities by introducing a new type of use case stereotyped *"variant"*. Entire sections of use case diagrams may also be marked as variable and XML tags *'variant' '/variant'* may be used to mark optional and alternative steps (and scenarios) in use case descriptions. The approach in [5] is very similar to Gomma's approach [4]; in that both extend the traditional use case modeling approach to accommodate modeling of commonality and variability is software product lines. The approach illustrates how a particular single-system use case can be extended to capture product line information, especially variabilities. However, while defining variabilities with a use case stereotyped as *"variant"* is an effective way to separate variable behaviours from the common behaviours in a product line, it forces requirements engineers to again adapt to a new type of use case. Furthermore, using XML tags *'variant' '/variant'* to mark optional and alternative steps (and scenarios) within use case specifications causes significant cluttering and thereby reduces readability. In our approach, variabilities and commonalities are modeled as normal use cases. We introduce a *"variability"* relation that links the selected variabilities to the base use cases based on pointcut expressions. The application of aspect- orientation allows variabilities and commonalities to be separately modeled and thus improves readability and maintainability. An additional advantage of our approach over [5] is that it allows the automated derivation of an executable specification for each product starting from textual use cases.

Our approach can also be seen as related to approaches that focus on the application of aspects to product lines modeling in general. There has been a significant interest in the AOSD community emphasizing the relations between variability and commonality in a software product line with *aspects* at the requirements engineering stage. *Aspect-Oriented Software Product Lines (AOSPL)* [21] is part of AOSD which focuses on *early aspects* in product lines. Various approaches suggest that variabilities in software product lines be modeled in the same manner as *crosscutting concerns*. For example Saleh and Gomaa [10] suggest grouping *optional* and *alternative* source code based on features in a variable source code file. This variable source file corresponds to an aspect file. Desired features can be selected and checked for consistency with a prototype tool which then are automatically integrated and compiled with the kernel source code to generate an executable member of a product line. Loughran *et al*. [9] use aspect-oriented techniques with natural language processing to facilitate requirements analysis and concern identification for the derivation of suitable feature-oriented models for implementation. This approach is implemented in a tool called NAPLES. The tool takes textual requirements, deduces concerns, aspects, feature commonalities and variabilities to ease implementation. Siy *et al*. [11] present an approach to separate functional requirements as *viewpoints* and non-functional requirements into aspects similar to ARCADE [22]. Their approach includes parameterization of requirements and a composition mechanism.

Our approach is similar to [9–11] by the application of AOSD to product lines. A major distinction is that we model crosscutting requirements and variabilities using use cases in textual form. Use cases enable the application of aspect-orientation early in the development lifecycle and thus, prevent crosscutting requirements from being overlooked.

5. Discussion & Conclusion

In this paper, we presented an approach for use case based modeling of software product line systems. We introduced a *"variability"* relation; a specialization of an *"aspect"* relation proposed for modeling crosscutting concerns [8]. Use case description elements are used as joinpoints and variable requirements are weaved with common requirements based on specified pointcut expressions. Selected variabilities and commonalities are composed and transformed into Petri net models as a step toward UML State Charts generation. The whole approach is automated and tool supported by adding extensions to UCEd, an existing use case-modeling tool. UCEd includes facilities for simulation of generated state charts and test generation. The simulation of a product model enables early validation of specific product members in the product family.

We noted some limitations to our proposed approach. For instance, some of the steps and operations required to be re-written in order to match them with appropriate pointcut expressions and ensure correct composition. For example, in our Microwave Oven example the "Analog Weight" advice use case is weaved *after operations User puts* *. The modeler should verify the base use case to make sure that the "Analog Weight" advice is only weaved at the relevant places and there are no other operations that would match the specified pointcut. We also found that not all variabilities could be modeled easily. Examples of such variabilities are "One-Line Display Unit" and "Display Languages" [4]. These variabilities are more of initialization type. Therefore, multiple weaving at different locations is not appropriate. Thus, we cater a workaround in that; these variabilities are initialized before the execution of affected base use cases. For instance, "One-Line Display Unit" is weaved *before step 1* of use case "Cook Food". Another potential problem is the scalability and complexity of resulting Petri net models when a great number of variability options are composed together for a specific product in a product family. This may make it harder to read and understand the resulting models and leave only simulation as validation approach.

In our future work, we plan to apply our approach to more case studies and investigate further ways of modeling variabilities and commonalities for software product lines in UCEd. We also plan to explore other composition types (besides before, after, around and concurrent) and define a more detailed joinpoint structure.

REFERENCES

[1] P. Clements and L. Northrop, "Software product lines: Practices and patterns," Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[2] E. Ecklund, L. Delcambre, and M. Freiling, "Change cases-Use cases that identify future requirements," Proceedings of OOPSLA 96, San Jose, Ca, pp. 342−358, October 1996.

[3] I. Jacobson, M. Griss, and P. Jonsson, "Software reuse–Architecture," Process and Organization for Business success, Addison-Wesley, 1997.

[4] H. Gomaa, "Designing software product lines with UML–From use cases to pattern-based software architectures," Addison-Wesley, 2004.

[5] I. John, and D. Muthig, "Product line modeling with generic use cases," SPLC-2 Workshop on Techniques for Exploiting Commonality Through Variability Management, Second Software Product Line Conference, San Diego, USA, August 2002.

[6] S. Somé, "Supporting use cases based requirements engineering," Information and Software Technology, Vol. 48(1), pp. 43–58, 2006.

[7] Use Case Editor (UCEd) toolset, http://www.site.uott-awa.ca/~ssome/UseCaseEditorUCEd.html.

[8] S. S. Somé and P. Anthonysamy, "An approach for aspect-oriented use case modeling," Workshop on Early Aspects at ICSE 2008: Aspect-Oriented Requirements Engineering and Architecture Design, May 2008.

[9] N. Loughran, A. Sampaio, and A. Rashid, "From requirements documents to feature models for aspect oriented product line implementation," Workshop on MDD in Product Lines at MODELS, 2005.

[10] M. Saleh and H. Gomaa, "Separation of concerns in software product line engineering," In Proceedings of the 2005 Workshop on Modeling and Analysis of concerns in Software. St. Louis, Missouri, 2005.

[11] H. Siy, P. Aryal, V. Winter, and M. Zand, "Aspectual support for specifying requirements in software product Lines," Workshop on Early Aspects at ICSE. Minneapolis, USA, May 2007.

[12] OMG: UML 2.1.2 Superstructure Object Management Group, 2007.

[13] S. S. Somé, "Petri nets based formalization of textual use cases," Technical Report TR-2007-11, SITE, University of Ottawa, 2007.

[14] C. A. Petri, "Communication with Automata," PhD thesis, Technische Universitat Darmstadt, 1962.

[15] I. Jacobson and P. W. Ng, "Aspect-oriented software development with use cases," Addison Wesley, 2005.

[16] J. Araújo and A. M. D. Moreira, "An aspectual use case driven approach," In VIII Jornadas Ingeniería del Software y Bases deDatos (JISBD 2003), pp. 463−468, 2003.

[17] R. Chitchyan, A. Rashid, P. Rayson, and R. Waters, "Semantics-based composition for aspect-oriented requirements engineering," In: AOSD '07: Proceedings of the 6th International conference on Aspect-oriented software development, ACM, pp. 36–48, 2007.

[18] AspectJ Project, http://www.eclipse.org/aspectj/.

[19] R. Stoiber, S. Meier, and M. Glinz, "Visualizing product line domain variability by aspect-oriented modeling," Proceedings of the 2nd International Workshop on Requirements Engineering Visualization (REV'07), in conjunction with RE'07. New Delhi, India.

[20] M. Griss, J. Favaro, and M. d'Alessandro, "Integrating feature modeling with the RSEB," Proceedings of Fifth International Conference on Software Reuse, Victoria, B.C., 1998.

[21] Aspects and Software Product Lines (EA-SPLC), http://trese.cs.utwente.nl/workshops/early-aspects-SPLC2005/.

[22] A. Rashid, A. Moreira, and J. Araújo, "Modularization and composition of aspectual requirements," In Proc. of the 2nd International Conference on Aspect-Oriented Software Development (AOSD'03), Boston, MA, pp. 11–20, March 2003.

J. Software Engineering & Applications, 2009, 2: 259-266
doi:10.4236/jsea.2009.24033 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

Nonparametric Demand Forecasting with Right Censored Observations

Bin ZHANG[1,2] ; Zhongsheng HUA[2]

[1]Institute for Economics and Lingnan College, Sun Yat-sen University, Guangzhou, China; [2]School of Management, University of Science and Technology of China, Hefei, China.
Email:[1] bzhang3@mail.ustc.edu.cn

Received July 17th, 2009; revised August 10th, 2009; accepted August 18th, 2009.

ABSTRACT

In a newsvendor inventory system, demand observations often get right censored when there are lost sales and no backordering. Demands for newsvendor-type products are often forecasted from censored observations. The Kaplan-Meier product limit estimator is the well-known nonparametric method to deal with censored data, but it is undefined beyond the largest observation if it is censored. To address this shortfall, some completion methods are suggested in the literature. In this paper, we propose two hypotheses to investigate estimation bias of the product limit estimator, and provide three modified completion methods based on the proposed hypotheses. The proposed hypotheses are verified and the proposed completion methods are compared with current nonparametric completion methods by simulation studies. Simulation results show that biases of the proposed completion methods are significantly smaller than that of those in the literature.

***Keywords*:** *Forecasting, Demand, Censored, Nonparametric, Product Limit Estimator*

1. Introduction

In a newsvendor inventory system, a decision maker places an order before the selling season with stochastic demand. If too much is ordered, stock is left over at the end of the period, whereas if too little is ordered, sales are lost. The optimal order quantity is often set based on the well-known critical ratio [1], therefore demand observations often get right censored when there are lost sales and no backordering. Because lost sales cannot be observed, the available sales data actually reflect the stock available for sale, rather than the true demand. Demands for newsvendor-type products are often forecasted from censored observations.

The problem of demand forecasting in the presence of stockouts is a well-known problem of handling censored observations, which was recognized by [2]. Approaches of handling censored observations can be divided into two classes: (1) parametric method, which often assumes that the observations come from specific theoretical distribution and then estimate parameters of the assumed distribution by applying maximum likelihood estimation or some updating procedures [3]; This method is often used in density forecasting [4]; (2) nonparametric method, which is often established based on the product limit estimator [5], and attempts to address the problem of the "undefined region" beyond the largest observation when it is censored [6].

Parametric methods for demand forecasting from censored observations have been investigated in [7−14]. These works have been briefly reviewed in [15], and it has been indicated in [15] that it is difficult to determine the shape or family of demand distribution in advance when demand observations are censored.

The product limit (PL) estimator is a nonparametric maximum likelihood estimator of a distribution function based on censored data. If the largest observation is censored, the PL estimator is developed to estimate the left-hand side of demand distribution, but it is undefined for the right-hand side of distribution function. Under the assumption that there are more information besides the censored observations, Lau and Lau [3] and Zhang et al. [15] have investigated the problems of estimating the right-hand side of demand distributions.

Without additional information besides the censored observations, truncation techniques or completion methods are usually employed to define the whole distribution function. Truncation techniques are based on the data-driving rules, which include two common truncation rules: (1) truncating at the largest observation if it is censored, and (2) truncating at ($n-l$)th order statistics [6]. These truncation rules may intuitively appear to have good properties by avoiding problems in tail, but they will incur large bias because the location of the ignored

region is a random event. Completion methods aim to redefine the PL estimator beyond the largest observation if it is censored. We will briefly review nonparametric completion methods in the next section.

In this paper, we propose two hypotheses to investigate estimation bias of the PL estimator, and provide three modified completion methods based on the proposed hypotheses. The proposed hypotheses are verified and the proposed completion methods are compared with current nonparametric completion methods in the literature by simulation studies.

The remainder of this paper is structured as follows. We briefly introduce the PL estimator and review current nonparametric completion methods suggested in the literature. Then we propose two hypotheses to investigate estimation bias of the PL estimator, and provide three modified completion methods. We further verify the two hypotheses and compare the proposed completion methods with current nonparametric completion methods by simulation studies. The paper ends with some concluding remarks.

2. Nonparametric Completion Methods

In this section, we first introduce the PL estimator in the context of an inventory system, and then we review current nonparametric completion methods for the PL estimator suggested in the literature.

2.1 Product Limit Estimator

Let X_i, $i=1,2,\ldots,n$, be iid (independent identically-distributed) demand from distribution *F*, and inventory level Y_i, $i=1,2,\ldots,n$ be iid from distribution *G*. It is often to assume that both *F* and *G* are continuous and defined on the interval $[0,\infty)$. In an inventory system, demand X_i is censored on the right by the available inventory level Y_i, and we observe $Z_i = \min\{X_i, Y_i\}$ and $\delta_i = I(X_i \le Y_i)$, $i=1,2,\ldots,n$, where $I(\cdot)$ stands for the indicator function, and δ_i indicates whether demand observation Z_i is censored ($\delta_i = 0$) or not ($\delta_i = 1$).

Kaplan and Meier [5] introduced the PL estimator for the survival function $S(t) = 1 - F(t)$, which is estimated as follows:

$$\hat{S}(t) = \Pi_{i=1}^{n}\left(1-\frac{\delta_{i:n}}{n-i+1}\right)^{I(Z_{i:n}\le t)} \quad (1)$$

where $Z_{i:n}$ denotes the *i*th ordered observation among all Z_i, and $\delta_{i:n}$ corresponds to $Z_{i:n}$. From the above definition, it is observed that the PL estimator is undefined beyond the largest observation, i.e., for $t \ge Z_{n:n}$ and $\delta_{n:n} = 0$.

2.2 Review of Current Completion Methods

To overcome the shortfall of the PL estimator that it is undefined beyond the largest observation, some completion methods are suggested in the literature. Efron [16] introduced the notion of self-consistency, i.e.,

$$\hat{S}_E(t) = 0, \text{ for } t \ge Z_{n:n}. \quad (2)$$

Gill [17] defined the survival function by

$$\hat{S}_G(t) = (1-\delta_{n:n})\Pi_{i=1}^{n}\left(1-\frac{\delta_{i:n}}{n-i+1}\right)^{I(Z_{i:n}\le t)} \text{ for } t \ge Z_{n:n} \quad (3)$$

Chen and Phadia [18] modified it as

$$\hat{S}_C(t) = c(1-\delta_{n:n})\Pi_{i=1}^{n-1}\left(1-\frac{\delta_{i:n}}{n-i+1}\right)^{I(Z_{i:n}\le t)} \text{ for } t \ge Z_{n:n} \quad (4)$$

where $c \in [0,1]$ is determined by minimizing the mean squared error loss

$$E\int_0^\infty \left(\hat{F}(t) - F(t)\right)^2 dF(t) = -\int_0^\infty E\left(\hat{S}(t) - S(t)\right)^2 dS(t) \quad (5)$$

Clearly, the extreme values of scalar *c* yield Efron's and Gill's versions, respectively.

Besides the above three constant completion methods, there are two curve completion methods suggested in the literature. Brown *et al* [19] suggested an exponential completion method as follows:

$$\hat{S}_B(t) = e^{-\lambda_B t}, \text{ for } t \ge Z_{n:n}. \quad (6)$$

The parameter λ_B is set by solving $\hat{S}(Z_{n:n}^-) = e^{-\lambda_B Z_{n:n}}$, where $Z_{n:n}^- = \lim_{\varepsilon>0,\varepsilon\to 0}(Z_{n:n} - \varepsilon)$. Let $Z_{(i)}$, $i=1,\ldots m$, denote the *m* ordered uncensored demand observations, the remaining $n-m$ observations are censored ones. Moeschberger and Klein [20] attempted to complete $\hat{S}(t)$ by a two-parameter Weibull function as follows:

$$\hat{S}_M(t) = e^{-\lambda_M t^k}, \text{ for } t \ge Z_{n:n} \quad (7)$$

The two parameters λ_M and *k* in Equation (7) are determined by solving $\hat{S}(Z_{(m)}) = e^{-\lambda_M t_{(m)}^k}$ and $\hat{S}(Z_{(m-1)}) = e^{-\lambda_M t_{(m-1)}^k}$.

When a completion method is used, the bias of $\hat{S}(t)$, $B(t) = E\{\hat{S}(t)\} - S(t)$, is entirely determined by the completion method [21]. For a completion method, it is clear that the "undefined region" has the most contribution to the bias of the PL estimator. One might think that this region could be in some sense ignored, as it is sug-

gested in truncation techniques. Because the location of this region is a random event, simply ignoring the "undefined region" will result in a large bias [6].

The bias of $\hat{S}_E(t)$ is negative and asymptotically zero as $t \to \infty$, whereas the bias of $\hat{S}_G(t)$ is positive and increasing as $t \to \infty$. The bias using any other completion method will be bounded by the biases of $\hat{S}_E(t)$ and $\hat{S}_G(t)$ [6]. The bias of $\hat{S}_C(t)$ changes from negative to positive and it is increasing as $t \to \infty$. If an estimator is asymptotically zero as $t \to \infty$, we say that it has completeness, which is necessary for estimating moments of distribution. $\hat{S}_E(t)$, $\hat{S}_B(t)$, and $\hat{S}_M(t)$ have completeness since they are asymptotically zero as $t \to \infty$, whereas $\hat{S}_G(t)$ and $\hat{S}_C(t)$ do not have the completeness. The curve completion methods, $\hat{S}_B(t)$, and $\hat{S}_M(t)$ satisfy the downward sloping monotonicity of survival function, but the constant completion methods, $\hat{S}_E(t)$, $\hat{S}_G(t)$ and $\hat{S}_C(t)$ do not.

3. New Completion Methods

In this section, we first propose two hypotheses to investigate estimation bias of the PL estimator at two special points, and provide three modified completion methods based on the proposed hypotheses. Then we simplify show the nonparametric completion methods by an example.

3.1 Estimation Bias of the PL Estimator

If demand observations X_i, $i = 1, 2, \ldots, n$, are observable, then its empirical survival function $\bar{S}(t)$ can be expressed as follows:

$$\bar{S}(t) = 1 - \frac{1}{n}\sum_{i=1}^{n} I(X_i \le t) \tag{8}$$

Since $X_{n:n} > Z^-_{n:n}$, the value of $\bar{S}(t)$ at point $Z^-_{n:n}$ can be rewritten as

$$\begin{aligned}\bar{S}(Z^-_{n:n}) &= 1 - \frac{1}{n}\sum_{i=1}^{n} I(X_i \le Z^-_{n:n}) \\ &= \Pi_{i=1}^{n-1}\left(1 - \frac{1}{n-i+1}\right)^{I(X_{i:n} \le Z^-_{n:n})}\end{aligned} \tag{9}$$

According to Equation (1), the estimation value of the PL estimator at point $Z^-_{n:n}$ is

$$\hat{S}(Z^-_{n:n}) = \Pi_{i=1}^{n-1}\left(1 - \frac{\delta_{i:n}}{n-i+1}\right)^{1} \tag{10}$$

To compare $\bar{S}(Z^-_{n:n})$ and $\hat{S}(Z^-_{n:n})$, we introduce

$$\tilde{S}(Z^-_{n:n}) = \Pi_{i=1}^{n-1}\left(1 - \frac{1}{n-i+1}\right)^{1} \tag{11}$$

From Equations (9–11), $\hat{S}(Z^-_{n:n})$ can be viewed as a modification of $\bar{S}(Z^-_{n:n})$ by replacing $I(X_{i:n} \le Z^-_{n:n})$ by 1 (from Equation (9–11)), and then replacing 1 by $\delta_{i:n}$ (from Equation (11,10)). By introducing these two replacements, it is clear that $\tilde{S}(Z^-_{n:n}) \le \bar{S}(Z^-_{n:n})$ and $\tilde{S}(Z^-_{n:n}) \le \hat{S}(Z^-_{n:n})$. This indicates that $\tilde{S}(Z^-_{n:n})$ will underestimate $\bar{S}(Z^-_{n:n})$, $\hat{S}(Z^-_{n:n})$ will overestimate $\tilde{S}(Z^-_{n:n})$, but $\hat{S}(Z^-_{n:n})$ will underestimate or overestimate $\bar{S}(Z^-_{n:n})$.

The sign of bias $\hat{S}(Z^-_{n:n}) - \bar{S}(Z^-_{n:n})$ is completely determined by $I(X_{i:n} \le Z^-_{n:n})$ and $\delta_{i:n}$, $i = 1, 2, \ldots, n-1$. Since $I(X_{i:n} \le Z^-_{n:n})$ and $\delta_{i:n}$ are random variables determined by X_i and Y_i, $i = 1, 2, \ldots, n$, the bias is also a random variable and its sign also depends on X_i and Y_i, $i = 1, 2, \ldots, n$. In the case when $X_{n-1:n} \le Z^-_{n:n}$ is satisfied and there is at least one censored observation among $Z_{i:n}$, $i = 1, 2, \ldots, n-1$, $\hat{S}(Z^-_{n:n})$ must overestimate $\bar{S}(Z^-_{n:n})$. We argue that $\delta_{i:n}$ has more important influence on the estimation bias than $I(X_{i:n} \le Z^-_{n:n})$ does, i.e., the PL estimator will statistically overestimate at point $Z^-_{n:n}$. Based on this perception, we present the following hypothesis:

Hypothesis 1:

Denote by $B(Z^-_{n:n}) = \hat{S}(Z^-_{n:n}) - \bar{S}(Z^-_{n:n})$, then $B(Z^-_{n:n})$ is statistically larger than zero.

Since the PL estimator is a piecewise right continuous function, and the largest uncensored observation $Z_{(m)}$ is a right continuous piecewise point, so the relative estimation bias of the PL estimator at point $Z_{(m)}$ should be statistically smaller than that at point $Z^-_{n:n}$. That is, the PL estimator statistically provide more accurate estimation at point $Z_{(m)}$ than at $Z^-_{n:n}$. Therefore, we have the following hypothesis:

Hypothesis 2:

Denote by $R(t) = \left|\hat{S}(t) - \bar{S}(t)\right| \Big/ \bar{S}(t)$, then $R(Z_{(m)})$ is statistically smaller than $R(Z^-_{n:n})$.

3.2 Modified Completion Methods

In the spirit of the exponential curve completion method suggested by [19], we provide three modified completion methods based on the proposed hypotheses.

Hypothesis 1 implies that the PL estimator will statistically overestimate at point $Z_{n:n}^{-}$. Therefore bias can be reduced if parameter of the exponential curve is set by solving $d\cdot\hat{S}\left(Z_{n:n}^{-}\right)=e^{-\lambda_D Z_{n:n}}$ instead of $\hat{S}\left(Z_{n:n}^{-}\right)=e^{-\lambda_B Z_{n:n}}$, where $d\in(0,1]$ is an adjusted factor for overcoming the overestimation of the PL estimator at point $Z_{n:n}^{-}$. Chen and Phadia [18] proposed an optimal constant completion by setting $\hat{S}\left(Z_{n:n}\right)=c\hat{S}\left(Z_{n:n}^{-}\right)$. Similarly, we set $d=\min(2c,1)$. This parameter setting is presented because scalar *d* should not be larger than one in solving λ_D.

Hypothesis 2 indicates that the relative estimation bias of the PL estimator at point $Z_{(m)}$ is statistically smaller than that at point $Z_{n:n}^{-}$. Therefore bias can be reduced if parameter of the exponential curve is set by solving $\hat{S}\left(Z_{(m)}\right)=e^{-\lambda_L Z_{(m)}}$ instead of $\hat{S}\left(Z_{n:n}^{-}\right)=e^{-\lambda_B Z_{n:n}}$. Since the exponential curve may approximately pass the two points $\left(Z_{(m)},\ \hat{S}\left(Z_{(m)}\right)\right)$ and $\left(Z_{n:n},\ d\cdot\hat{S}\left(Z_{n:n}^{-}\right)\right)$, the parameter of the exponential curve can also be set as $\lambda_A=\left(\lambda_L+\lambda_D\right)/2$.

3.3 An Illustrative Example

In a case study of a newsvendor inventory system, Lau and Lau [3] presented 20 ordered daily sales observations: 34, 34, 37*, 38, 44*, 45*, 47*, 50, 50, 50, 60, 60*, 65* (8 times), where asterisk indicates a censored observation, e.g., the third entry '37*' means that $Z_3=37$ was observed on a day when $Y_3=37$, implying that $X_3\geq 37$.

For this example, the various aforementioned completion methods are plotted in Figure 1. From Figure 1 it can be observed that, the five curve completion methods satisfy the downward sloping monotonicity of survival function, and that the five curve completion methods and Efron's self-consistent completion have the completeness.

4. Simulation Studies

In this section, we verify the two proposed hypotheses and compare the aforementioned completion methods by simulation studies.

4.1 Simulation Experiments

In our simulation studies, we design two experiments under an inventory system with some specific distributions as follows:

Experiment 1: Following [22−23], we take demand distribution F to be a Weibull distribution, $F(x)=1-\exp\left(-x^a\right)$ for $x\geq 0$ with a=1 and 2. To reflect a variety of censoring distribution patterns, we also follow [23] to take Weibull distribution, $G(y)=1-\exp\left[-(uy)^b\right]$ for $x\geq 0$ with $b=a/2$, $b=a$, and $b=2a$, as our censoring inventory distribution. This gives the hazard rate $h(t)=u^b t^{b-a}$, which is decreasing for $b/a<1$, constant for $b/a=1$, and increasing $b/a>1$. The scale factor *u* in $G(y)$ is adjusted in such a way so that the expected stockout probability (ESP)

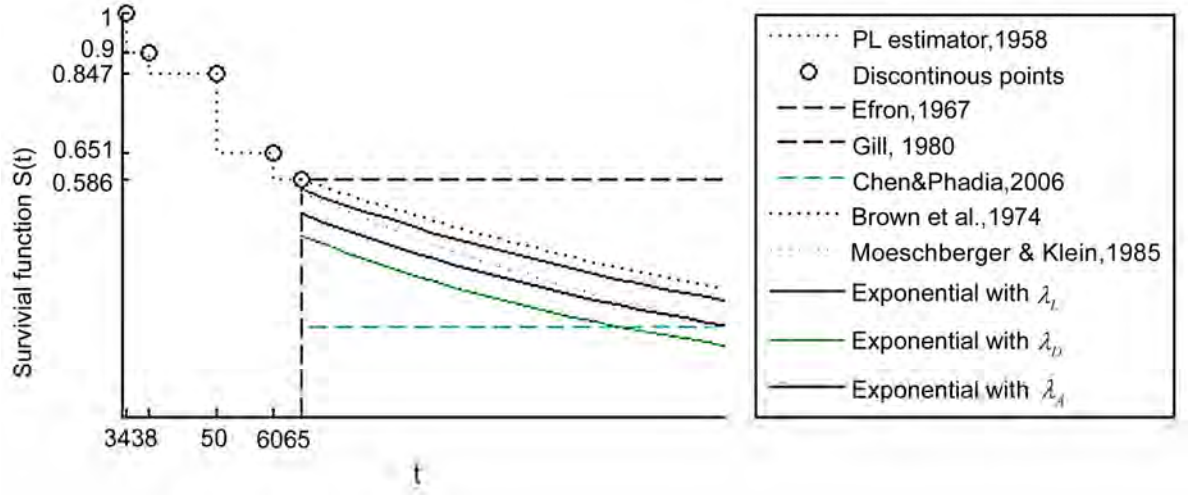


Figure 1. Comparison of the eight completion methods for the PL estimator

Table 1. Statistical results of $B\left(Z_{n:n}^{-}\right)$ in Experiment 1

		b/a =0.5		*b/a* =1		*b/a* =2	
		a=1	*a*=2	*a*=1	*a*=2	*a*=1	*a*=2
Mean of $B\left(Z_{n:n}^{-}\right)$		0.0082	0.0102	0.0269	0.0259	0.0300	0.0312
Std. Dev. of $B\left(Z_{n:n}^{-}\right)$		0.0667	0.0681	0.1075	0.1098	0.1487	0.1511
95% C.I. of $B\left(Z_{n:n}^{-}\right)$	Lower	0.0041	0.0060	0.0202	0.0191	0.0208	0.0218
	Upper	0.0123	0.0145	0.0335	0.0327	0.0392	0.0406

Table 2. Statistical results of $B\left(Z_{n:n}^{-}\right)$ in Experiment 2

		ESP =1/3		ESP =1/2		ESP =2/3	
		$\sigma=1$	$\sigma=2$	$\sigma=1$	$\sigma=2$	$\sigma=1$	$\sigma=2$
Mean of $B\left(Z_{n:n}^{-}\right)$		0.0758	0.0893	0.1637	0.1839	0.2758	0.3144
Std. Dev. of $B\left(Z_{n:n}^{-}\right)$		0.0607	0.0705	0.1053	0.1181	0.1419	0.1464
95% C.I. of $B\left(Z_{n:n}^{-}\right)$	Lower	0.0720	0.0849	0.1572	0.1766	0.2670	0.3053
	Upper	0.0796	0.0936	0.1703	0.1912	0.2846	0.3235

is 1/3, 1/2, or 2/3. These values thus completely specify the hazard rate. The reader is referred to [23] for further details. This experiment is applied for investigating the case when hazard rate is decreasing, constant or increasing.

Experiment 2: Analogous to [8], we express the relation between demand *X* and sales *Z* by writing sales as a random proportion of demand, i.e., $Z_i = W_i X_i$, where W_i is a random variable taking values on the interval [0.5,1]. For periods with no stockout, $W_i = 1$, and therefore sales and demand are equal; for periods in which a stockout has occurred, sales will be less than demand with $W_i < 1$. We assume that stockouts occur in each period (independently) with probability ESP and when a stockout occurs, sales Z_i is a random (uniformly distributed) proportion of demand X_i. In our case studies, we take *F* to be a lognormal distribution with location parameter 4, and shape parameter $\sigma = 1$ and 2, and we also set ESP=1/3, 1/2, and 2/3. This experiment is designed for investigating the case when hazard rate changes from increasing to decreasing.

In the above two experiments, we have four different cases in terms of hazard rate: decreasing, constant, increasing, and changing from increasing to decreasing. In comparison with Experiment 1, Experiment 2 makes an additional assumption on the relation between demand and sales, i.e., sales is a random (uniformly distributed) proportion of demand.

Considering the combination of the parameters in the above two experiments, under each of four cases of hazard rate, we have 6 different combinations of the parameters. Under each parameters' combination, we set the number of observations *n*=20, and randomly generate 1000 simulation runs. To ensure the applicability of the completion method suggested by Moeschberger and Klein [20], the number of uncensored observations in each simulation run is restricted to be larger than 3. For the convenience of comparison, the largest observation in each simulation run is restricted to be a censored one.

4.2 Hypotheses Verification

Under each of four cases of hazard rate, we calculate $B\left(Z_{n:n}^{-}\right)$ under 1000 simulation runs for verifying Hypothesis 1. Statistical results of $B\left(Z_{n:n}^{-}\right)$ are reported in Tables 1 and 2. In these tables, 95% C.I. is short for 95% confidence level.

Results shown in Tables 1 and 2 verify Hypothesis 1, i.e., the PL estimator statistically overestimates at point $Z_{n:n}^{-}$. Table 1 also illustrates that $B\left(Z_{n:n}^{-}\right)$ increases with the increase of *b/a*, this implies that the estimation bias of the case with increasing hazard rate is larger than that of the case with decreasing hazard case. Table 2 also illustrates that $B\left(Z_{n:n}^{-}\right)$ increases as the expected stockout probability increases.

To verify the correctness of Hypothesis 2, we calculate $R\left(Z_{n:n}^{-}\right)$ and $R\left(Z_{(m)}\right)$ under each of four cases of hazard

rate. Statistical results of $R\left(Z_{n:n}^{-}\right)$ and $R\left(Z_{(m)}\right)$ in the two experiments are reported in Tables 3 and 4, respectively. The last two rows of these two tables present results of paired 2-tailed *t*-tests on $\left(R\left(Z_{n:n}^{-}\right), R\left(Z_{(m)}\right)\right)$. From Tables 3 and 4, it is observed that the relative estimation bias of the PL estimator of point $Z_{(m)}$ is statistically smaller than that of point $Z_{n:n}^{-}$ at the 0.01 significance level. Table 4 also implies that the relative estimation biases of the PL estimator at points $Z_{(m)}$ and $Z_{n:n}^{-}$ increase with the increase of the expected stockout probability.

4.3 Comparison with Current Completion Methods

In this subsection, we assess performance of the proposed completion methods in terms of integral absolute bias, $\text{IAB} = -\int_0^{\infty} E\left|\hat{S}(t) - S(t)\right| dS(t)$. In our simulation results, Efron and Gill denote Efron's and Gill's completion methods respectively; CP, BHK and MK stand for the completion methods of Chen and Phadia [18], Brown *et al* [19], and Moeschberger and Klein [20], respectively; Left, Down and Ave represent the proposed exponential curve completion methods with parameter λ_L, λ_D and λ_A, respectively.

Results of paired 2-tailed *t*-tests on IAB among the compared eight completion methods under each of four cases of hazard rate are shown in Tables 5–8, respectively. These tables report *t*-statistics on IAB between the row method and column method. One negative *t*-statistic in these tables means that the row method is better than the corresponding column method in terms of IAB, whereas positive *t*-statistic implies that the column method is better than the corresponding row method. *t*-statistic in parentheses represents that the comparison is at the 0.05 significance level; *t*-statistic in square brackets implies that there is no significant difference between the row and column methods; *t*-statistic without parentheses or square brackets expresses that the comparison is at the 0.01 significance level.

According to the following results shown in Tables 5–8, we come to the following conclusions in terms of IAB: (1) Ave is the leading completion method; (2) Left performs better than the optimal constant completion method CP; (3) CP is always better than the current curve completion methods (i.e., BHK and MK); (4) Efron and Gill are the two worst completion methods.

Table 3. Statistical results of $R\left(Z_{n:n}^{-}\right)$ and $R\left(Z_{(m)}\right)$ in Experiment 1

	b/a =0.5		*b/a* =1		*b/a* =2	
	a=1	*a*=2	*a*=1	*a*=2	*a*=1	*a*=2
Mean of $R\left(Z_{n:n}^{-}\right)$	0.6077	0.6348	0.8337	0.7906	0.7842	0.7854
Std. Dev. of $R\left(Z_{n:n}^{-}\right)$	0.6497	0.6999	1.0403	1.0034	1.2163	1.1004
Mean of $R\left(Z_{(m)}\right)$	0.4161	0.4077	0.3987	0.3914	0.3592	0.3609
Std. Dev. of $R\left(Z_{(m)}\right)$	0.3011	0.3367	0.3524	0.3743	0.3295	0.3158
t-statistics	9.4889	10.3945	13.1338	12.6988	11.1364	12.0961
P-value	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Table 4. Statistical results of $R\left(Z_{n:n}^{-}\right)$ and $R\left(Z_{(m)}\right)$ in Experiment 2

	ESP =1/3		ESP =1/2		ESP =2/3	
	$\sigma = 1$	$\sigma = 2$	$\sigma = 1$	$\sigma = 2$	$\sigma = 1$	$\sigma = 2$
Mean of $R\left(Z_{n:n}^{-}\right)$	1.3879	1.6892	2.8948	3.4150	4.6486	5.7690
Std. Dev. of $R\left(Z_{n:n}^{-}\right)$	1.1267	1.3526	2.0145	2.3014	2.8261	2.9991
Mean of $R\left(Z_{(m)}\right)$	0.7473	0.9011	1.3246	1.5836	1.7312	2.0446
Std. Dev. of $R\left(Z_{(m)}\right)$	0.5737	0.6427	0.9715	1.0729	1.2893	1.3821
t-statistics	17.8034	18.6759	23.6086	23.1638	29.4864	33.4833
P-value	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Table 5. Results of paried 2-tail *t*-test on IAB in Experiment 1 with b/a=0.5

	Gill	CP	BHK	MK	Left	Down	Ave
Efron	6.1797	55.3806	30.5818	29.3516	45.4545	42.4871	47.1257
Gill		28.3342	34.3206	7.2690	27.5968	29.7606	29.1192
CP			(-2.0951)	-35.4652	3.6572	6.7757	9.1816
BHK				-19.0436	4.2341	8.7696	7.6550
MK					32.7036	30.0723	35.1446
Left						2.7375	11.0712
Down							(2.4231)

Table 6. Results of paried 2-tail *t*-test on IAB in Experiment 1 with b/a=1

	Gill	CP	BHK	MK	Left	Down	Ave
Efron	-13.6287	65.0251	21.7673	31.3227	51.5355	44.4413	52.3274
Gill		44.8327	51.0237	27.3941	45.5583	47.4534	46.4081
CP			-15.5159	-34.5069	6.0864	(2.5103)	11.0425
BHK				-5.5002	19.4179	24.3510	22.8946
MK					32.5922	27.1247	34.7498
Left						-3.4243	9.5686
Down							10.3352

Table 7. Results of paried 2-tail *t*-test on IAB in Experiment 1 with b/a=2

	Gill	CP	BHK	MK	Left	Down	Ave
Efron	-19.4571	91.0080	32.3403	41.2147	65.5794	65.9205	68.7712
Gill		57.9369	68.1601	37.2278	60.1003	62.0498	61.1882
CP			-16.7540	-39.6581	7.4145	14.3985	15.1355
BHK				-7.6101	23.9762	31.2396	28.7295
MK					36.1403	39.3110	41.0343
Left						7.4387	17.7037
Down							[1.6798]

Table 8. Results of paried 2-tail *t*-test on IAB in Experiment 2

	Gill	CP	BHK	MK	Left	Down	Ave
Efron	-43.8358	(2.3348)	-18.9354	(-2.3319)	19.2388	-3.3448	16.5648
Gill		48.8904	45.7360	44.1637	46.6532	45.9931	47.0639
CP			-33.3675	-4.8638	18.7680	-12.9462	19.9046
BHK				19.8107	31.3266	35.5785	34.2751
MK					21.2063	[-1.7481]	19.4924
Left						-21.9985	(-2.4131)
Down							26.5872

5. Conclusions

Demands for newsvendor-type products are often forecasted from censored observations. The Kaplan-Meier product limit estimator is the well-known nonparametric method to deal with censored data, but it is undefined beyond the largest observation. In this paper, we propose two hypotheses to investigate estimation bias of the PL estimator, and provide three modified completion methods based on the proposed hypotheses.

Simulation results show that biases of the proposed completion methods are significantly smaller than that of the completion methods in the literature. According to these results, we know that the proposed completion methods can improve demand forecasting with right censored observations. We also show that simulation is a useful way to verify probability result which is difficult to be proved by using classical statistical theory and methods.

The developed methods are easy to implement in software packages. Many forecasting techniques have been integrated into enterprise software packages such as management information systems, enterprise resources planning systems, decision support systems. The proposed forecasting techniques in this paper are simple and easily implemented in enterprise software packages.

6. Acknowledgements

This work is supported by national Natural Science Foundation of China (No. 70801065).

REFERENCES

[1] B. Zhang, X. Xu, and Z. Hua, "A binary solution method for the multi-product newsboy problem with budget constraint," International Journal of Production Economics, Vol. 117, No. 1, pp. 136–141, January 2009.

[2] G. Hadley and T. M. Whitin, "Analysis of inventory systems," Prentice-Hall, Englewood Cliffs, 1963.

[3] H. S. Lau and A. H. L. Lau, "Estimating the demand distributions of single-period items having frequent stockouts," European Journal of Operational Research, Vol. 92, No. 2, pp. 254–265, July 1996.

[4] Z. Hua and B. Zhang, "Improving density forecast by modeling asymmetric features: An application to S&P500 returns," European Journal of Operational Research, Vol. 185, No. 2, pp. 716–725, March 2008.

[5] E. L. Kaplan and P. Meier, "Nonparametric estimation from incomplete observations," Journal of the American Statistical Association, Vol. 53, No. 282, pp. 457–481, June 1958.

[6] B. Gillespie, J. Gillespie, and B. Iglewicz, "A comparison of the bias in four versions of the product-limit estimator," Biometrika, Vol. 79, No. 1, pp. 149–155, March 1992.

[7] S. A. Conrad, "Sales data and the estimation of demand," Operational Research Quarterly, Vol. 27, No. 1, pp. 123–127, 1976.

[8] W. Wecker, "Predicting demand from sales data in the presence of stockouts," Management Science, Vol. 24, No. 10, pp. 1043–1054, June 1978.

[9] D. B. Braden and M. Freimer, "Informational dynamics of censored observations," Management Science, Vol. 37, No. 11, pp. 1390–1404, November 1991.

[10] N. S. Agrawal and A. Smith, "Estimating negative binomial demand for retail inventory management with unobservable lost sales," Naval Research Logistics, Vol. 43, No. 6, pp. 839–861, September 1996.

[11] P. C. Bell, "Adaptive sales forecasting with many stockouts," Journal of the Operational Research Society, Vol. 32, No. 10, pp. 865–873, October 1981.

[12] P. C. Bell, "A new procedure for the distribution of periodicals," Journal of the Operational Research Society, Vol. 29, No. 5, pp. 427–434, May 1978.

[13] P. C. Bell, "Forecasting demand variation when there are stockouts," Journal of the Operational Research Society, Vol. 51, No. 3, pp. 358–363, March 2000.

[14] X. Ding, "Estimation and optimization in discrete inventory models," Ph.D. thesis, The University of British Columbia, Vancouver, Canada, 2002.

[15] W. Zhang, B. Zhang, and Z. Hua, "Quasi-bootstrap procedure for forecasting demand from sales data with stockouts," in Proceedings of the 38th International Conference on Computers and Industrial Engineering, Vol. 1–3, pp. 423–429, October 2008.

[16] B. Efron, "The two sample problem with censored data," in Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, Vol. 4, pp. 831–852, 1967.

[17] R. D. Gill, "Censoring and stochastic integrals," Mathematical Centre Tract No. 124, Mathematisch Centrum, Amsterdam, 1980.

[18] Z. Chen and E. Phadia, "An optimal completion of the product limit estimator," Statistics & Probability Letters, Vol. 76, No. 9, pp. 913–919, May 2006.

[19] B. W. Brown, M. Jr. Hollander, and R. M. Korwar, "Nonparametric tests of independence for censored data, with applications to heart transplant studies," in: F. Proschan and R. J. Serfling, (Eds.), "Reliability and biometry: statistical analysis of lifelength," Philadelphia: Society for Industrial and Applied Mathematics, pp. 291–302, 1974.

[20] M. L. Moeschberger and J. P. Klein, "A comparison of several methods of estimating the survival function when there is extreme right censoring," Biometrics, Vol. 41, No. 1, pp. 253–259, March 1985.

[21] P. Meier, "Estimation of a distribution function from incomplete observations," in: J. Gani (Eds.), "Perspectives in probability and statistics," Applied Probability Turst, Sheffield, England, pp. 67–87, 1975.

[22] J. H. J. Geurts, "Some small-samplel nonproportional hazards results for the Kaplan-Meier estimator," Statistica Neerlandica, Vol. 39, No. 1, pp. 1–13, March 1985.

[23] J. H. J. Geurts, "On the small-sample performance of Efron's and of Gill's version of the product limit estimator under nonproportional hazards," Biometrics, Vol. 43, No. 3, pp. 683–692, September 1987.

J. Software Engineering & Applications, 2009, 2: 267-275
doi:10.4236/jsea.2009.24034 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

Secure Chained Threshold Proxy Signature without and with Supervision*

Zoe L. JIANG, S. M. YIU, Y. DONG, L. C. K. HUI, S. H. Y. WONG

Department of Computer Science, The University of Hong Kong, Hong Kong, China.
Email: {ljiang, smyiu, ydong, hui, shywong}@cs.hku.hk.

Received June 4th, 2009; revised July 15th, 2009; accepted July 24th, 2009.

ABSTRACT

Threshold Proxy Signature (TPS) scheme facilitates a manager to delegate his signing capability to a group of n_2 subordinates without revealing his own private key, such that a subgroup of at least $t_2 \leqslant n_2$ subordinates is required to generate a proxy signature. In reality, the situation can be more complicated. First of all, the subgroup may further delegate their proxy signing capabilities to another group of n_3 subordinates such that at least another subgroup of at least $t_3 \leq n_3$ subordinates are of the proxy signing capabilities (in the form of a chain). t_2 can be unequal to t_3 depending on the concrete requirement. This is a group-to-group delegation problem. In addition, a supervising agent (SA) may be introduced in the above chain to supervise the subordinates, such that proxy signing can only be successfully executed with SA's agreement. This is a delegation with supervision problem in the threshold delegation chain described above. These two extensions of delegation problems are not solved yet. This paper designs two provably secure cryptographic schemes Chained Threshold Proxy Signature (CTPS) scheme and Chained Threshold Proxy Signature with Supervision (CTPSwS) scheme to solve these two delegation problems.

***Keywords*:** *Delegation, Threshold Proxy Signature, Chained Threshold Proxy Signature with Supervision*

1. Introduction

1.1 Motivation

It is a common practice for a manager to delegate his signing right to a group of n subordinates when he is on leave so that a subgroup of at least $t \leq n$ of them can cooperate to sign a document on behalf of the manager. This is a threshold delegation problem, and can be solved by Threshold Proxy Signature (TPS) [2] scheme. In reality, the delegation may involve more than one level (in the form of a chain). Consider the following scenario. There is an email sent by the manager of software development department in corporation A, Simon, to the manager of the same department in corporation B, Sam. Since Sam is too busy to check every single detail of the data part before he replies with his signature, and the data are so important that he cannot rely on any single one of his three vice managers (his subordinates), he forwards this email to all of them. Further any two of them, as a subgroup, may forward this email to their employees checking the data part. As a result, to answer this email to Simon, it is desirable for a subgroup of the employees to cooperate on behalf of Sam. How the any two of the vice managers pass their proxy signing capabilities to their employees is referred to group-to-group delegation problem. In a more cautious situation, the contract part needs to be authorized by the manager of the software maintenance department in corporation B, Steven. In this case, besides delegation, Sam is also required to appoint Steven as his supervising agent (SA) such that only when Steven agrees to the contract part, the employees can compute a proxy signature on behalf of Sam. How Sam appoints Steven as an SA is referred to delegation with supervision problem in threshold delegation chain. In this paper, we propose two schemes, Chained Threshold Proxy Signature (CTPS) scheme and Chained Threshold Proxy Signature with Supervision (CTPSwS) scheme, to solve these two delegation problems.

1.2 Related work

Mambo *et al.* [3] introduced the first efficient proxy signature in 1996, where it allows a user to delegate his signing power to a designated signer, a proxy signer. It is widely applicable in all kinds of known standard signature schemes such as El Gamal scheme [4], Okamoto scheme [5] and Fiat-Shamir scheme [6]. In 1997, Kim *et al.* [2] proposed proxy signature for partial delegation with warrant combining the benefit of Mambo's partial delegation and Neuman's [7] delegation by warrant/certi-

*The preliminary work has been published in 2008 International Conference on Computer Science and Software Engineering [1].

ficate. They also extended it to a (t, n) Threshold Proxy Signature (TPS) such that any $t \leq n$ proxy signers using their proxy secret key shares can cooperate to generate a proxy signature on behalf of an original signer, but less than t can not, by deploying Ceredo's Schnorr type threshold digital signature scheme [8].

Lui *et al*. [9] proposed a chained delegation scheme with supervision scheme, in which the original signer sends, in ahead of time, his permission information about the proxy signer to a supervising agent (SA) who he trusts. Then the proxy signer can only generate a valid proxy signature under SA's supervision even when the original signer is not available. This delegation can be executed in multiple levels. However, on one hand, the scheme does not consider the threshold problem. On the other hand, the scheme sacrificed both the original signer and the supervising agent's undeniabilities due to the advantage the authors presented that there is no need for the verifier to be aware of whether supervision is performed or not. In many cases, this is unacceptable from the security point of view.

Boldyreva [10] defined a formal proof model for the security of proxy signature schemes, which enables the cryptographic analysis of such schemes, instead of just presenting attacks that fail. Then they proved the security of Triple Schnorr Proxy Signature scheme, a variant of Kim at al.'s proxy signature scheme, preserving its efficiency, in the random oracle model assuming the hardness of computation of discrete logarithm.

1.3 Our Contribution

Firstly, in this paper we propose Chained Threshold Proxy Signature (CTPS) scheme to solve the group-to-group delegation problem. Although, Threshold Proxy Signature (TPS) scheme [2] and Triple Schnorr Signature scheme [10] are two important components to design our scheme, we need to consider how to distribute the proxy shares from a subgroup of vice managers to another subgroup of employees in a group-to-group manner. Therefore, we deploy Herzberg *et al*.'s [11] proactive secret sharing idea into our scheme. Proactive secret sharing is proposed to periodically renew the shares without changing the secret, in such a way that any information learnt by the adversary about individual shares becomes obsolete after the shares are renewed. But in our scheme, the renewed shares should be securely passed to employees by each vice manager while old ones are kept secret by the vice managers themselves.

To solve the delegation with supervision problem in threshold delegation chain, we adapt Lui *et al*'s supervision idea in delegation chain (no threshold) into CTPS scheme to implement Chained Threshold Proxy Signature with Supervision (CTPSwS) scheme. Different with Lui *et al*'s idea, however, supervising agent is also actively involved in the delegation using his/her own private key, such that verification for the proxy signature requires supervising agent's public key as well, besides original signer and proxy signers' public keys.

We also provide formal security models and proofs to show that the schemes we designed are secure in the random oracle model assuming the hard problem of discrete logarithm.

1.4 Organization

A 3-level CTPS scheme and its security proof will be described in section 2 & 3, respectively. Then a 3-level CTPSwS scheme and security proof draft will be introduced in section 4 & 5, respectively. Section 6 concludes the paper and discusses some future work.

2. Chained Threshold Proxy Signature (CTPS)

Recall that Sam, the manager of software development department in corporation B, delegates his signing right to a group of n_2 vice managers, any subgroup of whom with t_2 members further delegate their proxy signing capabilities to a group of n_3 employees, such that any subgroup of t_3 employees can reply to Simon with their proxy signature on behalf of Sam.

We can formally define the above roles by letting Sam the original signer u_1 in level 1, vice managers a group of n_2 proxy signers in level 2, ($\{u_{2,j}\}_{n_2}$ for short), and employees a group of n_3 proxy signers in level 3, ($\{u_{3,k}\}_{n_3}$ for short). Any subgroup of $t_2 \leq n_2$ vice managers performing the delegation is defined as U_2. Similarly, any subgroup of $t_3 \leq n_3$ employees signing the replied email is defined as U_3. Note the difference between $\{u_{2,j}\}_{n_2}$ and U_2, $\{u_{3,k}\}_{n_3}$ and U_3. WLOG, we assume $U_2 = \{u_{2,1}, u_{2,2}, \cdots, u_{2,t_2}\} = \{u_{2,j}\}_{t_2}$ and $U_3 = \{u_{3,1}, u_{3,2}, \cdots, u_{3,t_3}\} = \{u_{3,k}\}_{t_3}$. Let ($sk_1$,$pk_1$), ($sk_{2,j}$, $pk_{2,j}$), (skg_2, pkg_2), ($sk_{3,k}$, $pk_{3,k}$) and (skg_3, pkg_3) denote u_1, $u_{2,j}$, $\{u_{2,j}\}_{n_2}$, $u_{3,k}$, and $\{u_{3,k}\}_{n_3}$'s secret/public key pairs respectively. By extending Boldyreva's proxy signature scheme model, CTPS scheme to achieve the delegation procedure should involve a one-to-group protocol run between the original signer and the group of proxy signers in level 2, a group-to-group protocol run between any subgroup of proxy signers in level 2 and the group of proxy signers in level 3, a chained threshold proxy signing algorithm and the corresponding verification algorithm. Additionally, there should be an algorithm that extracts the identities of the groups of proxy signers in both level 2 & 3.

Definition 1 describes the detailed components of a 3-level Chained Threshold Proxy Signature scheme. A list of important parameters and symbols is shown here for your reference.

ω_1: warrant including u_1 and $\{u_{2,j}\}_{n_2}$'s IDs, and other

information on the delegation

ω_2: warrant including $\{u_{2,j}\}_{n_2}$ and $\{u_{3,k}\}_{n_3}$'s IDs, and other information on the delegation

skt: secret key transformation generated by u_1

$skt_{1,j}$: share of secret key transformation sent to $u_{2,j}$

$skp_{2,j}$: proxy secret key share generated by $u_{2,j}$

$skt_{2,j}$: share of proxy secret key transformation generated by $u_{2,j}$

$skt_{2,j,k}$: sub-share of proxy secret key transformation sent to $u_{3,k}$

$skt'_{2,k}$: share of proxy secret key transformation retrieved by $u_{3,k}$

$skp_{3,k}$: proxy secret key share generated by $u_{3,k}$

$p\sigma_3$: chained threshold proxy signature.

Definition 1 (CTPS Scheme) *Let* ***CTPS*** = (*G, K, (TD, TP), (CTD, CTP), CTPS, CTPV, CTPID*) *be a chained threshold proxy signature scheme, where the constituent algorithms run in polynomial time*.

G is a random *parameter-generation* algorithm, and it will output some global parameters params.

K is a random *key-generation* algorithm, and it will output secret/public key pairs for original signer u_1 and proxy signers in both level 2 & 3, $\{u_{2,j}\}_{n_2}$ and $\{u_{3,k}\}_{n_3}$, in the scheme.

(*TD, TP*) is a *Threshold Designation-Proxy* protocol between the original signer u_1 and the proxy signers in level 2, $\{u_{2,j}\}_{n_2}$. Both *TD* and *TP* take as input the public keys pk_1 and pkg_2, respectively. *TD* also takes as input the secret key sk_1 of u_1, and *TP* also takes as input the secret key $sk_{2,j}$ of $u_{2,j}$. As the result of the interaction, the expected local output of *TP* is $skp_{2,j}$, the proxy secret share which is kept secret by each $u_{2,j}$.

$$[TD\ (pk_1, pkg_2, sk_1), TP\ (pk_1, pkg_2, sk_{2,j})] \rightarrow skp_{2,j}$$

(*CTD, CTP*) is a *Chained Threshold Designation-Proxy* protocol between U_2 and $\{u_{3,k}\}_{n_3}$. Both *CTD* and *CTP* take as input the public keys pk_1, pkg_2 and pkg_3, respectively. *CTD* also takes as input the proxy secret shares $\{skp_{2,j}\}_{t_2}$ of $\{u_{2,j}\}_{t_2}$. *CTP* takes as input the secret key $sk_{3,k}$ of $u_{3,k}$. The expected local output of *CTP* is $skp_{3,k}$, the proxy secret key share which is kept secret by each $u_{3,k}$. Note that for each $u_{3,k}$ to generate $skp_{3,k}$, the subgroup U_2 is involved in *CTD*, but not just a certain proxy signer in U_2.

$$[CTD\ (pk_1, pkg_2, pkg_3, \{skp_{2,j}\}_{t_2}),$$
$$CTP\ (pk_1, pkg_2, pkg_3, sk_{3,k})] \rightarrow skp_{3,k}$$

CTPS is the (possibly) randomized *Chained Threshold Proxy Signing* algorithm. It takes as input $\{skp_{3,k}\}_{t_3}$ and a message $M \in \{0,1\}^*$, and outputs a chained threshold-proxy signature $p\sigma_3$.

$$CTPS\ (M, \{skp_{3,k}\}_{t_3}) \rightarrow p\sigma_3$$

CTPV is the deterministic *Chained Threshold Proxy Verification* algorithm. It takes as input a message *M*, a proxy signature $p\sigma_3$, and (pk_1, pkg_2, pkg_3), and outputs 0 or 1.

$$CTPV\ (M, p\sigma_3, pk_1, pkg_2, pkg_3) = 0/1$$

CTPID is the *Chained Threshold Proxy IDentification* algorithm. It takes as input a valid proxy signature $p\sigma_3$ and outputs identities of two proxy signer groups, i.e., public keys.

$$CTPID\ (p\sigma_3) = (pkg_2, pkg_3)/\perp$$

SIGNATURE VERIFICATION CONDITION: If *CTPV* =1 and *CTPID* = (pkg_2, pkg_3), we say $p\sigma_3$ is a valid chained threshold proxy signature by proxy signers in U_2 and U_3 on behalf of u_1.

The definition clearly describes what kinds of individual algorithms and interactive protocols are required to be run by original signer and proxy signers. After define the structure of CTPS scheme, we design a concrete scheme based on the definition.

We give a high-level description of our scheme here, followed by the concrete calculation. First of all, by using public parameters and secret/public key pairs generated through *G* and *K*, u_1 generates the certificate of warrant ω_1 in (1), which is actually a signature of ω_1 using sk_1. We call it secret key transformation skt_1 in our scheme since it masks u_1's secret key sk_1 and will be used for $\{u_{2,j}\}_{n_2}$ to generate proxy signing keys. In order to designate $\{u_{2,j}\}_{n_2}$ as u_1's threshold proxy signers, each share $skt_{1,j}$ generated by (2) will be distributed to $u_{2,j}$ securely. After verifying $skt_{1,j}$ as a signature generated by u_1 using (3), each $u_{2,j}$ computes proxy secret key share $skp_{2,j}$ in (4).

As the applications we described above, if any $t_2 \le n_2$ vice managers, such as U_2, want to further delegate their proxy signing capabilities to their employees $\{u_{3,k}\}_{n_3}$, which we call a group-to-group delegation, each $u_{2,j} \in U_2$ computes secret key transformation $skt_{2,j}$ in (5) as u_1 does in (1), then divides it to n_3 shares, $skt_{2,j,k}$(k=1,2, ⋯ ,n_3), as calculated in (6), which are sent to $u_{3,k}$ securely. As a result, each $u_{3,k}$ verifies $skt_{2,j,k}$(k=1,2, ⋯ ,t_2) he receives using (8) and computes $skt'_{2,k}$ by accumulating them in (9). By comparing (5) and (9), $skt_{2,j}$ and $skt'_{2,k}$ are generated by two different random polynomials $F_2(j)$ and $F'_2(k)$ with same constant $sktg_2$. For how (9) is deduced, please refer to Lagrange Formula, which was also used in [2]. Then each $u_{3,k}$ can successfully generate the proxy secret key share $skp_{3,k}$ in (11).

Let us discuss a little more about the difference and difficulty of (*CTD, CTP*) protocol compared with (*TD, TP*) here. During (*TD, TP*) protocol, $skt_{1,j}$ carrying secret information sk_1 inside is generated and delivered to $u_{2,j}$ as the mark for u_1 to designate $u_{2,j}$ as one of his proxy signers. Similarly in (*CTD, CTP*) protocol, $skt_{2,j}$ carrying secret information $sk_{2,j}$ should also be delivered to $\{u_{3,k}\}_{n_3}$, but in an indirect way for the reason that there are a group of delegators and a group of delegatees. Sending $skt_{2,j}$ to $u_{3,k}$ where $j = k$ one by one does not work because U_2 and $\{u_{3,k}\}_{n_3}$ may have different numbers t_2 and n_2. However, from the group point of view, we need a scheme to reshuffle $\{skt_{2,j} = F_2(j)(j = 1,2, \cdots, t_2)\}$ to $\{skt'_{2,k} = F'_2(k)(k = 1, 2, \cdots, n_3)\}$, satisfying that $F_2(0) = F'_2(0) = sktg_2$. It seems that we keep the group secret key transformation $sktg_2$ unchanged and make secret key transformation shares updated. With this purpose, we found a good candidate of proactive secret sharing approach [11], which was proposed to periodically renew the shares, like $\{skt_{2,j}\}_{t_2}$, to the new ones, like $\{skt'_{2,k}\}_{n_3}$, without changing the secret, like $sktg_2$. However, in our scheme, the renewed ones belong to $\{u_{3,k}\}_{n_3}$, but not $\{u_{2,j}\}_{n_2}$.

Schnorr signature scheme [12] is used in *CTPS* and *CTPV* where partial proxy signatures $\{ps_{3,k}\}_{t_3}$ are published among U_3 such that each $u_{3,k} \in U_3$ can calculate proxy signature $p\sigma_3$ in (13). The following details how the algorithms and protocols are performed.

The system runs *G*. On input 1^k, params = $G(1^k)$ = (p,q,g,G,H), such that $2^{k-1} \le p < 2^k$, $q|p-1$, $g \in Z_p^*$ of order q, two hash functions G: $\{0,1\}^* \to Z_q$, and H : $\{0,1\}^* \to Z_q$.

The system runs *K*. On input (p,q,g,G,H), *K* generates $sk_1 \in_R Z_q$, $pk_1 = g^{sk_1} \bmod p$, $sk_{2,j} = SK_2(j)$ and $pk_{2,j} = g^{sk_{2,j}} \bmod p$, where $SK_2(x)$ is a random polynomial with random constant skg_2 and degree $t_2 - 1$. $pkg_2 = g^{skg_2} \bmod p$. Note that although the intuitive idea for the above procedure is to generate $SK_2(x)$ and distribute $sk_{2,j}$ to $u_{2,j}$ securely by a trusted dealer, we deploy the protocol for generating random number in [2] to implement it without a dealer for security consideration. As a result, each $u_{2,j}$ can only calculate his own secret key $sk_{2,j}$ without knowing skg_2. skg_3, pkg_3, $\{sk_{3,k}\}_{n_3}$ and $\{pk_{3,k}\}_{n_3}$ are generated in the same way.

u_1 runs *TD*.

$$skt_1 = e_1 \cdot sk_1 + k_1 \bmod q, \text{ where} \quad (1)$$

$$r_1 = g^{k_1} \bmod p, k_1 \in_R Z_q^*,$$

$$e_1 = G(0 \| pk_1 \| pkg_2 \| \omega_1, r_1).$$

$$skt_{1,j} = F_1(j) = skt_1 + a_{1,1}j + a_{1,2}j^2 + \cdots + a_{1,t_2-1}j^{t_2-1} \bmod q. \quad (2)$$

$F_1(j)$ is a random polynomial privately owned by u_1. r_1 and $\{g^{a_{1,m}} \bmod p\}_{t_2-1}$ are broadcast.

$u_{2,j}$ runs *TP*.

$$g^{skt_{1,j}} = (pk_1^{e_1} r_1) \cdot A \bmod p, \text{ where} \quad (3)$$

$$A = \prod_{m=1}^{t_2-1} (g^{a_{1,m}})^{j^m} \bmod p.$$

$$skp_{2,j} = e_1 \cdot sk_{2,j} + skt_{1,j} \bmod q. \quad (4)$$

$u_{2,j}$ runs *CTD*.

$$\begin{aligned} skt_{2,j} &= e_2 \cdot skp_{2,j} + k_{2,j} \bmod q \\ &= e_2(e_1 \cdot SK_2(j) + F_1(j)) + K_2(j) \\ &= F_2(j), \text{ where} \end{aligned} \quad (5)$$

$$e_2 = G(0 \| pkg_2 \| pkg_3 \| \omega_2, r_2),$$

$$F_2(0) = e_2(e_1 \cdot skg_2 + skt_1) + k_2 = sktg_2.$$

$$skt_{2,j,k} = F_{2,j}(k) \bmod q, \text{ where} \quad (6)$$

$$F_{2,j}(k) = skt_{2,j} + b_{2,j,1}k + b_{2,j,2}k^2 + \cdots + b_{2,j,t_3-1}k^{t_3-1} \bmod q. \quad (7)$$

$\{k_{2,j}\}_{n_2}$ are generated by running the protocol for generating random number in [2] such that each $u_{2,j}$ can calculate $k_{2,j}$ without knowing any other's secret shares $\{k_{2,x}\}_{x \ne j}$ and satisfying that $k_{2,j} = K_2(j)$, where $K_2(j)$ is a random polynomial with constant kg_2 and degree $t_2 - 1$. $F_{2,j}(k)$ is a random polynomial privately owned by each $u_{2,j}$ with constant $skt_{2,j}$ and degree $t_3 - 1$. $\{r_{2,j} = g^{k_{2,j}} \bmod p\}_{t_2}$ and $r_2 = g^{kg_2} \bmod p$ are broadcast.

$u_{3,k}$ runs *CTP*.

$$g^{skt_{2,j,k}} = \left(\left(pk_1 pk_{2,j} \right)^{e_1} r_1 A \right)^{e_2} r_{2,j} B, \text{ where} \quad (8)$$

$$B = \prod_{m=1}^{t_3-1} (g^{b_{2,j,m}})^{k^m} \bmod p$$

$$\begin{aligned} skt'_{2,k} &= \sum_{j=1}^{t_2} \delta_j skt_{2,j,k} = \sum_{j=1}^{t_2} \delta_j F_{2,j}(k) \\ &= F'_2(k), \text{where} \end{aligned} \quad (9)$$

$$F'_2(0) = \sum_{j=1}^{t_2} \delta_j F_{2,j}(0) = \sum_{j=1}^{t_2} \delta_j skt_{2,j} = sktg_2 \quad (10)$$

$$\delta_j = \prod_{l=1, l \ne j}^{t_2} -l / j - l$$

$$skp_{3,k} = e_2 \cdot sk_{3,k} + skt'_{2,k} \bmod q. \quad (11)$$

U_3 runs *CTPS*.

$$ps_{3,k} = c_3 \cdot skp_{3,k} + k_{3,k}, \text{ where} \quad (12)$$

$$c_3 = H(1 \| M \| pkg_2 \| pkg_3 \| \omega_2 \| r_1, r_2), \text{ and}$$

$$r_3 = g^{k_3} \bmod p.$$

$$p\sigma_3 = \sum_{u_{3,k} \in U_3} \delta_k \cdot ps_{3,k}, \quad (13)$$

$$\delta_k = \prod_{l=1,l\neq k}^{t_3} -l \big/ k-l$$

Verifier runs *CTPV*.

$$g^{p\sigma_3} = PKP^{c_3} \cdot r_3 \bmod p, \text{ where} \quad (14)$$

$$PKP = \left(pk_1 \cdot pkg_2\right)^{e_1} \cdot r_1^{e_2} \cdot r_2 \cdot pkg_3^{e_2}.$$

PROOF OF (14) Let θ_3 represents $u_{3,k} \in U_3$.

$$\begin{aligned}
p\sigma_3 &= \sum_{\theta_3} \delta_k ps_{3,k} = \sum_{\theta_3} \delta_k (c_3 + skp_{3,k} + k_{3,k}) \\
&= c_3 \sum_{\theta_3} \delta_k skp_{3,k} = \sum_{\theta_3} \delta_k k_{3,k} \\
&= c_3 \sum_{\theta_3} \delta_k (e_2 sk_{3,k} + skt'_{2,k}) + k_3 \\
&= c_3 (e_2 \sum_{\theta_3} \delta_3 sk_{3,k} + \sum_{\theta_3} \delta_k skt'_{2,k}) + k_3 \\
&= c_3 (e_2 skg_3 + \sum_{\theta 2} \delta_j skt_{2,j}) + k_3 \\
&= c_3 (e_2 skg_3 + \sum_{\theta 2} \delta_j (e_2 skp_{2,j} + k_{2,j})) + k_3 \\
&= c_3 (e_2 skg_3 + e_2 \sum_{\theta 2} \delta_j skp_{2,j} + k_2) + k_3 \\
&= c_3 (e_2 skg_3 + e_2 (e_1 (skg_2 + sk_1) + k_1) + k_2) + k_3 \bmod q, \\
L.H.S. &= g^{p\sigma_3} \\
&= g^{c_3 (e_2 skg_3 + e_2 (e_1 (skg_2 + sk_1) + k_1) + k_2) + k_3} \bmod p \\
&= R.H.S.
\end{aligned}$$

The proof proves the correctness of our CTPS scheme from the computation point of view. In the following section, we will prove its security.

3. Security of the CTPS Scheme

In the section, we set up CTPS security model and prove that our CTPS scheme is secure against adaptive chosen-message attack in random oracle model.

As discussed in [10], the formal model includes a rather powerful adversary who is able to corrupt all other users' secret keys except the *Single Honest User* (*SHU*). Furthermore, A is of the capability to launch adaptive chosen-message attacks according to three kinds of roles that the *SHU* can play, namely *Role 1*: the original signer u_1, *Role 2*: one of the required proxy signers in U_2, say $u_{2,2}$, and *Role 3*: one of the required proxy signers in U_3, say $u_{3,2}$. All kinds of attacks will be described in the model later. Besides, A can access to a chained threshold proxy signing oracle. So the goal of the adversary A includes:

- A forgery *CTPS* on behalf of u_1 (*SHU*);
- A forgery *CTPS* by proxy signers in U_3, who are delegated by U_2 including $u_{2,2}$ (*SHU*), on behalf of u_1;
- A forgery *CTPS* by proxy signers including $u_{3,2}$ (*SHU*) in U_3 on behalf of u_1.

Definition 2 (Security of CTPS Scheme) *Let* ***CTPS*** = (*G*, *K*, (*TD*, *TP*), (*CTD*, *CTP*), *CTPS*, *CTPV*, *CTPID*) *be a chained threshold proxy signature scheme. Consider an experiment* $\boldsymbol{Exp}_{CTPS,A}(k)$ *related to* ***CTPS***, *adversary A and parameter k. In the extreme case, adversary A should represent all proxy signers if the SHU is the original signer; or the original signer and all other proxy signers except the SHU if the SHU is one of the proxy signers in level 2 or 3. First, system parameters* params *and secret/public key pairs are generated by running G and K. Empty array* $\boldsymbol{skp}_{3,2}$ *and empty sets* $\boldsymbol{pkg}_2$ *and* $\boldsymbol{pkg}_3$ *are created. The adversary A is of all the secret keys except SHU's, and it can make the following requests and queries.*

R1: u_1(*SHU*) designates $\{u_{2,j}\}_{n_2}$ and $\{u_{3,k}\}_{n_3}$ as his proxy signers. *A* requests to interact with u_1(*SHU*) running *TD*, and plays the role of $\{u_{2,j}\}_{n_2}$ running *TP*, and the roles of U_2 and $\{u_{3,k}\}_{n_3}$ running (*CTD*, *CTP*). And $\boldsymbol{pkg}_2$ is set to $\boldsymbol{pkg}_2 \cup pkg_2$.

R2: u_1 designates $\{u_{2,j}\}_{n_2}$ and $\{u_{3,k}\}_{n_3}$, where $u_{2,2}$ is the *SHU*, as his proxy signers in level 2 and 3 respectively. *A* requests to interact with $\{u_{2,j}\}_{n_2}$ running *TP*, and plays the role of u_1 running *TD* and the roles of all other proxy signers in level 2 except $u_{2,j}$. Then *A* requests again to interact with $u_{2,2}$ running *CTD*,, and plays the role of U_2 except $u_{2,2}$, and $\{u_{3,k}\}_{n_3}$ running (*CTD*, *CTP*). $\boldsymbol{pkg}_3$ is set to $\boldsymbol{pkg}_3 \cup pkg_3$.

R3: u_1 designates $\{u_{2,j}\}_{n_2}$ and $\{u_{3,k}\}_{n_3}$, where $u_{3,2}$ is the *SHU*, as his proxy signers in level 2 and 3 respectively. *A* requests to interact with $\{u_{3,k}\}_{n_3}$ running *CTP*, and plays the role of U_2 running *CTD*, and the roles of u_1 and $\{u_{2,j}\}_{n_2}$ running (*TD*, *TP*). The private output $\boldsymbol{skp}_{3,2}$ by $u_{3,2}$ (*SHU*) is stored in $\boldsymbol{skp}_{3,2}$. *A* does not have access to $\boldsymbol{skp}_{3,2}$.

Q1: Chained threshold proxy signature query by U_3, where $u_{3,2}$ is the *SHU*, on behalf of u_1. *A* can make a query (*M*, 32, *x*) to oracle O_{CTPS} ($skp_{3,k}$(k = 1,3, ⋯ ,t_3), $\boldsymbol{skp}_{3,2}[x]$,·,·,·,). If $\boldsymbol{skp}_{3,2}[x]$ has been defined, we say that this query is valid and the oracle returns $p\sigma_3 = O_{CTPS}$ ($skp_{3,k}$(k = 1,3, ⋯ ,t_3), $\boldsymbol{skp}_{3,2}[x]$,*M*,32,*x*,). Eventually *A* outputs a forgery (*M*, $p\sigma_3$, pk_1). The output of the experiment is 1, if

- *CTPID* ($p\sigma_3$) \ $pkg_3 \notin \boldsymbol{pkg}_2$, or
- *CTPID* ($p\sigma_3$) \ $pkg_2 \notin \boldsymbol{pkg}_3$, or
- No valid query (*M*, 32, *x*)to O_{CTPS} ($skp_{3,k}$(k = 1,3, ⋯ ,t_3), $\boldsymbol{skp}_{3,2}[x]$,·,·,·,).

Otherwise, the output is 0.

We define the advantage of adversary A as

$\boldsymbol{Adv}_{CTPS,A}(k) = \Pr[\boldsymbol{Exp}_{CTPS,A}(k) = 1]$.

We say that ***CTPS*** is a secure chained threshold proxy signature scheme if the function $\boldsymbol{Adv}_{CTPS,A}(k)$ is negligible for *A* of time complexity polynomial in the security parameter *k*.

SECURITY OF CTPS. The following theorem states

our result about the security of Chained Threshold Proxy Signature scheme. The proof of Theorem 1 is in Appendix A.

Theorem 1 *Let* ***CTPS*** = (*G, K,* (*TD, TP*), (*CTD, CTP*), *CTPS, CTPV, CTPID*) *be our proposed chained threshold proxy signature scheme in random oracle model. If the Schnorr signature scheme is secure, then* ***CTPS*** *scheme is secure in random oracle model.*

PROOF IDEA. The conclusion that a Chained Threshold Proxy Signature scheme is provably secure can be deduced with respect to the contradiction that if a forgery of a chained threshold proxy signature scheme by *A* successes in polynomial time, i.e. $\boldsymbol{Adv}_{CTPS,A}(k)$ is not negligible, then a well-known standard signature scheme, i.e. the Schnorr signature scheme, is broken.

4. Chained Threshold Proxy Signature with Supervision (CTPSwS) Scheme

Recall when Sam, the software development department, appoints Steven, the software maintenance department, as the supervising agent to supervise proxy signers such that only with permission of Steven, proxy signers can perform the signing capabilities. *CTPSwS* scheme in this section extended from *CTPS* fits into this kind of scenario by deploying Lui *et al.*'s [9] idea into our *CTPS* scheme.

Different from the *CTPS* model, there is a new protocol (TD_{SA}, TP_{SA}) run between original signer u_1 and his supervising agent SA_1. To differentiate the protocol run between u_1 and $\{u_{2,j}\}_{n_2}$ in *CTPSwS* scheme with that in *CTPS*, we define the former as (TD_u, TP_u). The signing and verification algorithm should also include SA_1's public key. The *CTPSwS* scheme model is as follows in Definition 3. Besides the notations described in Section 2, we have several new ones shown here.

(sk_{SA_1}, pk_{SA_1}) : SA_1's secret and public key pair

skp_{SA_1} : partial proxy secret key generated by SA_1

ps_{SA_1} : partial chained threshold proxy signature generated by SA_1

Definition 3 (CTPSwS Scheme) *Let* ***CTPSwS*** = {*G, K,* (TD_{SA},TP_{SA}), (TD_u,TP_u), (*CTD*,*CTP*), *CTPSwS, CTPVwS, CTPIDwS*} *be a chained threshold proxy signature with supervision scheme, where the constituent algorithms run in polynomial time. G and K are similar to those in CTPS except that K is also responsible to generate* SA_1*'s secret/public key pair* (sk_{SA_1}, pk_{SA_1}).

(TD_{SA}, TP_{SA}) is a *Threshold Designation-Proxy* protocol between the original signer u_1 and the supervising agent SA_1. Both TD_{SA} and TP_{SA} take as input the public keys pk_1, pkg_2 and pk_{SA_1}, respectively. TD_{SA} also takes as input the secret key sk_1 of u_1, and TP_{SA} takes as input the secret key sk_{SA_1} of SA_1. As the result of the interaction, the expected local output of TP_{SA} is skp_{SA_1}, the *partial* proxy secret key of SA_1. The undeniability of both u_1 and SA_1 is achieved by adding sk_{SA_1} in the protocol.

$$[TD_{SA}(pk_1, pkg_2, pk_{SA_1}, sk_1),$$

$$TP_{SA}(pk_1, pkg_2, pk_{SA_1}, sk_{SA_1})] \rightarrow skp_{SA_1}.$$

(TD_u, TP_u) is a *Threshold Designation-Proxy* protocol between u_1 and $\{u_{2,j}\}_{n_2}$. u_1 runs TD_u to send warrant ω_1 to $\{u_{2,j}\}_{n_2}$. TP_u run by each proxy signer $u_{2,j}$ takes as input the public keys pk_1, pkg_2 and the secret key $sk_{2,j}$ respectively. As the result of the interaction, the expected local output of TP_u is $skp_{2,j}$, the *partial* proxy secret share of $u_{2,j}$. Note the difference of this $skp_{2,j}$ with that in *CTPS* scheme. We call it partial here because all $\{skp_{2,j}\}t_2$ can not perform anything without another part skp_{SA_1}.

$$[TD_u, TP_u(pk_1, pkg_2, sk_{2,j})] \rightarrow skp_{2,j}$$

(*CTD, CTP*) is a Chained Threshold Designation-Proxy protocol between U_2 and $\{u_{3,k}\}_{n_3}$. It is similar to that is defined in Definition 1.

$$[CTD(pk_1, pkg_2, pk_{SA_1}, pkg_3, \{skp_{2,j}\}t_2),$$

$$CTP(pk_1, pkg_2, pk_{SA_1}, pkg_3, sk_{3,k})] \rightarrow skp_{3,k}$$

CTPSwS is the (possibly) randomized *Chained Threshold Proxy Signing with Supervision* algorithm. It is run by U_3 with agreement of SA_1, and takes as input the t_3 out of n_3 corresponding partial proxy secret shares $\{skp_{3,k}\}t_3$ and SA_1's partial proxy secret skp_{SA_1} and a message $M \in \{0,1\}^*$, and outputs a chained threshold proxy signature with supervision $p\sigma_3$.

$$CTPSwS[\{skp_{3,k}\}t_3, skp_{SA_1}, M] \rightarrow p\sigma_3$$

CTPVwS is the deterministic *Chained Threshold Proxy Verification with Supervision* algorithm as follows.

$$CTPVwS[M, p\sigma_3, pk_1, pkg_2, pk_{SA_1}, pkg_3] = 0/1$$

CTPIDwS is the *Chained Threshold Proxy IDentification with Supervision* algorithm. It takes as input a valid proxy signature $p\sigma_3$, and outputs identities, i.e., public keys.

$$CTPIDwS(p\sigma_3) = [pkg_2, pk_{SA_1}, pkg_3] / \perp$$

Based on Definition 3, we give a draft of a concrete *CTPSwS* scheme here. Different from *CTPS*, u_1 needs to run TD_{SA} to calculate skt_1 and sends to his supervising agent SA_1. Different from Lui *et al.*'s idea, SA_1 runs TP_{SA} to generate $skp_{SA_1} = sk_{SA_1}e_1 + skt_1 \mod q$ using his secret key sk_{SA_1}. Without knowing $skt_{1,j}$, each $u_{2,j}$ runs TP_u to generate $skp_{2,j} = sk_{2,j}\, e_1 \mod q$. (*CTD, CTP*) run between

U_2 and $\{u_{3,k}\}_{n_3}$ is the same as that in *CTPS*. At last when each $u_{3,k} \in U_3$ agrees to the data part, they must forward this email to SA_1, Steven, for him to check the contract part. U_2 can not generate a valid proxy signature on behalf of Sam until Steven agrees to the contract part and contributes his partial proxy signature $ps_{SA_1} = skp_{SA_1} c_3 + k_{SA_1} \mod q$ where $k_{SA_1} \in_R Z_q$ and $r_{SA_1} = g^{k_{SA_1}} \bmod q$. Since SA_1's secret key sk_{SA_1} is involved, security level of our scheme is enhanced by adding SA_1-protected property.

5. Security of CTPSwS

The security model of *CTPSwS* is similar to that of *CTPS* defined in Definition 2, except that *A* can be the fourth role, *Role 4*: the supervising agent of u_1, SA_1. In terms of this role, the goal of *A* also includes a forgery *CTPSwS* by U_2, U_3 and SA_1 (*SHU*) on behalf of u_1.

Definition 4 (Security of CTPSwS) *Let* ***CTPSwS*** = {*G, K,* (*TD*$_{SA}$, *TP*$_{SA}$), (*TD*$_u$, *TP*$_u$), (*CTD, CTP*), *CTPSwS, CTPVwS, CTPIDwS*} *be a chained threshold proxy signature scheme. Consider an experiment* $\boldsymbol{Exp}_{CTPSwS,A}(k)$ *related to* ***CTPSwS****, adversary A and parameter k. In the extreme case, adversary A should represent all proxy signers and* SA_1 *if the SHU is the original signer, or the original signer,* SA_1*, and all other proxy signers except the SHU if the SHU is one of the proxy signers in level* 2 *or* 3*, or the original signers and all proxy signers if the supervising agent* SA_1 *is the SHU. Empty arrays* $\boldsymbol{skp}_{3,2}$, $\boldsymbol{skp}_{SA_1}$ *and empty sets* $\boldsymbol{pkg}_2$, $\boldsymbol{pkg}_3$ *are created. The adversary A can make the following requests and queries:*

R1: u_1(*SHU*) designates $\{u_{2,j}\}_{n_2}$ and $\{u_{3,k}\}_{n_3}$ as his proxy signer groups in level 2 & 3, respectively, and specifies SA_1 as u_1's supervising agent. *A* requests to interact with u_1(*SHU*) running *TD*$_{SA}$ and *TP*$_{SA}$, and plays roles of all others running(*TD*$_{SA}$, *TP*$_{SA}$) and (*CTD, CTP*). $\boldsymbol{pkg}_{SA_1}$ is set to $\boldsymbol{pkg}_{SA_1} \cup pkg_{SA_1}$.

R2: u_1 designates $\{u_{2,j}\}_{n_2}$ and $\{u_{3,k}\}_{n_3}$ as his proxy signer groups in level 2 & 3, where $u_{2,2}$ is the *SHU* and specifies SA_1 as u_1's supervising agent. *A* requests to interact with $u_{2,2}$ (*SHU*) running *TP*$_u$ and *CTD*, and plays roles of all others running (*TD*$_{SA}$, *TP*$_{SA}$), *TD*$_u$ and *CTP*. $\boldsymbol{pkg}_2$ is set to $\boldsymbol{pkg}_2 \cup pkg_2$.

R3: u_1 designates $\{u_{2,j}\}_{n_2}$ and $\{u_{3,k}\}_{n_3}$ as his proxy signer groups in level 2 & 3, respectively, and specifies SA_1 (*SHU*) as u_1's supervising agent. *A* requests to interact with SA_1 (*SHU*) running *TP*$_{SA}$, and plays roles of all others running the remanent algorithms and protocols. $\boldsymbol{skp}_{SA_1}$ is set to $\boldsymbol{skp}_{SA_1} \cup skp_{SA_1}$.

R4: u_1 designates $\{u_{2,j}\}_{n_2}$ and $\{u_{3,k}\}_{n_3}$ as his proxy signer groups in level 2 & 3, where $u_{3,2}$ is the *SHU*, and specifies SA_1 as u_1's supervising agent. *A* requests to interact with $u_{3,2}$ (*SHU*) running *CTP*, and plays roles of all others running the remanent algorithms and protocols. $\boldsymbol{skp}_{3,2}$ is set to $\boldsymbol{skp}_{3,2} \cup skp_{3,2}$.

Q1: Chained threshold proxy signature with supervision query, where $u_{3,2}$ is the *SHU*, on behalf of u_1. *A* can make a query (*M*,32,*x*) to oracle $O_{CTPVwS}(skp_{3,k}(k=1,3,\cdots,t_3), \boldsymbol{skp}_{3,2}[x], skp_{SA_1}, \cdot,\cdot,\cdot)$. If $\boldsymbol{skp}_{3,2}[x]$ has been defined, we say this query is valid and the oracle returns $p\sigma_3 = CTPSwS(skp_{3,k}(k=1,3,\cdots,t_3), \boldsymbol{skp}_{3,2}[x], skp_{SA_1}, M, \cdot,\cdot)$.

Q2: Chained threshold proxy signature with supervision query, where SA_1 is the *SHU*, on behalf of u_1. *A* can make a query (M,SA_1,x) to oracle $O_{CTPVwS}(skp_{3,k}(k=1,2,\cdots,t_3), \boldsymbol{skp}_{SA_1}[x], \cdot,\cdot,\cdot)$. If $\boldsymbol{skp}_{SA_1}[x]$ has been defined, we say this query is valid and the oracle returns $p\sigma_3 = CTPSwS(skp_{3,k}(k=1,2,\cdots,t_3), \boldsymbol{skp}_{SA_1}[x], M, \cdot,\cdot)$.

Eventually, *A* outputs a forgery (M,$p\sigma_3$,pk_1). The output of the experiment is determined as follows:

- $CTPID_wS(p\sigma_3) \setminus \left(pkg_3 \cup pkg_{SA_1}\right) \notin \boldsymbol{pkg}_2$, or

- $CTPID_wS(p\sigma_3) \setminus \left(pkg_2 \cup pkg_{SA_1}\right) \notin \boldsymbol{pkg}_3$, or

- No valid query (*M*,32,*x*) to $O_{CTPVwS}(skp_{3,k}(k=1,3,\cdots,t_3), \boldsymbol{skp}_{3,2}[x], \cdot,\cdot,\cdot)$,

- No valid query (M,SA_1,x) to $O_{CTPVwS}(skp_{3,k}(k=1,2,\cdots,t_3), \boldsymbol{skp}_{SA_1}, \cdot,\cdot,\cdot)$.

Otherwise, the output is 0.

We define the advantage of adversary *A* as

$$\boldsymbol{Adv}_{CTPSwS,A}(k) = \Pr[\boldsymbol{Exp}_{CTPSwS,A}(k) = 1]$$

We say that ***CTPSwS*** is a secure chained threshold proxy signature with supervision scheme if the function $\boldsymbol{Adv}_{CTPSwS,A}(k)$ is negligible for *A* of time complexity polynomial in the security parameter *k*.

Security proof details follow the similar logic as the *CTPS* scheme, but are more tedious and skipped in this paper.

6. Conclusion and Discussion

This paper designs two provably secure schemes in random oracle model, Chained Threshold Proxy Signature (*CTPS*) scheme and Chained Threshold Proxy Signature with Supervision (*CTPSwS*) scheme, to solve the group-to-group delegation problem and delegation with supervision in delegation chain problem. They are very useful in email with signature system where a manager wants to delegate his signing right to his vice managers, who can further perform the delegation to their employees. In some cases, the delegatee's proxy signing rights can be supervised by manager's supervising agent. For future work, we hope to develop more flexible delegation

scheme that enables delegatees in different levels, say any one of vice managers and any two employees can cooperate to generate proxy signature. Also, more efficient schemes for these problems are always desirable.

REFERENCES

[1] Zoe L. Jiang, S. M. Yiu, L. C. K. Hui, Y. Dong, and S. H. Y. Wong, "Chained threshold proxy signature without and with Supervision," In 2008 International Conference on Computer Science and Software Engineering (CSSE'08), HongKong, China, pp. 837−840, December 2008.

[2] S. Kim, S. Park, and D. Won, "Proxy signatures, revisited," In 1st International Conference on Information and Communications Security (ICICS'97), LNCS 1334, Beijing, China, pp. 223−232, 1997.

[3] M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures for delegating signing operations," In 3rd ACM Conference on Computer and Communication Security, New Delhi, India, pp. 48−57, 1996.

[4] T. El Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," In IEEE Transactions on Information Theory, Vol. 31, No. 4, pp. 469−472, 1985.

[5] T. Okamoto, "Provably secure and practical identification schemes and corresponding signature schemes," In Advances in Cryptology (Crypto'92), LNCS 740, pp. 31−53, 1992.

[6] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," In Advances in Cryptology-Eurocrypt 1986 (EuroCrypt'86), LNCS 263, pp. 186−194, 1987.

[7] B. C. Neuman, "Proxy-based authorization and accounting for distributed systems," In Proceedings of 13th International Conference on Distributed Computing Systems, Pittsburgh, USA, pp. 283−291, May 1993.

[8] M. Cerecedo, T. Matsumoto, and H. Imal, "Efficient and secure multiparty generation of digital signatures based on discrete logarithms," In IEICE Transactions on Fundamentals of Electronics, Communications & Computer Science, Vol. E76-A, No. 4, pp. 532−545, 1993.

[9] R. W. C. Lui, L. C. K. Hui, and S. M. Yiu, "Delegation with supervision," In Information Sciences, Vol. 177, No. 19, pp. 4014−4030, 2007.

[10] A. Boldyreva, A. Palacio, and B. Warinschi, "Secure proxy signature schemes for delegation of signing rights," http://eprint.iacr.org/2003/096.

[11] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. "Proactive secret sharing or: How to cope with perpetual leakage," In Advances in Cryptology (Crypto'95), LNCS 963, Vol. 963, pp. 339−352, 1995.

[12] C. P. Schnorr, "Efficient signature generation by smartcards," In Journal of Cryptology, Vol. 4, No. 3, pp. 161−174, 1991.

APPENDIX

A Proof of Theorem 1

Suppose adversary *A* is a successful forger against CTPS scheme in polynomial time. Let adversaries *B*, *C*, and *D* wrap all communication channels from and to *A*, and have access to a standard signing oracle O_S, a chained threshold proxy signing oracle O_{CTPS}, and two random oracles functioning as hash functions *G* and *H* to answer *A*'s requests and queries. First, system parameters params and secret/public key pairs are generated by running *G* and *K*. Empty array $\boldsymbol{wskp}_{3,2}$ and empty sets $\boldsymbol{pkg}_2$ and $\boldsymbol{pkg}_3$ are created. The adversary *A* is of all the secret keys except *SHU*'s, and it can make the following requests and queries.

R1: *A* requests to interact with u_1(*SHU*) running *TD*, and plays the role of $\{u_{2,j}\}_{n_2}$ running *TP*. The request is interrupted by *B*. *B* creates an appropriate warrant ω_1 and makes query $(0 \| pk_1 \| pkg_2 \| \omega_1)$ to its signing oracle $O_S(sk_1,\cdot)$. Upon receiving an answer (skt_1,r_1), it forwards (ω_1,skt_1,r_1) to $\{u_{2,j}\}_{n_2}$. After a successful run, $\boldsymbol{pkg}_2$ is set to $\boldsymbol{pkg}_2 \cup pkg_2$.

R2: *A* requests to interact with $\{u_{2,j}\}_{n_2}$ running *TP*, where $u_{2,2}$ is the *SHU*, and plays the role of u_1 running *TD*. *C* creates an appropriate warrant ω_2 and makes query $(0 \| pkg_2 \| pkg_3 \| \omega_2)$ to its signing oracle $O_S(skp_{2,2},\cdot)$. Upon receiving an answer $(skt_{2,2}, r_2)$, it divides the answer into n_3 shares using random polynomial $F_{2,2}(x)$ and forwards $(\omega_2, \{skt_{2,2,k}\}_{n_3}, r_2)$ to $\{u_{3,k}\}_{n_3}$. $\boldsymbol{pkg}_3$ is set to $\boldsymbol{pkg}_3 \cup pkg_3$.

R3: *A* requests to interact with $\{u_{3,k}\}_{n_3}$ running *CTP*, where $u_{3,2}$ is the *SHU*, and plays the role of U_2 running *CTD*. When *A* outputs ω_2, $skt_{2,j,2}$, r_2, *D* verifies them. If all the verifications pass, *D* stores $(\omega_2, skt'_{2,2}, r_2)$ in the last unoccupied position of $\boldsymbol{wskp}_{3,2}$.

Q1: *A* can make a query (*M*,32,*x*) to oracle $O_{CTPS}(skp_{3,k}(k = 1,3,\cdots,t_3),\boldsymbol{skp}_{3,2}[x],\cdot,\cdot,\cdot)$. If $\boldsymbol{wskp}_{3,2}[x]$ is not defined, it returns $\perp$. Otherwise, *D* performs the following operations:

- Pick a random number $c_3 \in Z_q$.
- Pick a random number $ps_{3,2} \in Z_q$.
- Make a query $(0 \| pkg_2 \| pkg_3 \| \omega_2, r_2)$ to oracle *G* and let e_2 be the response.
- Compute commitment

$$r_{3,2} = g^{ps_{3,2}} (pk_{3,2}{}^{e_1} \cdot g^{skt_{2,j}})^{-c_3} \bmod p$$

- Set $H(1 \| M \| pkg_2 \| pkg_3 \| \omega_2 \| r_2, r_3) \leftarrow c_3$
- Return $(r_{3,2}, ps_{3,2})$ to *A*

Suppose after the experiment $\boldsymbol{Exp}_{CTPS,A}(k)$, *A* outputs a forgery in polynomial time of *k* such that at least one of the following events occurs:

E1. $CTPID(p\sigma_3) \setminus pkg_3 \notin \boldsymbol{pkg}_2$.

E2. $CTPID(p\sigma_3) \setminus pkg_2 \notin \boldsymbol{pkg}_3$.

E3. no valid query (*M*,32,*x*) to $O_{CTPS}(skp_{3,k}(k = 1,3,\cdots,t_3),\boldsymbol{skp}_{3,2}[x],\cdot,\cdot,\cdot)$.

Assume **E1** occurs. Since all the coming in and out channels are wrapped by the adversaries, the query $H(1 \| M \| pkg_2 \| pkg_3 \| \omega_2 \| r_2, r_3)$ made by *A* must be grabbed and responded by c_3. By using forking lemma, *B* can rewind *A* to this point and gives *A* another random $c'_3 \neq c_3$. With non-negligible probability, *A* produces a forgery with respect to the same query, such that

$$g^{p\sigma_3} = PKP^{c_3} \cdot r_3 \bmod p,$$
$$g^{p\sigma'_3} = PKP^{c'_3} \cdot r_3 \bmod p.$$

Hence,

$$g^{(p\sigma_3 - p\sigma'_3) \bmod q} = PKP^{(c_3 - c'_3) \bmod q} \bmod p,$$
$$PKP = g^{SKP} \bmod p,$$
$$SKP = e_2 \cdot skg_3 + e_2(e_1(skg_2 + sk_1) + k_1) + k_2 \bmod q.$$

B subtracts $e_2 \cdot skg_3 + e_2(e_1 \cdot skg_2) + k_2$ for *B* knows skg_2 and skg_3. Then it obtains $\sigma'_1 = sk_1 \cdot e_1 + k_1 \bmod q$, a successful Schnorr signature of u_1.

Assume **E2** occurs. The deduction is similar to the above. However since the *SHU* is $u_{2,2}$, the adversary *C* should subtract the corresponding parts, and obtain $\sigma'_{2,2} = sk_{2,2} \cdot e_1 + k_1 \bmod q$, a successful Schnorr signature of $u_{2,2}$.

Assume **E3** occurs. The deduction is similar to the above. However since the *SHU* is $u_{3,2}$, the adversary *D* should subtract the corresponding parts, and obtain $\sigma'_{3,2} = sk_{3,2} \cdot e_2 + k_2 \bmod q$, a successful Schnorr signature of $u_{3,2}$.

J. Software Engineering & Applications, 2009, 2: 276-282
doi:10.4236/jsea.2009.24035 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

A CORBA Replication Voting Mechanism for Maintaining the Replica Consistent

Guohua WU, Xiaojun LI, Qiuhua ZHENG, Zhen ZHANG

College of Computer Science, Hangzhou Electronic University, Hangzhou, China.
Email: lixjun007@gmail.com

Received April 23rd, 2009; revised June 29th, 2009; accepted July 8th, 2009

ABSTRACT

Nowadays, more and more applications are being developed through distributed object computing middleware, such as CORBA, their requirements for fault-tolerance, especially real time and critical system, become more and more critical. Despite almost ten years have passed since the earliest FT-CORBA standard was promulgated by Object Management Group (OMG), CORBA is still facing many challenges when it is used for distributed applications developing, as the standard is complex and lack of understanding. This paper focus on the consistency of the replicated object and the network partition problem, it incorporates a CORBA Replication Voting Mechanism (CRVM) to meet the challenge which makes a good performance.

Keywords*: *distributed computing, CRVM, fault-tolerance, replica consistent, networking partition

1. Introduction

The progress of the distributed application system and object-oriented programming technology has led to distributed object middleware, where objects are distributed across processors. Typical middleware applications contain sending client objects' requests and receiving reply from server objects, which implementing through message sent across the network. The Common Object Request Broker Architecture (CORBA) [1] is a standard for middleware and it is established by the Object Management Group.

CORBA has become one of the most popular middleware developing technologies, which is supported on almost every combination of hardware and operating system in existence. CORBA uses OMG Interface Definition Language (IDL) to define interface for objects, which is CORBA's fundamental abstraction mechanism for separating object interfaces from their implementations. A client only needs to know the IDL interface without the language-specific implementation of the server object. Under the CORBA's standard, clients and servers can communicate with the TCP/IP-base Internet Inter-ORB Protocol (IIOP), careless of the heterogeneity in their respective platform and operating system. Clients are allowed to invoke operation without caring about the server objects' physical location. These are attributed to the ORB, which make clients and servers transparent to each other's differences in platform, programming language and location.

More and more distributed system desire highly-performance, including dependability, efficiency, reliability etc, thus adding fault-tolerant standard to CORBA becomes more and more pressing. OMG adopted Fault-Tolerant CORBA standard in the late 1990s (1999.2 Version1.0, 2001.9 Version2.0). Although almost ten years have passed, due to the diverse set of fault-tolerance requirements and the large varieties of distributed applications requiring fault-tolerance, the current version of FT-CORBA standard compromises on the number of interfaces, policies, and features it provides. As a result, FT-CORBA vendors are free to provide proprietary extensions [2].

Replication is the most basic way we adopt to achieve fault tolerance, however it still faces many challenges. How to maintain replica consistency is a typical problem, as Fault tolerance will obviously fail if replicas are in different status during a servicing time. Another problem is network partition. There is no support for the consistent remerging of the replicas of CORBA objects following a network partition [3]. This paper focuses on these two problems as mentioned above and it introduces a voting mechanism CRVM to meet the challenges.

The remainder of this paper is organized as follows. Section 2 describes FT-CORBA standard and existing fault-tolerance strategies. Section 3 provides the related work about fault-tolerance in CORBA. Section 4 describes the dynamic voting algorithm in detail. Section 5, we put forward to vote mechanism (CRVM). Section 6 describes an implementation. Section 7 concludes the

paper.

2. The review of FT-CORBA Standard and Strategies

2.1 The FT-CORBA Standard

To achieve the purpose of fault-tolerant, FT-CORBA standard provides three mechanisms: Replication, Fault Detection and Fault Recover.

1) Replication: There is a Replication Manager (RM) responsible for replicating objects and distributing the replications across the system. Each replica has an individual Interoperable Object Reference (IOR) to identify them，RM itself also has many copies. The replication styles can be divided into the following two types [4]:

- Passive replication: only one of the server replicas is designated as the primary one, which responses for the client's requests. In the warm passive replication styles, the remaining passive replicas, called backups, are updated periodically with the primary replica so that one of them can be selected to replace when the primary one fails. In the cold passive replication styles, the remaining backup replicas are "cold", they would neither process the client's requests nor make update with the primary one. To allow for recovery, the state of the primary replica is periodically checked and stored in a log. Once the primary replica failed, a backup replica is selected and initialized from the log to replace as the new primary one.
- Active replication: all of the server replicas maintain the same state, each one responses to client's requests. There is no influence when any one of them failed, keeping the running of replicas without any problem.

2) Fault Detection: Fault Detection implement by FD which is in charge of detecting the possible failure in the system and generates related reports. FD is arranged into a hierarchical structure: object level, process level and host level due to different detected objects. FD would generate a fault report to the Fault Notifier (FN) when a failure is detected, then the FN transmit it to RM and the objects which have been registered to FN and are interested in it. There are two kinds of detecting styles, pull-based and push-based.

- pull-based: FD periodically send a message Is_alive () to the detected objects, which should echo "I_am_alive ()" in the limited time. The objects would be considered erring in case of the FD did not receive the response in a certain time, and then FD would send fault report to RM for handling. It is passive for detected objects to echo, they won't return "I_am_alive ()" unless receive FD's "Is_alive ()".
- push-based: In this scheme, it is also known as a "heartbeat monitor". The detected object would send the message "I_am_alive ()" forwardly at set intervals. It is same to the pull-based styles, the object would be considered erring if FD did not receive the response in a certain time. It is active for detected objects to send the message "I_am_alive ()" periodically.

3) Recovery: With FT-CORBA fault-tolerant mechanism, the objects' state would be logged automatically if the replica inherit and implement the Checkpoint table and Updateable interfaces, once receive the notice the object would be returned to the latest state.

2.2 The Strategies of Existing FT-CORBA System

Generally speaking, research on FT-CORBA and its applications can be divided into the three strategies [5], the integration strategy, the interception strategy, and the service strategy, which are described in detail as below:

- Integration strategy: In this strategy it has to modify the ORB core to provide the necessary fault tolerance support, thus, it is certain to violate the FT-CORBA standard.

Because the modification is added to the ORB, application's interfaces to the ORB remains unchanged, it implies that the replication of server objects can be made transparent to the client objects. Electra [6] and Orbix+Isis [7] are examples of the integration strategy.

- Interception strategy: In this strategy, an interceptor is added into the architecture to capture the system's call, and then to modify the request parameter to change the behavior of the application without the application or the ORB being aware of the interceptor's existence and operation, finally repackage the call and deliver it through multicast. However, the interceptor has to be ported to every operating system which is intended to run the CORBA application. The examples of this strategy are Enternal System [8], and AQuA framework [9], etc.
- Service strategy: This strategy enhances CORBA through adding a new service which just likes one of CORBA Common Object Service. In order to achieve fault tolerance, a set of interfaces are defined to provide the policies and mechanisms. In other words, fault tolerance can be provided as apart of the standard suite of CORBA ORBs. Since the CORBA service is a collection of CORBA object fully above the ORB, the ORB is certainly unnecessary to be modified. However, the application code may require modification for supporting the fault-tolerance service.

The comparison of different strategies outlined above is illustrated in Table 1 [3].

2.3 Replication

No matter adopting which kind of strategy to achieve the purpose of fault-tolerant, replication is the most basic way, which has been widely applied in distributed systems. The purpose of replication is to provide multiple, redundant, identical copies, or replicas, of an object so that the object can continue to provide useful services,

Table 1. Comparison of three strategies

Strategy	Advantages	Disadvantages	System
Intergration	Transparent to client applications (Without modify any co- de of client ap- plications)	Bad portable due to modify ORB and IDL	Orbix+Isis Electra
Interception	Without modify ORB and transparent to the applications	Interceptor needs to be ported to every operating system which intend to use it	Eternal
Service	Has a good portability to client applications	Sever applications are lack of transparent.(Program developer have to rewrite interface of the related objects)	DOORS AQuA

even though some of its replicas fail, or as the processors hosting some of its replicas fail [10].

This technology duplicates system resources and distributes them into hosts which locate in different places. Certainly, the status and behavior of the replicas must been maintaining the same, once a certain replica failed, the back-up replicas could take over and continue to provide services. This process is transparent to the client which is not aware of the server object weather has failed or not since the back-up replicas still keep on proving service to meet the client's demand.

Some advantages of replication:

- Enhanced system available: Replicative resource distribute in different hosts, the remaining replica could keep on providing service when one certain host failed. For example, suppose there are n hosts and the error probability of each host is p, obviously, the available of the resource could be expressed: 1-pn，it implies that the more replicas it would be the stronger available.
- Fault-tolerant: it is described above.

Although replication has many advantages, the challenges resulted are also critical .One important issue is how to ensure the consistency of the replicas. Besides, there is no good solution to resolve network partition. The FT-CORBA standard does not provide mechanism to handle faults due to network partitioning [2].

In this paper we introduce CRVM to resolve the problem mentioned above: replica consistency and network partition.

3. Related Works

Previous study on fault tolerance has made great help to FT-CORBA standard which is shown as follows.

Electra [6] was developed at the University of Zurich, as one of the earliest implementations in accordance with fault tolerance CORBA standard, which adopted integration strategy to achieve maintaining replica consistency. It contains a modified ORB core and the Hours toolkit [11] which is exploited to provide reliable totally ordered group communication mechanisms. In an Electura host, a CORBA client can request a replicated server object for operating with out caring about where is the server object or the number of them, even if the server object exists.

Orbix+Isis [11] was developed by IONA technologies which is similar to Electra, this product also modifies the internals of the ORB in order to adjust to Isis toolkit [12] which provides the reliable ordered multicast protocols. Besides, Orbix+Isis is the first commercial product which complies with the standard.

Distributed Object-Oriented Reliable System (DOORS) [2] which was developed at Lucent technologies, provided a service strategy to manage the object groups and replica objects. This product was absorbed in passive replication which employs libraries for the transparent checkpointing of applications and state recovery.

Eternal [8] which was developed at University of California, Santa Barbara, adopted interception strategy to support both active and passive replication styles. The mechanism implemented in different parts of the Eternal system together with its logging recovery mechanisms to ensure strong replica consistency without modifying either the application or the ORB.

4. Dynamic Voting Algorithm

Dynamic voting algorithm [13,14] proposed by two scholars Sushil Jajodia and Dvid Mutchler is used to control replication. The algorithm is designed for the fault-tolerant of replicated database, its major purpose is to maintain highly-consistency of the database and enhance system available. Actually, it has a good performance in fault-tolerant as well as in maintain consistency of data.

In our algorithm, we assume that all of the hosts and networks may go wrong, except for the Byzantium fault. In other words, the host would stop running without processing any order once the host failure occurred. Since the property of replicated database itself can tolerate host fault or process fault, another contribution of the algorithm is put forward FT strategy to resolve the network partition problem.

It adopts majority decision to determine which part of network contains more database, called majority partition, can continue running. In comparison, the remaining database should stop immediately to avoid two parts of database running simultaneously without communicating to update for consistency. The remaining database keep on running may lead to inconsistent, which is illustrated in Figure 1.

In simple terms, fault-tolerant of this algorithm is to duplicate data, distribute them to different hosts, and maintain data consistency through voting method. It means before accessing to a replica data, a voting step must be processed to insure the replica data is contained

in the majority partition.

Some parameters are necessary to meet the demand of the algorithm, which is needed when processing judgment.

- Version Number (VN): an integer, expresses the update times of the replica data;
- Site Cardinality (SC): an integer, expresses the number of replica data, i.e. number of hosts that replica is stored.
- Distinguished Site (DS): a replica data identifier, which is assigned as primary one.

Client send request to a single replica data without being aware of how does the replica data communicate with each other, it's transparent to client. Since every replica data can process the request from client, it also enhances the system's performance. The running diagram of dynamic voting algorithm is illustrated in Figure 2.

Algorithm processes and procedures are described as follows:

1) Set synchronous control: We assume that replica S receives request operation, it would send Concurrency_Request to remaining replicas at first. This part achieves synchronous control by using time stamp, thus, ensure there is only one replica is updated at a certain time. Bernstein [10] had ever advanced an algorithm to achieve mutual exclusion in distributed system which is described in detail as below:

Assume that process Pi wants to enter a critical section, a new time stamp (TS) would be generated and then (Pi, TS) would be dispatched to all processes. How does Pi reply to (Pi, TS), assent or delay depends on the three factors as showed below:

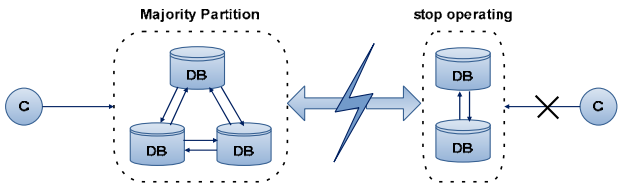


Figure 1. The network partition fault

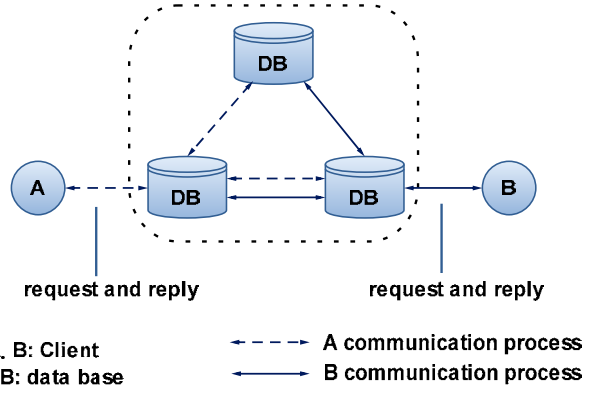


Figure 2. The communication among replicated DBs

1).If Pj is just alive in the critical section, it will chose to delay the response;

2) If Pj is not interested in the critical section, it will reply immediately;

3) If Pj also wants to enter the critical section, it will compare its TS with Pi's, obviously if TSj > TSi, send reply immediately, otherwise delay the reply.

The algorithm mentioned above has several features: Obtain mutual exclusion function, never get dead lock or starvation for whether process could enter the critical section obeys to the value of TS which ensures process served by FCFS.

2) Send Vote_Request: S execute Lock_Request after finishing Concurrency_Request, if lock successfully, S then send Vote_Request to remain replica data, supposed as Si, otherwise, S must execute termination protocol, and then do Lock_Request again, if still failed, the operation would be ended.

With executing termination protocol, S would send Decision_Request (contain DS and request identifier) to all remaining replica data. The replicas which receive request then reply whether update operation has finished, if finished, S then executes Realese_Lock.

3) Return status: Si will lock itself at first when receives Vote_Request, if locked successfully, returns VN SC DS to S, otherwise, executes termination protocol, later locks again, if still failed, returns null to S.

4) Determine majority partition: S collects all status replied by Si, and then executes Is_Distinguished to determine whether the network S is contained in the majority partition which is shown as follows:

m = number of replicas
M = max{VNi: $1 \leqslant i \leqslant m$} (an integer)
I = {i: VNi =M, $1 \leqslant i \leqslant m$ } (a replica set)
N = {SCi: $i \in I$} (an integer)
If $|I| > \frac{N}{2}$ or ($|I| = \frac{N}{2}$ and DS exists) then
Do_Update
else
end

If S doesn't belong to majority partition, the update request has to abort. Firstly, S executes Release_Lock and sends message "Abort" to Si, and then Si would execute Release_Lock once receives "Abort".

If S belongs to majority partition, determines whether the status of S is the latest first. If negative, S should ask Si for it through message"Catch_Up".

5) Response the request and deliver out latest status: S updates firstly and then executes Do_Updata: VNi+=1, SCi=m. After that, S delivers message "Commit" (contain Request identifier and status) to Si. Finally, S replies the result to client.

6) Si will update status once receives "Commit" and then executes Release_Lock request.

5. CRVM for FT-CORBA

Despite the dynamic voting algorithm is not designed especially for CPRBA object fault-tolerant, however, there are many similarities between replicated database and replicated object. Our research indicates that dynamic voting algorithm is proper to maintain consistency of replicated object, resolve network partition, crash failure etc. Thus in this paper we propose a new CORBA Replication voting mechanism (CRVM) for FT-CORBA. Figure 3 illustrates the CRVM working graph.

The same as original algorithm, we have to set some parameters to satisfy judgment demand, which is expressed as follows:

- Object State (OS): an integer used to signify update time of the object.
- Object Cardinality (OC): an integer used to indicate how many replica objects exist which maintain the same status.
- Distinguished Object (DO): an IOR related to an identified object which is used when CRVM wants to judge whether current object belongs to the majority partition.

The CRVM is expressed as follows:

```
    [Illustrate]
    cur_ser_obj: the current server object which link to
client and raise voting;
    alive_obj: the objects reply voting request from
cur_ser_obj;
    T: the objects' time stamp formed by IOR and system
time;
    R: client request;
    S: status of object.
    OM: object management;

    OS, OC and DO initiated by OM
status ∈ {null, active, voting, commit, abort}, initially null
  lock ∈ {true, false}, initially false
  LockedTimeStamp ∈ T, initially null
  TimeStamp ∈ T, initially null
  RequestList, a table store request and result
```

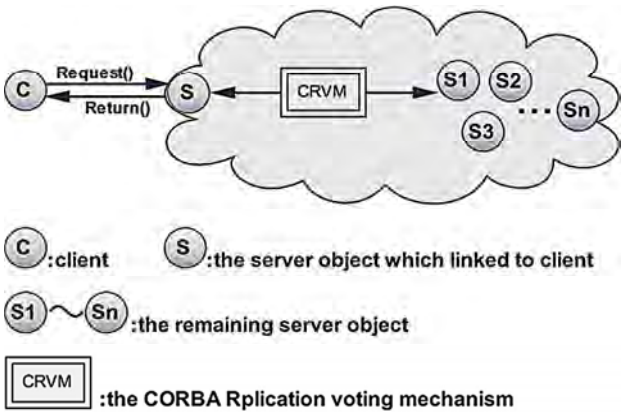


Figure 3. CRVM between replicas

```
[status transfer]
if status == active
send Concurrency_Request to all alive_obj
  status = voting
if status == voting
if (!Lock_Request)
    send Decision_Request to all alive_obj
    if (Lock_Request)
    send Vote_Request to all alive_obj
       if (majority partition)
          store request and result in Request Record
          status = commit
       else
          status = abort
if status = commit
  send Commit and object states to all alive_obj
status = idle
if status = abort
  send Abort to all alive_obj
    status = idle
```

```
[request processing]
receive ("Request",req,ts ), req ∈ R, ts ∈ T
if req not in RequestRecord
    status = active
else
    return request-record(ts)
receive("Concurrency_Request",ts), ts ∈ T
  if not conform to Berdtein's algorithm
    waiting
  else
    return
receive ("Decesion_Request",ts), ts ∈ T
  if ts found in LockRecord
    return false
  else
    return true
receive ("Vote_Request",ts), ts ∈ T
  if (!Lock_Request)
    send Decision_Request to all objects
  if (Lock_Request)
    LockedTimeStamp = ts
    store ts in LockRecord
    return OS, OC, DO
else
     return null
receive ("Commit",obj),obj ∈ S
lock = false
setup object state
receive ("Abort")
    lock = false
```

6. Implementation

FT-CORBA uses redundancy mechanism to achieve fault-tolerance, replication itself can tolerate process fault and host fault, our voting mechanism focus on maintaining consistency of replicated object and network partition fault.

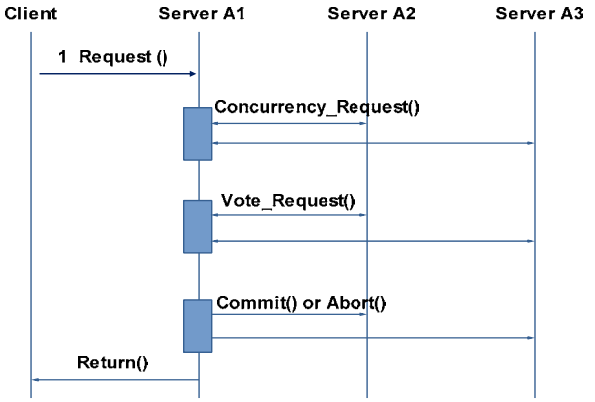


Figure 4. CRVM voting processing

The process of our voting mechanism is illustrated in Figure 4.

1) Firstly Client sends a request to server, OM then chooses a object（maybe a replicated object）to serve it, for instance we can make it through pinging the fastest object;

2) The object who receives the request would start voting mechanism, if the request has been severed then returns null, if not it sends Concurrency_Request to A2 A3 for synchronous control;

3) Send Voting_Request. A1 executes Lock_Request itself after finishing Concurrency_Request, case lock operation successfully, A1 then sends Vote_Request to A2 A3, otherwise, A1 must execute termination protocol, and then do Lock_Request again, if still failed, the operation would end this time;

With executing termination protocol, A1 would send Decision_Request (contain DS and request identifier)to A1 A2 who receive request then reply whether update operation has finished, if finished, A1 then execute Realese_Lock;

A2 A3 would lock itself through Lock_Request when receive Vote_Request, case lock successfully, return its VN SC DS to S, otherwise, execute termination protocol, later lock again, if still failed, return null to A1;

4) Determine majority partition: A1 collects all status replied by A2 A3, and then executes Is_Distinguished to determine whether the network A1 contained in the majority partition. According to the result, it decides to send Commit () or Abort ().

5) A1 returns the result.

The following is an example of CRVM process. We simulated the ATM depositing and withdrawing money operation to show how to ensure the consistency of replicated object and resolve network partition fault, IDL file is shown as follows:

```
interface atmOperate
{
  float inquiry ();                  // balance inquiries
  void deposit(in float num);        // depositing money
  void withdraw(int float num);      // withdrawing money
}
```

We define three operations: inquiry () deposite (in float num) and withdraw (in float num). The server-side object is responsible for deposits withdraws and inquiries business, in order to achieve fault-tolerance, server-side starts three replicated objects A1 A2 and A3, their status can be expressed by balance which we set it as S in following figures. For simplicity, we set an account with its balance $300.00 and had operated 5 times, the distinguished object is A1. So the initial status of A1 A2 and A3 is illustrated in Figure 5.

When a network partition fault occurred between A1 A2 and A3, after being requested by the operation deposit (500.00), A2 gets synchronous with A1 through CRVM while A3 remains constant which is illustrated in Figure 6.

When the fault is removed and a withdraw(300.00) request is sent to A1, since CRVM would find all normal objects to participate voting, A3 can re-join voting group and get synchronous with A1 which is illustrated in Figure7.

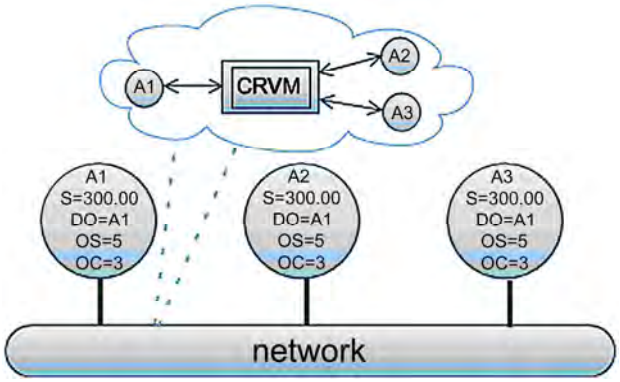


Figure 5. Initial status

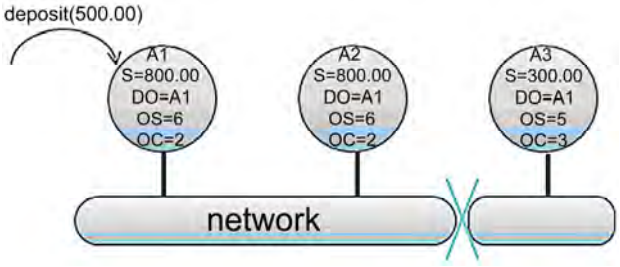


Figure 6. The status after operation deposit (500.00)

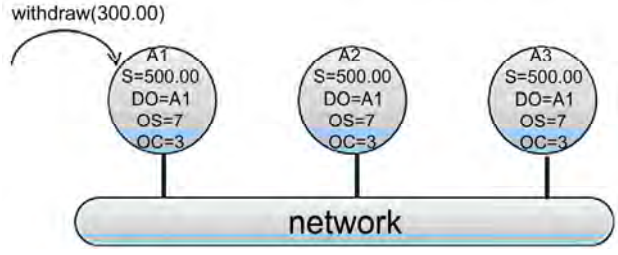


Figure 7. The status after operation withdraw (300.00)

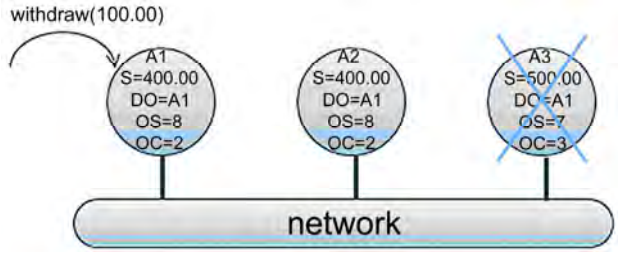


Figure 8. The status after operation withdraw (100.00)

Sending a withdraw (100.00) request to A1 while A3 gets a crash fault, the result is illustrated as Figure 8.

7. Conclusions

Since distributed applications development, fault-tolerance requirement has become more complex. The existed fault-tolerant technology is facing great challenges, for example, how to maintain the consistency of the object replication, how to control concurrent implementation and how to resolve network partition etc.

Despite FT-CORBA standard has been adopted by OMG, there is still a long way to improve it. The real-world applications is more complex and requires more stringent, current standard only provide a framework for fault-tolerance and can only be used for very simple applications. Thus FT-CORBA can not provide a ready solution for complicated applications now.

In this paper we design the CRVM to maintain the consistency of replicated object and resolve network partition problem, we will do further study and improvement on it next step.

8. Acknowledgements

This work is supported by National Fundamental Research Program of China under Grant NO.A1420080190.

REFERENCES

[1] Object Management Group, "Common Object Request Broker Architecture: Core Specification Version 3.0.2 " OMG Technical Committee Document date formal/02 −12−02.

[2] B. Natarajan, A. Gokhale, S. Yajnik, and D. C. Schmidt, "Doors: Towards high-performance fault-tolerant CORBA," Proc. Second Int'l Symp. Distributed Objects and Applications (DOA '00), pp. 39−48, February 2000.

[3] P. Felber and P. Narasimhan, "Experiences, strategies and challenges in building fault-tolerant CORBA Systems," IEEE Transactions on Computers, Vol. 53, No. 5, May 2004.

[4] S. Mullender, ed., Distributed Systems, Addison-Wesley, 1993.

[5] P. Narasimhan, "Transparent fault tolerance for CORBA," Ph.D.dissertation, University of California, Santa Barbara, December 1999.

[6] S. Maffeis, "Adding group communication and fault-tolerance to CORBA," Proc. Object-Oriented Technologies, June 1995.

[7] Kenneth Birman, Robbert, and van Renesse, "Reliable distributed computing with the Isis Toolkit," IEEE Computer Society Press, Los Alamitos, 1994.

[8] P. Narasimhan, L. E. Moser, and P. M. Melliar Smith, "Eternal-a component-based framework for transparent fault-tolerant CORBA," Software -Practice and Experience, pp. 771−788, 2002.

[9] M. Cukier, J. Ren, C. Sabnis, W. H. Sanders, D. E. Bakken, M. E. Berman, D. A. Karr, and R. E. Schantz, "AQUA: An adaptive architecture that provides Dependable Distributed Objects," IEEE Symposium on Reliable and Distributed Systems (SRDS), pp. 245−253, October 1998.

[10] P. Narasimhan, L. E. Moser, and P. M. Melliar Smith, "Strong replica consistency for fault-tolerant CORBA applications," Object-Oriented Real-Time Dependable Systems, 2001. Proceedings, Sixth International Workshop, pp. 10−17, January 2001.

[11] R. van Renesse, K. P. Birman, and S. Maffeis, "Horus: A flexible group communication system," Comm. ACM, Vol. 39, No. 4, pp. 76−83, April 1996.

[12] K. P. Birman and R. V. Renesse, "Reliable distributed computing with the Isis toolkit," IEEE Computer Society Press, March 1994.

[13] S. Jajodia and D. Mutchler, "Dynamic voting," ACM SIGMOD International Conference on Management of Data, San Francisco, pp. 227−238, 1987.

[14] S. Jajodia and D. Mutchler, "Dynamic voting algorithms for maintaining the consistency of a replicated database," ACM transactions on Database Systems, Vol. 15, No. 2, pp. 230−280, June 1990.

J. Software Engineering & Applications, 2009, 2: 283-287
doi:10.4236/jsea.2009.24036 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

Research on the Trust Model Based on the Groups' Internal Recommendation in E-Commerce Environment

Nan REN[1], Qin LI[2]

School of Economics and Management, Jiangsu University of Science & Technology, Jiangsu, China.
Email:[1]rennan_hb@sohu.com, [2]lei731@gmail.com

Received June 30th, 2009; revised August 2nd, 2009; accepted August 7th, 2009.

ABSTRACT

The trust plays an extremely important role in online shopping. In order to make online shopping trusty, this paper puts foreword a new trust model in e-commerce environment GIR-TM (Groups' Internal Recommendation Trust Model). First, it regarded the network as a combination of groups, and then did the internal recommendation based on these groups. The GIR-TM, in the process of recommendation, distinguished clearly between the trust degrees of recommendation node and the trust degrees of recommended node, and then calculated the integrated credibility value of the recommended node according to the weight of recommendation node in the group, the partial trust degree and the degree of recommendation when the recommendation node recommends the recommended node, and the overall credibility value of recommended node as well. Lastly through listing the experimental data and comparing with the HHRB-TM (History and Honest Recommendation Based Trust Model) on the same condition, it is verified that GIR-TM is feasible and effective.

Keywords*: *E-commerce, Groups, Internal Recommendation, the Credibility Value

1. Introduction

Trust is the basis of co-operation, and it plays an extremely important role in online shopping. At present, there are still some issues in peer-to-peer trust model, such as over-reliance on the recommendations of others, trust calculations' inaccuracy, difficulty of dealing with the united malicious attacks, dynamic strategy malicious nodes and so on [1]. In order to promote e-commerce to develop stably and quickly, researchers around the world have done some researches in the field of trust model: M. H. Hanif Durad *et al.* [2] discussed how to utilize the trust management to strengthen its security in the grid environment. G. Liang *et al.* [3] discussed how to use the trust management system such as reputation systems to solve the trust problem among the users. F. Almenarez *et al.* [4] discussed how to use the auto-negotiation technologies to solve the dynamic trust management in general environment. Jøsang A *et al.* [5] pointed out that in the P2P environment, the core problem of the trust mechanism based on the reputation were that: in the given application, what trust factors are the most appropriately used to infer the measurement of trust and reputatition? How to generate, acquire and aggregate the information about these trust factors? Whether the trust mechanisms can resist various attacks which are controlled by strategic individuals? Li Wen [6] put forward a History and Honest Recommendation Based Trust Model in Peer-to-Peer Networks, and improved the evaluation algorithm of trust. Chen Xiaoliang [7] classified the impact factors, built the calculation model of initial trust, and figured out that Consumers had distrust of web sites and online stores which is a bottleneck in e-commerce development.

To solve the core problem that consumers are lacking in trust of e-commerce currently, this paper establishes a trust model based on the groups' internal recommendation in E-commerce environment, in which the comprehensive credibility value of recommended node is calculated by the weight of recommendation node in the group, the partial trust degree when recommendation node recommends recommended node, and the degree of recommendation and the overall credibility value of recommended node. The calculated result can supply the basis for restraining malicious acts effectively (such as joint defamation, malicious exaggeration, providing false information, etc.).

2. GIR-TM

2.1 Group Mechanisms

2.1.1 Group's Structure

First of all, according to the credibility value of every node, the peer-to-peer network can be divided into three small collections [8]:

$$WholeN = \{Good, Bad, \phi\}$$

In which: *Good* is a collection of nodes with good credibility value gained through the good service provided to others.

Bad is a collection which is composed of malicious nodes;

ϕ is a collection of nodes whose credibility are unsure.

When a node p joins in the network, it is put into the collection ϕ, and its credibility value is 0. The nodes with good performance can increase their credibility value into a particular value until it is placed into the collection *Good*. On the contrary, if the node performs badly and its credibility value will be less than 0, then it will be moved to the collection *Bad*, meanwhile, the information about its bad performance will be notified in the whole network. As received the notice, the nodes will not trade with the notified node any more and then cut off the connection with it.

Then, the nodes in collection *Good* will be grouped. Some nodes in this collection have a certain credibility value, higher reliability and stability, which compose the group called Trusted Group (TG), and we assume that its scale is Q. When the credibility value of a node in collection *Good* reaches a certain degree, it can set up its own trust group or apply to join in the existing groups. Those nodes that have not joined in the TG are put into another group (*AG*).

The administrator of a TG is a node which creates the group initially. Administrators must maintain a connection with all the nodes inside of group, and we assume that each node in group maintains k as external connection. Administrators can choose which node to join in, while the node can also choose its trust group. In order to clarify the information of each trade and the credibility value of each node in trading, we suggest that it is necessary to establish a Node's Information (NI) for each node in the net to record its own series of activities, just as shown in Table 1.

After a node establishes its own TG, the node needs to notify its information to the nodes which do not belong to any TG and the administrators of other trust groups (TGs). After the administrator of another TG receives the notice, it will inform the message to the nodes in its own group.

According to the above strategy, the entire network is divided into n TGs, collection *Bad*, collection ϕ, as well as *AG*. As shown in Figure 1, we assume that all nodes in collection *Good* have entered the trust group, just as the two trust groups TG_1 and TG_2 in Figure 1, and A, B respectively represent the administrators of TG_1 and TG_2, and the number of their foreign connection is K (K = 1, 2 ... n).

2.1.2 Connection of Nodes

After the node's credibility value in collection ϕ reaches a certain trust degree R_ϕ (the credibility value of collection ϕ) through the good behavior, it will be moved into collection *Good* according to the principle "two-way choice", that is, the node can choose trust group, and the administrator selects a node, while the node can choose to stay in *AG*, join in or build a TG.

As shown in Figure 1, through transacting with other nodes, node C's credibility value satisfies $R_C \geq R_\phi$, then it can enter the collection *Good*. When the node C decides to join in the TG_1, it needs to send application information to the administrator A, the information includes its ID and the kinds of commodities S_c .After receiving the application, the administrator A must carry out the following steps:

Step1: First of all, calculate the number of the nodes in TG_1, if the number reaches Q, reject the node C's connection, otherwise continue to Step 2;

Step2: Judge that whether the node c is a malicious node or not, if it is, refuse its connection, otherwise continue to Step 3;

Step3: The administrator reviews the node C's credibility value, R_{TG_1} is the credibility value of TG_1, if $R_{TG_1} > R_c \geq R_\phi$, reject the node C's connection, otherwise continue to Step4;

Step4: The node p belongs to TG_1, if $\forall$ p$\in$ TG_1, the kinds of commodities of p satisfies $S_p \neq S_c$, the node C is allowed to joining in, continue to Step5;

Table 1. The Node's Information (NI)

Group ID	Group Administrator	Value of credibility	Transaction node				
			Node's ID	Number of successful	Number of failure	Credibility Value	Integrated credibility value
			.....				
			.....				

Step5: The administrator allows the node C to join in TG_1, and establishes a connection with the node C;

Step6: C creates connection with all the nodes in this group and its own NI table.

After C receives the refused news, it can choose other TG, and repeat the above steps. If it is rejected by all administrators, it can set up its own group or stay in *AG*.

2.1.3 Departure of Nodes

Nodes' departure has two ways: one is active departure, and the other is passive departure. Active departure can withdraw from the peer-to-peer actively when the node completes the transactions. If the node is also the administrator, before it leaves, it will choose the node with the highest credibility value in the group as administrator, and copy the information of the group to it. Passive departure happens when a node's credibility value is less than the credibility value of the group, and the administrator ejects it out of the trust group, and puts it into collection *Bad*.

2.2 Internal Recommendation Mechanisms

2.2.1 Recommendation Ideas

As same as the real life, and on the basis of the Trust Group, the basic framework of GIR-TM is established, as shown in Figure 2,

In the Figure 2, the closed circle equals to a TG, in which each node (A, B, and C…) has its own transaction nodes. In order to clarify the information of each transaction and the credibility value of each transaction node, we proposed to establish a table called Node's Information (table NI) for every node. The table NI includes the ID, administrators of the group where the node stays, and the credibility value of node, as well as the transaction information such as the ID of other transactions nodes, the number of successful transactions and one of failed transactions, the credibility value of other nodes and the integrated credibility value (the calculation of the integrated credibility value is referred to the following section) after transacting. Assuming that A has transacted with the node E and with the completion of each

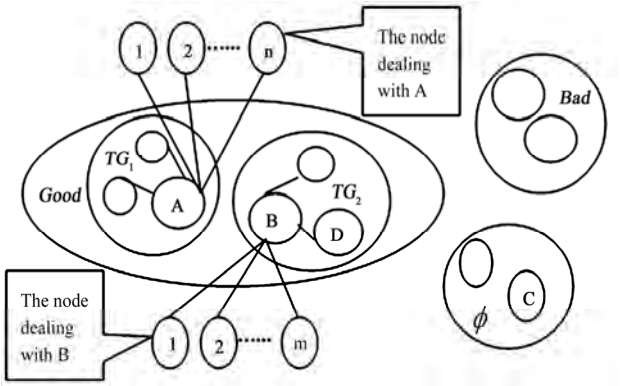


Figure 1. Network schematic of trust group-based

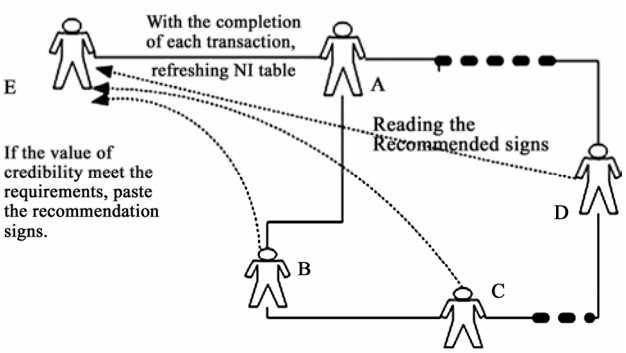


Figure 2. Basic framework of groups' internal recommendation trust model

transaction, the node A will refresh its own NI table (this NI table is to be shared, the shared region is the integrated credibility value with a recommendation tip sign), and then through comparing to judge whether the integrated credibility value of E meets the requirements or not, that is, if the integrated credibility value of E is greater than the overall credibility value of the group which includes node A, it will be signed with the recommended tip, which will be shared by the other nodes in this group, otherwise, giving up their recommendation.

2.2.2 Calculation of the Integrated Credibility Value

In order to calculate the Integrated Credibility Value of GIR-TM in Figure 2, we firstly introduced the following 3 definitions [9] about the partial trust, the degrees of recommendation, and the integrated credibility value:

Definition2.1 partial trust: $L_{u\to v}$ represents the partial view of the node U on node V, which directly comes from the historical transaction experience between them, given that

$$L_{u\to v} = \frac{S_{UV}}{Nuv} \tag{1}$$

In which, S_{uv} represents the number of successful transaction with node V in view of U;

N_{uv} represents the number of total transactions between U and V within the last interval time Δt (Δt is illustrated that the trust model pay more attention to the time limit of the nodes' behavior). If N_{uv} = 0, then $L_{u\to v}$ = 0.

Definition 2.2 the degrees of Recommendation: the degrees of recommendation how node X recommends the node V is calculated as follows:

$$R_{x\to v} = \frac{S_{xv} - F_{xv}}{S_{xv} + F_{xv}} \tag{2}$$

In which, F_{uv} represents that in node U's view, the number of failure transaction with node V. If $S_{xv} + F_{xv}$ =0, then set $R_{x\to v}$ =0. Or if $S_{xv} - F_{xv}$ <0, then set $R_{x\to v}$ =0. As shown in definition 2.2, if the nodes perform badly in the trading, and gain poor assessment from others, then his

credibility value will not be increased, but drastically reduced. So to some extent, this way can restrain the malicious acts of malicious nodes.

Definition 2.3 the integrated credibility degree: set $G_{u\to v}$ representing the projection of the trust level which node U trusts node V (U≠V) in the trust scope λ:

$$G_{u\to v} = [\alpha\ L_{u\to v} + (1-\alpha) T_v] \times \lambda \quad (3)$$

Here T_v is the overall credibility degree of the node V, T_v <1. In which, α is a constant and 0 <α <1, and λ is the trust scope, λ > 1.

Based on the quantitative description of the trust mentioned above, Document [6] put forward a HHRB-TM (History and Honest Recommendation Based Trust Model) in Peer-to-Peer Networks, and the corresponding integrated trust credibility value is calculated as the Formula (4):

$$G_{u\to v} = \omega_u\ L_{u\to v} + (1-\omega_u) \times (\sum_{i=1}^{N} R_{x\to v} \times Cr_x) \quad (4)$$

In which, Cr_x is the credibility value of node X, ω_u [10] is a weight factor about node U referring to its own direct history transaction experience. ω_u is dynamic, and changes with the time or the number of transactions.

But the Formula (4) does not consider the weight of recommendation node, and the role of the node with high credibility value does not play completely. Aiming at such problem, this paper put forward the definition of comprehensive weights, and the calculation formula is:

Definition 2.4 comprehensive weights: set Gt_x as comprehensive weights of a trust group, and the calculation formula is:

$$Gt_x = \frac{Cr_x}{\sum_{i=1}^{m} Cr_i} \quad (5)$$

Here Cr_x is the credibility value of node X, $\sum_{i=1}^{m} Cr_i$ is the sum of the credibility value of all the nodes in the group in which node X is included.

According to the principle of the higher credibility value, the higher credibility, the more accurate recommendation information, and then the bigger contribution rate, the weight of recommendation node is added into the calculation of the credibility value of recommended node, just as:

$$Ct_v = Gt_x \times R_{x\to v} \times Cr_x \quad (6)$$

In which, Gt_x is comprehensive weight of the trust group in which the recommendation node X is included;

$R_{x\to v}$ is the recommending evaluation that recommendation Node X puts foreword on the recommended node V;

Cr_x is the overall credibility value of recommended node.

Ct_v is gained by the feedback information about the Node V's recommendation, $Ct_v \le 1$;

Combining the partial trust with recommendation trust through (7):

$$G_{u\to v}' = \omega_u\ L_{u\to v} + (1-\omega_u)\, Ct_v \quad (7)$$

In which, $G_{u\to v}$'is the integrated credibility value to represent how the node U recommends node V;

$L_{u\to v}$ represents accumulated direct history transaction experience when the node U transacts join with node V;

So, the integrated credibility value is as follows:

$$G_{u\to v}' = \omega_u\ L_{u\to v} + (1-\omega_u)\, Gt_x \times R_{x\to v} \times Cr_x \quad (8)$$

3. Model Validations

We compared HHRB trust model with the trust model GIR in this article in the same experiment situation. In order to verify the model, enumerating 20 groups of experimental data, and assuming S_{uv} =380, N_{uv} =430, ω_u =0.9.

These 20 groups of experimental data include the number of successful transactions S_{xv}, the number of failure transactions F_{xv} and the credibility value of each node Cr_x. Each group of experimental data is listed randomly, because the GIR-TM is based on the TG. We could think the nodes in the group are reliable, and the failure rata is smaller, and then we could abide the principle that the listed data of the failure transactions number is always less than the successful transactions number, in addition, the failure transactions number needs to be much less. And the Cr_x of each node needs to be more than 0.5 (because every node is in the TG and should have a higher credibility value, and we assume that the threshold value of each TG's credibility value is 0.5). As the real life, the reputable people will form a group, and they are all reliable to have much more possibility to transact each other successfully. In generally, these experimental data are realistic and are shown in Table 2:

The experimental results in Figure 3 are calculated according to Table 2. As shown in Figure 3 (GN is group number), although the integrated credibility value of the HHRB model fluctuates sometimes, the integrated credibility values obtained by the HHRB model and the GIR model are more or less the same, just between 0.8825 and 0.8838. So it can be concluded that the GIR-TM is verified to be feasible and effective.

Table 2. Experimental data

	S_{xv}	F_{xv}	Cr_x		S_{xv}	F_{xv}	Cr_x		S_{xv}	F_{xv}	Cr_x		S_{xv}	F_{xv}	Cr_x
1	250	3	0.6	6	79	1	0.91	11	31	0	0.65	16	283	2	0.95
2	60	1	0.7	7	301	3	0.68	12	173	0	0.92	17	291	3	0.98
3	80	1	0.57	8	123	2	0.73	13	92	1	0.835	18	68	1	0.74
4	99	0	0.73	9	161	1	0.88	14	182	2	0.72	19	136	0	0.9
5	59	1	0.82	10	83	1	0.79	15	259	2	0.935	20	197	1	0.54

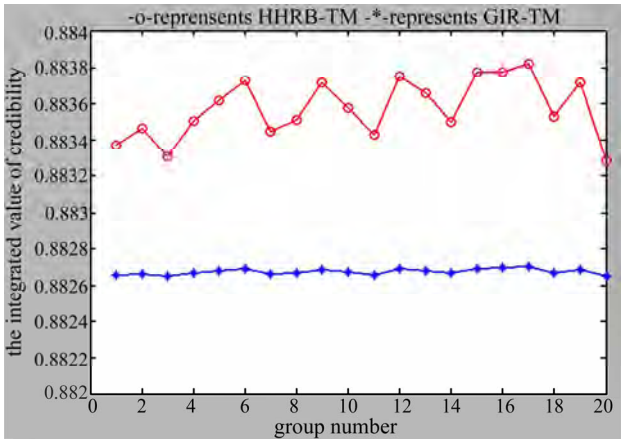


Figure 3. Comparing the integrated credibility value between $G_{u\to v}$ and $G_{u\to v}$

4. Summary and Prospect

The paper puts forward a Trust Model Based on the Groups' Internal Recommendation by analyzing the current issue of trust in electronic commerce. The model is composed of Trust Group and internal recommendation mechanism. Generally speaking, the achievements are as follows:

1) Put forward the GIR model based on the TG;

2) Put forward the algorithm of the integrated credibility value of the GIR model by improving the algorithm of the integrated credibility value of the HHRB;

3) Verify the effectiveness of the GIR model by comparing it with HHRB model.

Theoretically, the model can provide a good trading environment for customers, and reduce the occurrence of malicious actions. However, besides effectiveness verification of the GIR model, the model needs to be verified in the following aspects:

1) Verify the fairness of the transaction and the accuracy of the algorithm described in the model;

2) Further improve the model according to tested results;

3) Based on this paper, study on the rewarding and punishment mechanism to reward the reputable people and punish the malicious node.

5 Acknowledgments

This research was supported by Qing Lan Project of Jiangsu Province of China.

REFERENCES

[1] X. Li and L. Liu, "A reputation-based trust model for peer-to-peer e-commerce communities[C]," Proc of IEEE Conference on E-Commerce, ACM Press, California, pp. 275–284, 2003.

[2] M. H. Hanif Durad and C. Yuazlda, "A vision for the trust managed grid [A], Proceedings of the sixth IEEE International Symposium on Cluster Computing and the Grid [C], IEEE Computer Society Washington, DC, USA, 2006.

[3] G. Liang, L. Junzhou, and X. Yaobin, "Developing and managing trust in peer-to-peer systems [A]," proceedings of the 17th International Conference on Database and Expert Systems Applications[C], IEEE Computer Society Washington，DC，USA，2006.

[4] F. Almenarez, A. Marin, D. Diaz，and J. Sanchez, "Developing a model for trust management in pervasive devices [J]," 2006.

[5] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision [J]," Decision Support Systems, Vol. 43, No. 2, pp. 618−644, 2007.

[6] W. Li, "A history and honest recommendation based trust model in peer-to-peer Networks [D]," Hunan University, pp. 28–29, September 2007.

[7] X. L. Chen, "Research on the modeling and application of initial trust for e-commerce[D]," Huazhong University of Science and Technology, 2008.

[8] A. Gummadi and J. P. Yoon. "Modeling group trust for peer-to-peer access control," Database and Expert Systems Applications, Proceedings 15th International Workshop, pp. 971–978, 2004.

[9] W. Dou, H. M. Wang, and Y. Jia, "A recommendation-based peer-to-peer trust model [J]," Journal of software, Vol. 15, No. 4, pp. 571–583, 2004.

[10] G. Zacharia and P. Maes, "Trust management through reputation mechanisms [J]," Applied Artificial Intelligence, Vol. 14, No. 9, 2000.

J. Software Engineering & Applications, 2009, 2: 288-294
doi:10.4236/jsea.2009.24037 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

Adaptive Fuzzy Sliding Controller with Dynamic Compensation for Multi-Axis Machining

Hu LIN [1], Rongli GAI[2]

[1]Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang, China; [2]School of Computer Science and Technology, University of Science and Technology of China, Hefei, China.
Email: {linhu,gairli}@sict.ac.cn

Received June 31st, 2009; revised August 30th, 2009; accepted September 10th, 2009.

ABSTRACT

The precision of multi-axis machining is deeply influenced by the tracking error of multi-axis control system. Since the multi-axis machine tools have nonlinear and time-varying behaviors, it is difficult to establish an accurate dynamic model for multi-axis control system design. In this paper, a novel adaptive fuzzy sliding model controller with dynamic compensation is proposed to reduce tracking error and to improve precision of multi-axis machining. The major advantage of this approach is to achieve a high following speed without overshooting while maintaining a continuous CNC machine tool process. The adaptive fuzzy tuning rules are derived from a Lyapunov function to guarantee stability of the control system. The experimental results on GJ-110 show that the proposed control scheme effectively minimizes tracking errors of the CNC system with control performance surpassing that of a traditional PID controller.

***Keywords*:** *Fuzzy Sliding Control, Adaptive, Compensation Control, Tracking Error*

1. Introduction

While basic machine tool errors form one of the major sources of inaccuracy in multi-axis machining, achieving high precision in actual machine tool performance also critically depends upon dynamic performance of the individual axis controllers [1]. Axis controllers in processing of CNC machining need to synchronize multi-axis motions to generate the required machined surface [2]. In general, each particular machine tool axis has its own position and velocity, being driven separately along the desired tool path generated by the interpolator of the CNC system. When multi-axis machine tools are machining linear segments in "step" mode, the impact of tracking error on machining accuracy is not a serious problem. Otherwise, it would be difficult to eliminate tracking errors while tracking the axial position command, which then contributes to contour error formation. This situation would be especially problematic on the conditions of continuous processing of the multi-axis machining. As one example, Figure 1 provides an illustration indicating the influence of tracking error in a 2-axis system while machining in the "auto" mode. The interpolator is responsible for generating the ideal position commands for coordinated movement of the axes, which are indicated by positions B_i and B_{i+1}. Guided by the interpolator, the axis controller manages the task of controlling the movement of a particular axis. The actual machining points of A_i and A_{i+1} then correspond to interpolation points B_i and B_{i+1} due to the tracking error. Accordingly, the tracking error is about $|B_i\text{-}A_i|$ and $|B_{i+1}\text{-}A_{i+1}|$, thus the machining error is $|BD|$, which is called contour error, and is the result of the tracking error of each axis.

Over the past several decades, many advanced controllers have been designed which attempt to rectify these tracking errors. Some examples of these include the self-tuning fuzzy PID type controller [3−4], fuzzy sliding controller [5], adaptive sliding controller with self-tuning fuzzy compensation [6−10], adaptive neural fuzzy control [11], fuzzy controller and turning algorithm [12], and self-tuning fuzzy logic controller [13].

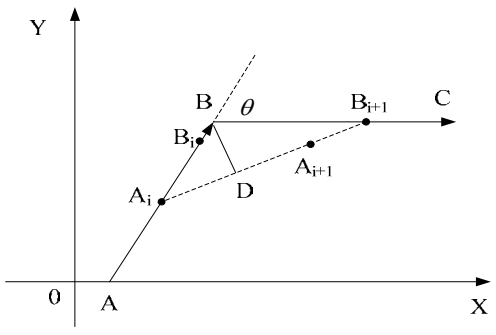


Figure 1. The influence of tracking error

Overshooting and oscillation of the multi-axis controller are special concerned in the article. Since it is difficult to establish an accurate dynamic model of the multi-axis machine tools, here, a novel adaptive fuzzy sliding model controller with dynamic compensation is proposed. The advantages of this model include its capacity to reduce tracking error, thus contour error and improve multi-axis machining precision.

2. Adaptive Fuzzy Sliding Controller with Dynamic Compensation

Adaptive fuzzy sliding controller with dynamic compensation is the core of the multi-axis CNC system. The basic function of this system is to generate the position of servo axes of multi-axis machine tools. Digital encoder feedbacks servo to monitor the position of the servo motor to the CNC system referred to as the actual axis location of the machine tool. Interpolation of the CNC system generates the ideal trajectory of each axis, which is called the ideal location of axis of the machine tools. While avoiding overshooting and oscillation in CNC machining, the main role of the adaptive fuzzy sliding controller with dynamic compensation is to minimize the tracking error, as computed by the actual versus ideal locations (Figure 2). The adaptive fuzzy sliding controller with dynamic compensation is a steady-state control system that includes two components - the dynamic compensation and adaptive fuzzy sliding control.

2.1 Dynamic Compensation

Dynamic compensation is calculated from an array of parameters including computing of tracking error, deadzone of tracking error, parameters of dynamic compensation and compensation control. The purpose of these determinations is to minimize tracking error of the CNC machining tools (Figure 3).

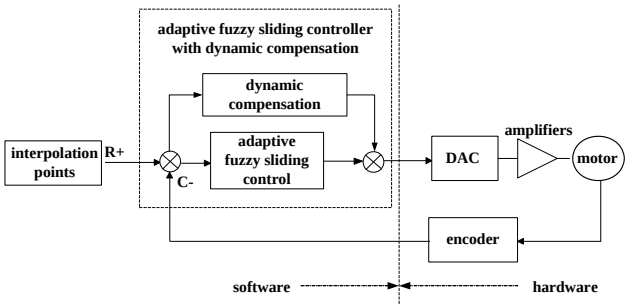


Figure 2. The servo axis control system using adaptive fuzzy sliding controller with dynamic compensation

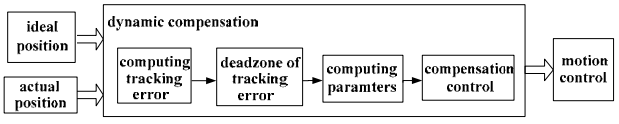


Figure 3. The frame of dynamic compensation

2.1.1 Computing Tracking Error

The method used for computing tracking error consists of calculating tracking error in each interpolation cycle based on the actual versus ideal locations of the CNC machining tool, as expressed in the following formula:

$$e_i = p_i - c_i \tag{1}$$

where $e_i(e_{i_X}, e_{i_Y}, e_{i_Z}, e_{i_A}, e_{i_B}, e_{i_C})$ is the tracking error in the ith interpolation cycle, $p_i(x_i, y_i, z_i, a_i, b_i, c_i)$ is the position of the ideal location coordinate of the ith interpolation cycle, $c_i(x_i, y_i, z_i, a_i, b_i, c_i)$ is the position of the actual location coordinate of the ith interpolation cycle, as indicated by the digital encoder.

2.1.2 Deadzone of Tracking Error

The deadzone of tracking error parameter is used to define a range, in which dynamic compensation is not used. The calculation of this parameter is based on the assumption that the maximal distance between the actual locations of multi-axis and the corresponding interpolation positions of multi-axis is within an interpolation step when the program is running at the programming velocity. That is:

$$e_i \le v \times \Delta t \tag{2}$$

where e_i is defined as presented in Equation (1), Δt is the interpolation cycle time of the multi-axis CNC system, v is the programming velocity of the particular workpiece program. It has been proven by Rogier that the trajectory planner makes the maximal interpolation error [14], expressed by $e_{\max}$, when it passes the adjacent program with programming velocity. Since the maximal interpolation error is accepted by the operator of CNC system, it can be applied to bound the dynamic compensation of tracking error, and be referred to as the deadzone of the tracking error. If, and only if, the tracking error is greater than $e_{\max}$, the dynamic compensation is active. Practical experience shows that the pursuit of reducing the tracking error to zero easily leads to vibration and overshooting in multi-axis machining, so the introduction of a deadzone of tracking error is conducive to the stability of multi-axis controller.

2.1.3 Parameters of Dynamic Compensation

The overshooting of multi-axis machine tools will result in oversize cutting. Since this is unacceptable for high precision machining, dynamic compensation needs to avoid overshooting and vibration of the multi-axis CNC system. Parameters of dynamic compensation include the following principles:

1) The parameters of dynamic compensation are zero when the velocity of trajectory planning is zero.

2) The parameters of dynamic compensation are zero when the tracking error achieves the constraint of the deadzone, defined by chapter 2.1.2.

3) The parameters of dynamic compensation can in-

crease in a step-wise manner when the multi-axis CNC system is running in an acceleration phase.

4) The parameters of dynamic compensation require a step-wise reduction when the multi-axis CNC system is running in a deceleration phase.

5) The parameters of dynamic compensation achieve a peak value when the multi-axis CNC system is running in the programming velocity.

Based on the principles of dynamic compensation, the following function can be selected as the parameter of dynamic compensation:

$$comp_i = \frac{v_i}{v} \times e_i \qquad le_i > e_{\max} \tag{3}$$

Where v_i is the velocity in the ith trajectory cycle, le_i is the length of the tracking error of multi-axis in the trajectory cycle, defined by Equation (4), $comp_i(comp_{i_X}, comp_{i_Y}, comp_{i_Z}, comp_{i_A}, comp_{i_B}, comp_{i_C})$ if the parameter of dynamic compensation is in the trajectory cycle.

$$le_i = \sqrt{e_{i_X}{}^2 + e_{i_Y}{}^2 + e_{i_Z}{}^2 + e_{i_A}{}^2 + e_{i_B}{}^2 + e_{i_C}{}^2} \tag{4}$$

2.1.4 Compensation Control

The output of the adaptive fuzzy sliding controller with dynamic compensation for multi-axis machining proposed in this paper can be expressed by the following equation:

$$u_{col} = u_c + u \tag{5}$$

Where u_{col} is the total output of the position controller of the multi-axis CNC system, $u_c$is the output of the dynamic compensation controller and u is the output of the adaptive fuzzy sliding controller. We can define the rules for compensation control as follows:

$$\begin{cases} u_c = k_p \times comp_i & le_i > e_{\max} \\ u_c = 0 & 0 \le le_i \le e_{\max} \end{cases} \tag{6}$$

Where k_p is the proportional gain of the CNC system, while the meanings of other variables are presented as Equation (3), (4) and (5).

The dynamic compensation controller adds a compensation variable to each axis according to the vector of the multi-axis tracking error. It can produce a rapid reduction in the tracking error while simultaneously reducing the frequency of adjusting parameters of the adaptive fuzzy sliding controller. In addition, the dynamic compensation controller can improve the stability of the numerical multi-axis control system.

2.2 Adaptive Fuzzy Sliding

There exist a number of nonlinearities and uncertainties in the multi-axis control system. These result from structural or unstructured uncertainties, such as backlash, saturation and friction. It is very difficult to establish the boundary for these nonlinearities and uncertainties in the CNC system, particularly in a multi-axis system. Dynamic compensation parameters are changed according to the trajectory velocity and therefore can contribute to an increase in the uncertainty of the multi-axis system. As an approach to rectify these problems, an adaptive fuzzy sliding control structure is proposed. This structure is referred to as adaptive fuzzy sliding controller with dynamic compensation since it incorporated the dynamic compensation algorithm discussed above.

Dynamic systems with multiple kinds of nonlinearities and uncertainties, such as multi-axis machine systems, can be expressed as the following Formula [15–17]:

$$x^{(n)}(t) = \hat{f}(X,t) + \Delta f(X,t) + u \tag{7}$$

Where $X = (x, \dot{x}, ..., x^{(n-1)})^T$ is the state of the system and $u \in R^n$ and $x \in R^n$ are the control inputs and outputs of the system, respectively. The nonlinear model system consists of the reference model $\hat{f}(X,t)$ and multiple nonlinearities and uncertainties $\Delta f(X,t)$, which include the dynamic compensation and inherent nonlinear characteristics of the multi-axis CNC system. A hypothesis can be proposed based on:

$$\left|\Delta f(x,t)\right| \le F(x,t) \tag{8}$$

The time-varying sliding surface is defined as:

$$s(x,t) = \left(\frac{d}{dt} + \lambda\right)^{n-1} e = 0. \tag{9}$$

It is referred to as the sliding switching line in 2D space, where λ is a positive constant and $e = X - X_d = (e, \dot{e}, ..., e^{(n-1)})^T$ is the tracking error vector.

In general, when considering second-order system as an example, the ideal fuzzy sliding control law is:

$$u^* = -\hat{f}(X,t) + \ddot{x}_d - \lambda\dot{e} - k(x,\dot{x})sat(s,\varphi) \tag{10}$$

Where $k(x,\dot{x})$ and $sat(s,\varphi)$ are defined as:

$$\dot{k}(x,\dot{x}) \ge \eta + F(x) \tag{11}$$

$$sat(s,\varphi) = \begin{cases} -1, & s/\varphi \le -1 \\ s/\varphi, & -1 < s/\varphi < 1 \\ 1, & s/\varphi \ge 1 \end{cases} \tag{12}$$

And η is a positive constant, while φ is the thickness of the boundary layer.

Suppose:

$$\varepsilon(x) = \frac{\prod_{i=1}^{2} \mu_{A_i^{l_i}}(x_i)}{\sum_{l_1=1}^{m_1}\sum_{l_2=1}^{m_2}(\prod_{i=1}^{2} \mu_{A_i^{l_i}}(x_i))} \tag{13}$$


```
G61G05.1Q1F10000
X-1.513   Y215.223   Z-164.657   A86.462   C-91.007
X-1.426   Y216.091   Z-165.266   A86.556   C-91.002
X-1.339   Y216.975   Z-165.907   A86.659   C-91.992
X-1.253   Y217.874   Z-166.575   A86.77    C-90.976
X-1.166   Y218.784   Z-167.264   A86.887   C-90.956
X-1.08    Y219.706   Z-167.976   A87.011   C-90.929
X- .992   Y220.635   Z-168.701   A87.139   C-90.898
X- .905   Y221.572   Z-169.443   A87.272   C-90.86
X- 82     Y222.512   Z-170.194   A87.408   C-90.815
X- .733   Y223.458   Z-170.958   A87.548   C-90.765
X- .648   Y224.408   Z-171.732   A87.691   C-90.707
M2
```

Figure 4. The program for test

Where the adaptive law of fuzzy sliding controller can be designed as [18]:

$$\dot{\theta} = \gamma e^T p_2 \varepsilon(x) \tag{14}$$

And γ is a positive quantity, P is a definite symmetric matrix and P_2 is the final array of the definite symmetric matrix P. Stability of the adaptive fuzzy sliding controller is guaranteed by the Lyapunov function.

3. Experiment

The experimental platform is a multi-axis CNC system, referred to as GJ-310 CNC. This system is based on the PC architecture, in which the servo board and the I/O board are connected by the SSB bus. The adaptive fuzzy sliding controller with dynamic compensation described above is used as an axis position controller in the motion control component of the GJ-310 CNC.

We selected a program with a simultaneous 5-axis moving to test the proposed adaptive fuzzy sliding controller as shown in Figure 4. The program consisted of a micro-line segment, whose length was approximately 1mm. Corners which were present between the small line segments enabled for an easy detection of the tracking error effect upon machining accuracy.

3.1 Experimental Parameters

The parameter settings of the GJ-310: the axis number is 6, the encoder input equivalent is 16384, the servo periodical time is 2*ms*, the interpolation cycle time is 2*ms*, the maximal error is 0.2*mm*, the maximal shape error is 0.05*mm*, and the other parameters are shown as table 1.

According to established guidelines of adjusting numerical control machines, we obtained the control rules, as shown in table 2. Triangle membership functions of the input variables are shown in Figure 5.

3.2 Experiment Results

When machining the program (Figure 4) with the GJ-310 CNC system, while separately using the adaptive fuzzy sliding controller with dynamic compensation and the PID controller, we can get the position of each axis, as shown in Figure 6. In Figure 6, series 1 is the ideal position of each axis generated by the trajectory planner (indicated with a dot), series 2 is the actual position of each axis generated by the PID controller (indicated with a dash) and series 3 is the actual position of each axis generated by the adaptive fuzzy sliding controller with dynamic compensation (indicated with a solid). We obtained the tracking errors of each axis as shown in Figure 7, where series 1 is the tracking error generated by the PID controller (indicated with a dot) and series 2 is the tracking error generated by the adaptive fuzzy sliding controller with dynamic compensation (indicated with a solid). Note, the starting position of the program is indicated by -1*mm*, 215*mm*, -150*mm*, 86*mm*, -91*mm*.

3.3 Experimental Analysis

From Figure 6 it is clear that the adaptive fuzzy sliding

Table 1. The parameters of GJ-310 CNC

category / Parameters	axis X	Axis Y	Axis Z	units	Axis A	Axis B	Axis C	units
D/A channel	1	2	3		5	6	8	
Encoder channel	1	2	3		5	6	8	
Proportional gain	33.33	33.33	33.33		33.33	33.33	33.33	
Integral gain	0.1	0.1	0.1		0.1	0.1	0.1	
Differential gain	0.01	0.01	0.01		0.01	0.01	0.01	
Maximum velocity	30000	40000	40000	*mm/min*	20000	25000	30000	*deg/min*
Maximum error	15	15	15	*mm*	10	10	10	*arcmm*
Output offset	0.1099	0.2004	0.1061	*mm*	0.1796	0.2083	-0.1091	*arcmm*
Jog velocity	18000	18000	18000	*mm/min*	15000	15000	15000	*deg/min*
Transfer velocity	20000	20000	20000	*mm/min*	15000	17000	18000	*deg/min*
Maximum voltage	8	8	8	*v*	8	8	8	*v*
Jog acc-time	400	400	400	*ms*	600	600	600	*ms*
Transfer acc-time	400	400	400	*ms*	600	600	600	*ms*

Table 2. The control rules

The output u		Error change rate Δ*e*						
		LN	MN	SN	ZZ	SP	MP	LP
The Tracking Error *e*	LP	LP	LP	LP	LP	LP	LP	LP
	MP	SP	SP	SP	MP	MP	LP	LP
	SP	ZZ	SP	SP	SP	MP	MP	LP
	ZZ	ZZ	ZZ	ZZ	ZZ	ZZ	ZZ	ZZ
	SN	LN	MN	MN	SN	SN	SN	ZZ
	MN	LN	LN	MN	MN	SN	SN	SN
	LN	LN	LN	LN	LN	LN	LN	LN

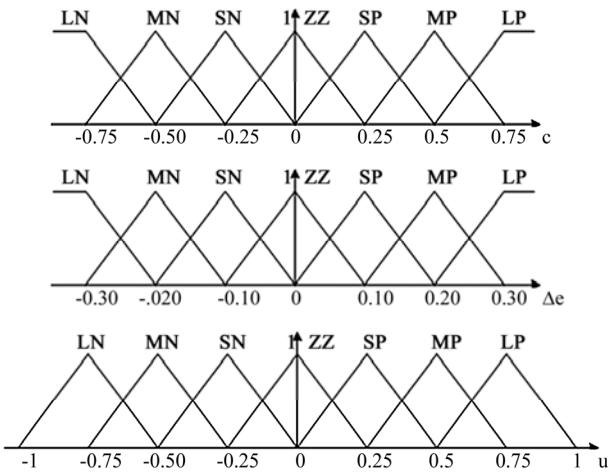


Figure 5. The triangle membership functions of the input variables

controller with dynamic compensation and the PID controller produce nearly identical ending points. Moreover, use of the adaptive fuzzy sliding controller with dynamic compensation results in smooth control, thereby producing an accurate tracking of the trajectory with no overshooting or vibration in the machining.

From Figure 7 it is apparent that the adaptive fuzzy sliding controller with dynamic compensation can reduce tracking errors of each servo axis. As one example of the program, an examination of axis Z which has the longest moving trajectory, provides for a means of comparison of the effects between the two controllers. The adaptive fuzzy sliding controller with dynamic compensation reduced the tracking error by almost 44% of the PID controller and the tracking error of axis C, whose moving trajectory was the shortest in the example of program, was reduced by about 19% of the PID controller.

From the above simulation results, the proposed adaptive fuzzy sliding controller with dynamic compensation demonstrates a perfect performance, which can abolish effects of the system trace performance caused by tracking nonlinearities and uncertainties disturbances. In this way, the control system can produce a high degree of precision in multi-axis machining.

4. Conclusions

Strictly speaking, the CNC system typically has more than one axis. Therefore, an ideal linear system is not available. In this way, research is directed at resolving nonlinearity and uncertainty control problems of multi-axis machining tools, and reducing tracking errors of multi-axis. CNC systems have important implications for

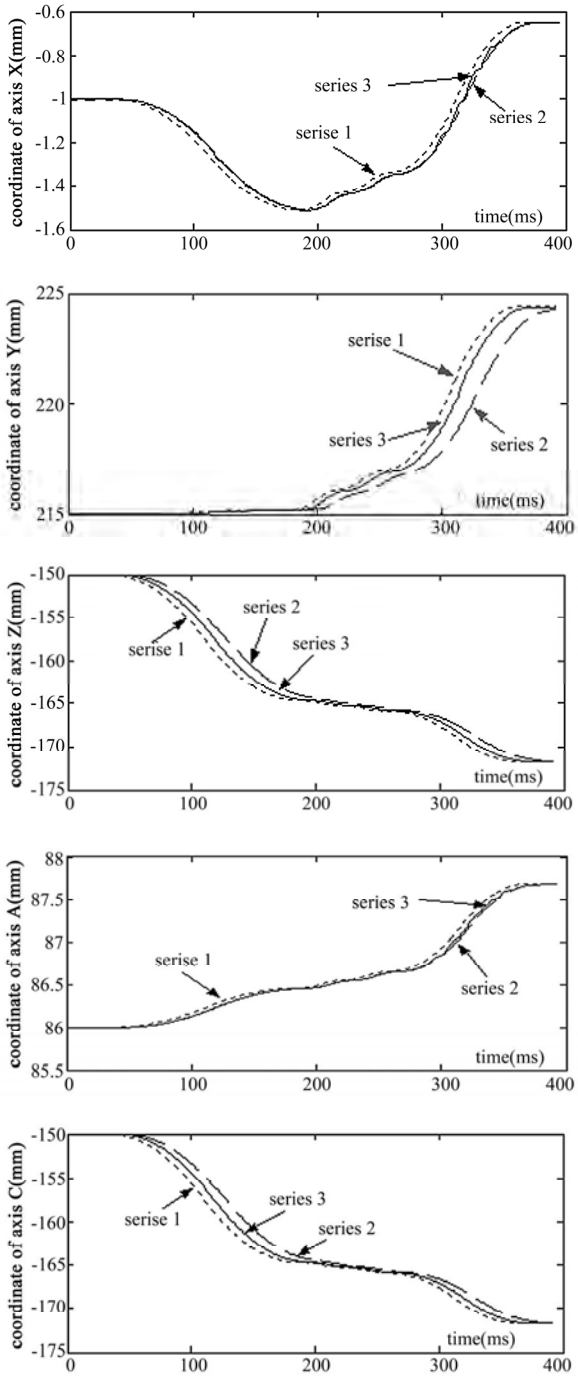


Figure 6. The position of each axis

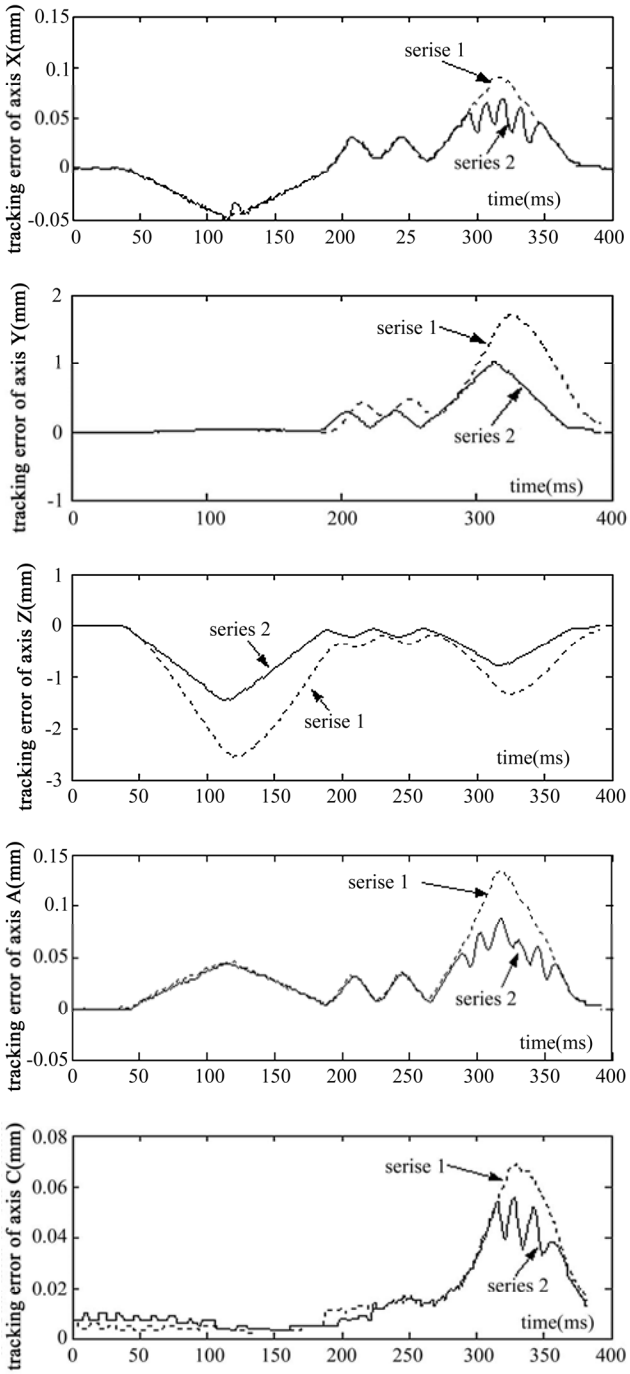


Figure 7. The tracking error of each axis

both theoretical considerations and practical applications. This paper combines dynamic compensation control with adaptive fuzzy sliding control for the design of an adaptive fuzzy sliding controller with dynamic compensation. The results of experiment on the GJ-310 system show that this controller can eliminate overshooting and vibration in a CNC machining control system, and improve the precision of multi-axis machining.

5. Acknowledgment

This paper was supported by National Technology Support Program of Ministry of Science and Technology under Grant 2007BAP20B01 and Chinese Academy of Sciences Knowledge Innovation Program under Grant KGCX2-YW-119.

REFERENCES

[1] R. Ramesh, M. A. Mannan, and A. N. Poo, "Tracking and contour error control in CNC servo systems," International Journal of Machine Tools and Manufacture, Vol. 45, No. 3, pp. 301–326, 2005.

[2] C.-C. Lo, "A tool-path control scheme for five-axis machine tools," International Journal of Machine Tools and Manufacture, Vol. 42, No. 1, pp. 79–88, 2002.

[3] E. Yeşil, M. Güzelkaya, and I. Eksin, "Self tuning fuzzy PID type load and frequency controller," Energy Conversion and Management, Vol. 45, No. 3, pp. 377–390, 2004.

[4] L. Wang, W. Du, H. Wang, and H. Wu, "Fuzzy self-tuning PID control of the operation temperatures in a two-staged membrane separation process," Journal of Natural Gas Chemistry, Vol. 17, No. 4, pp. 409–414, 2008.

[5] B. Wu, H. Lin, D. Yu, R. F. Guo, and R. L. Gai, "Design of fuzzy sliding-mode controller for machine tool axis control," The 33rd International Conference on Computers and Industrial Engineering, CIE270, 2004.

[6] Y. Zhang, F. Wang, T. Hesketh, D. J. Clements, R. Eaton, "Fault accommodation for nonlinear systems using fuzzy adaptive sliding control," International Journal of Systems Science, Vol. 36, No. 4, pp. 215–220, 2005.

[7] S.-J. Huang and W.-C. Lin, "Adaptive fuzzy controller with sliding surface for vehicle suspension control," IEEE Transactions on Fuzzy Systems, Vol. 11, No. 4, pp. 550–559, 2003.

[8] S.-J. Huang and H.-Y. Chen, "Adaptive sliding controller with self-tuning fuzzy compensation for vehicle suspension control," Mechatronics, Vol. 16, No. 10, pp. 607–622, 2006.

[9] R. J. Wai, C. M. Lin, and C. F. Hsu, "Adaptive fuzzy sliding-mode control for electrical servo drive," Fuzzy Sets and Systems, Vol. 143, No. 2, pp. 295–310, 2004.

[10] R. Shahnazi, H. M. Shanechi, and N. Pariz, "Position control of induction and DC servomotors: A novel adaptive fuzzy PI sliding mode control," IEEE Transactions on Energy Conversion, Vol. 23, No. 1, pp. 138–147, 2008.

[11] C.-S. Chen, "Dynamic structure adaptive neural fuzzy control for MIMO uncertain nonlinear systems," Information Sciences, Vol. 179, No. 15, pp. 2676–2688, 2009.

[12] D. Q. Truong and K. K. Ahn, "Force control for hydraulic load simulator using self-tuning grey predictor - fuzzy PID," Mechatronics, Vol. 19, No. 2, pp. 233–246, 2009.

[13] J. Jamaludin, N. A. Rahim and W. P. Hew, "Development

of a self-tuning fuzzy logic controller for intelligent control of elevator systems," Engineering Applications of Artificial Intelligence, pp. 1–12, 2009.

[14] R. S. Blom, "Design and development of a real-time trajectory planner for the enhanced machine controller," Intelligent Systems Division Gaithersburg, Maryland United States of America : National Institute of Standards and Technology Manufacturing Engineering Laboratory, 1999.

[15] J. E. Slotine, "Sliding controller design for nonlinear systems," International Journal of Control, Vol. 40, No. 2, pp. 421–434, 1984.

[16] C. L. Hwang and C. Y. Kuo, "A stable adaptive fuzzy sliding mode control for affine nonlinear systems with application to four-bar linkage systems," IEEE Transactions on Fuzzy Systems, Vol. 9, No. 2, pp. 238–252, 2001.

[17] I. Eker, S. A. Akinal, "Sliding mode control with integral augmented sliding surface: Design and experimental application to an electromechanical system," Electrical Engineering, Vol. 90, No. 3, pp. 189–197, 2008.

[18] R. L. Gai, H. Lin, X. P. Qin, D. Yu, and R. F. Guo, "Adaptive fuzzy sliding control design for the axis system with dynamic multiple kinds of nonlinearities and uncertainties," Proceedings of the 38th International Conference on Computers and Industrial Engineering. Vol. 3, pp. 3006–3011, 2008.

J. Software Engineering & Applications, 2009, 2: 295-300
doi:10.4236/jsea.2009.24038 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

Research of Publish and Subscribe Model Based on WS-Notification

Huilian FAN[1], Guangpu ZEN[1], Xianli LI[2]

[1]School of Mathematics and Computer Science, Yangtze Normal University, Fuling, China; [2]Chongqing college of Electronic Engineering, Chongqing, China.
Email: fhlmx@163.com

Received August 14th, 2009; revised September 14th, 2009; accepted September 27th 2009.

ABSTRACT

WS-Notification bundle of standards, WS-BaseNotification, WS-Topics, and WS-BrokeredNotification, can be used as a general purpose publish/subscribe interface for Service Oriented Architectures. We provide an overview of the WS-Notification specification and describe a modified publish and subscribe model based on WS-Notification. The model is an adaptive policy-driven notification framework that can help enterprises to meet the flexibility and responsiveness requirements of the enterprise. With the modified publish/subscribe model, information consumers can dynamically and declaratively create and configure entities on their behalves to manage their distribution requirements.

***Keywords*:** *WS-Notification, Publish/Subscribe, Notification Broker Service, Notification Consumer Proxy Service*

1. Introduction

In a Service Oriented Architecture (SOA), there is often a need to monitor situations. This occurs to a computer management perspective or a much more broad scope of keeping up to date on real world events such as weather, financial transactions, etc. To monitor these events, a client can poll status changes or subscribe for status changes. In polling case, the client is configured to actively ask the resource for its latest state. The more frequently the client polls state, the more likely the client has an accurate resource representation. However, frequent polling requires both of bandwidth and resources to handle the connection. Thus polling is useful when monitoring timeliness is not an issue or network and hardware resources are abundant. But in the SOAP world, polling is less common as a client typically receives requests from the producer of events. With this peer to peer style, a publish/subscribe system can be created. In this system, the client requests that notifications be sent when they occur. This reduces the latency between the event occurring and the client processing.

WS-Notification has been standardized by OASIS and it is a standard that can solve this business problem of event distribution in heterogeneous complex event processing systems. It specifies an interface for consumer to subscribe, filter notifications, and manage subscriptions. It is also for publishers to send notifications. Further, it describes a notification broker to allow for scaling of the system [1].

2. WS-Notification

Associated with the WS-Resource Framework, IBM, Sonic, and other companies introduced a family of related specifications called WS-Notification. The basic idea behind WS-Notification is to standardize the way that a Web service can notify interested parties (other Web services) that something of interest has happened. It is not meant to replace all messaging infrastructure such as low latency buses, industry standards, or existing infrastructure like JMS. However, WS-Notification systems should be able to integrate with these systems through simple adapters.

Obviously, the key value to WS-Notification is its ability as a standard to allow for greater interoperability with a greater number of vendors and, thus, a lower cost for implementation. The key features of WS- Notifications allow for it to be used as well in general purposes publish/subscribe situations. It defines a unified message format to achieve interoperability between kinds of systems, procedures and components in different platforms and systems, and it defines a set of a standard Web services approach using a topic-based publish/subscribe notification pattern .

WS-Notification family is made up of the following three components [2]: WS-Topics, WS-BaseNotification and WS-BrokeredNotification. Figure 1 shows the relationship between them.

Based on the WS-Notification, the publish/subscribe model is needed to handle the real-world information

integration scenario by allowing a subscriber to specify on behalf of an information consumer filtering rules and policy constraints, not only to select what types of messages or contents the subscriber wants the consumer to receive, but also to specify transformation, scheduling, distribution, and other constraints to be applied to selected messages before they reach the consumer. The architecture must enable the generation (on behalf of consumer) of a proxy service, or agent, which includes the engines to enforce and manage these constraints at run-time. The proxy service can receive the messages on behalf of the information consumer and apply the specified constraints to the message before delivering it to its consumer [3].

2.1 WS-Topics

The WS-Topics specification [4] defines a mechanism to organize and categorize items of interest for subscription known as "topics." This is achieved by associating each notification with a topic, and means that subscribers can define the specific category of event that they are interested in hearing about. A web service can publish a set of topics used to organize and categorize a set of notification messages that clients can subscribe, and receive a notification whenever the topic changes.

WS-Topics are very versatile, as they even allow us to create topic trees, where a topic can have a set of child topics. By subscribing to a topic, a client automatically receives notifications from all the descendant topics (without manual subscribing to each of them). As part of the publication of a notification-message, the publisher associates it with one or more topics.

WS-Topics also provide a coarse-grained filtering mechanism which allows large sets of uninteresting notifications to be excluded quickly. For example, in a sport results scenario a subscriber can indicate that he or she is only interested in receiving notifications about football, which excludes any events about baseball or hockey. More fine-grained control of filtering can be achieved using other filtering mechanisms, such as the message content filter defined in WS-BaseNotification. For example, by selecting only those football games in which the home team beat the away team. In many situations, the topic does not actually appear in the body of the notification message itself since the classification of the notification is made at a higher level than the generation of the notification content.

In order to avoid naming collisions, and to facilitate interoperation between independently developed Notification Producers and Subscribers, every WS-Notification Topic is assigned to an XML Namespace. The set of Topics associated with a given XML Namespace is termed a Topic Namespace.

2.2 WS-BaseNotification

The WS-BaseNotification specification defines the standard Web services interfaces for Notification Producers and Notification Consumers. It includes standard message exchanges to be implemented by service providers who wish to act in these roles, along with operational requirements expected by them. Notification producers have to expose a subscribing operation that notification consumers can use to request a subscription. Consumers, in turn, have to expose a notify operation that producers can use to deliver the notification [5,6]. Figure 2, "A typical WS-Notification interaction" shows that the five primary entities how to work together to pass data through the WS-BaseNotification. Initially, the subscriber is responsible for setting up a subscription between the NotificationProducer Web service and a NotificationConsumer Web service. This subscription is managed by the SubscriptionManager Web service working on behalf of the producer. Subsequently, when a Situation is observed by the Publisher, the Publisher creates a notification message and passes it to the NotificationProducer. It is the responsibility of the producer to establish whether the notification message matches the registered subscription or not, and, if so, send the notification message to the consumer.

2.3 WS-BrokeredNotification

Even in the simplest publish/subscribe environment, the amount of connections and boot strapping information can grow very quickly. If there are only *2* publishers and *2* consumers, and each consumer wants to be notified from each publisher, *4* connections need to be set up.

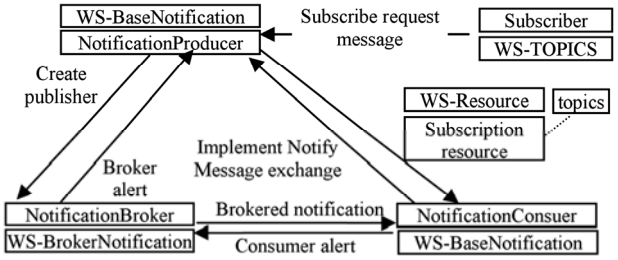


Figure 1. Relationship between WS-Topics, WS- aseNotification and WS-BrokeredNotification

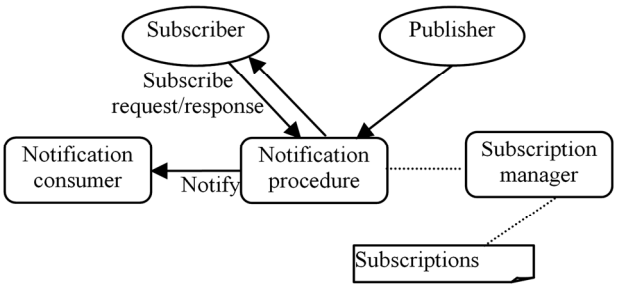


Figure 2. A typical WS-Notification interaction

Add one more consumer and you now have *6* connections. The number of connections starts to grow very quickly as more distributors and consumers are added; the required number of connections for *m* publishers and *n* consumers are $m \times n$ connections. To simplify this topology, WS-Notification standardizes a notification broker which acts as an intermediary between publishers and consumers. Here, publishers and consumers are decoupled from each other and substitute only required boot strap information to the broker. Therefore, in the scenario of *m* publishers and *n* consumers, the required number of connections is $m + n$.

The WS-BrokeredNotification specification extends the definitions made in the WS-BaseNotification specification to define the concept of NotificationBroker, which is an intermediary service to those producers and consumers can connect in order to pass notifications. Critically, the NotificationBroker is capable of accepting subscription requests from consumers, as well as receiving notification messages from producers. The broker is then responsible for matching the notifications with the subscriptions and sending them to the consumer. In this way, the broker takes on more painstaking functions of the producer, freeing developers of producer applications to concentrate on the business task of observing situations and generating the appropriate notifications without worrying about the challenge but mechanical task of managing subscriptions and matching them to notifications. Advantages of the brokered notification pattern are as follows:

- Relieves the publisher of having to implement message exchanges associated with the notification producer. For example, managing subscriptions (SubscriptionManager) and distributing notifications (NotificationProducer).
- Avoids the need for synchronous communications between the producer and the consumer.
- Can reduce the number of inter-service connections and references.
- Acts as a finder service. For example, if a new publisher is added that publishes notification *x*, a consumer does not have to issue a new subscription if it has already subscribed to the broker with *x*.
- Provides anonymous notification, which means that publishers and consumers do not need to be aware of each other's identity.

In many scenarios, the NotificationBroker service is implemented by a middleware provider, ensuring that the brokering facilities are written to enterprise quality expectations and often provide additional value-add services over and above the basic definition of the service, for example, logging, transformation, or quality of service enhancements above those required by the specification [7]. As shown in Figure 3, "A typical brokered WS-Notification interaction", the producer must register

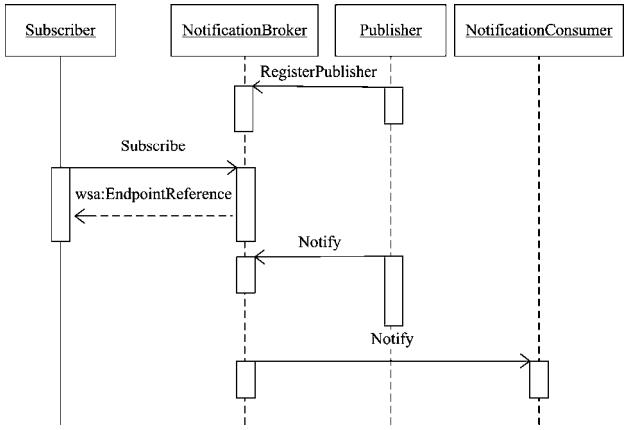


Figure 3. A typical brokered WS-Notification interaction

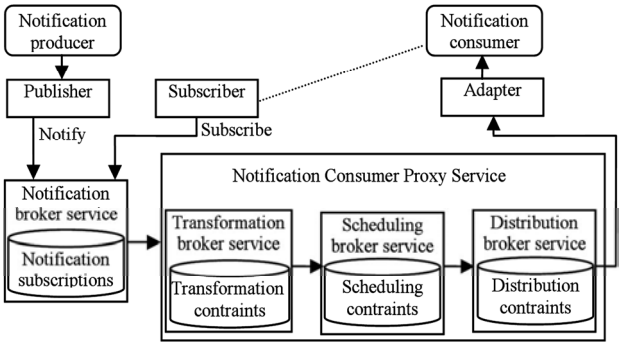


Figure 4. The Publish/Subscribe architectural model

with the broker and publish its topics there. The subscriber must also subscribe through the broker, not directly with the producer. Finally, when a notification is produced, it is delivered to the consumer through the broker.

3. Publish/Subscribe Model Based on WS-Notification

The modified publish/subscribe model is as shown in Figure 4. It extends the basic publish and subscribe pattern by extending the subscription capabilities to include the specification of transformation, distribution, and scheduling constraints as part of publish and subscribe subscription [8].

Additionally, this architecture enables non-pub/sub-enabled systems (that is, information consumers which are not able to consume notification messages of the pub/sub system) to participate in the pub/sub pattern by allowing the model to dynamically create a proxy service to receive pub/sub notifications on behalf of the consumer. This is the Notification Consumer Proxy Service (NCPS) shown in Figure 4, which also manages the distribution of notifications to the consumer based on the transformation, distribution, and scheduling constraints specified by the consumer upon subscribing.

As shown, the model highlights the following components:

Notification producer: Contains information of interest for consumer. Good examples of information producers are systems that manage business information for an enterprise and include master data stores for customer, product, order information, and so on, in addition to enterprise operational data stores.

Notification consumer: Depends on and must consume information from an information producer. For example, many enterprise business applications like order fulfillment systems depend on data from the business information sources.

Subscriber: Requests creation of a subscription. It sends a subscribe request message to a notification broker (pub/sub broker). The subscribe request message identifies a notification consumer. A subscription is an entity that represents the relationship between an information consumer and an information producer. It records the fact that the consumer is interested in some or all of the notifications that the producer can provide. It can contain filter expressions, and may be long-running or have a limited lifetime.

Publisher: Creates notification message instances. A publisher receives information from entities in the information producer that monitor and detect a situation. A situation is an occurrence that is noted by one party and is of interest to other parties. A notification is a one-way message that conveys information about a situation to other services.

Notification broker service: Performs a notification broker function between notification consumers and notification producers, and it is responsible for sending notifications to the appropriate consumers. It also acts as a subscription manager and manages requests to query, delete, or renew subscriptions.

Notification Consumer Proxy Service (NCPS): Receives notifications from the notification broker on behalf of the information consumer. Typically, the consumer is not able to receive notification messages, hence the need for this service to act on its behalf, collect the notifications, perform some business logic (if the scenario calls for it), enforce the transformation, scheduling, and distribution constraints for the consumer, and then send the results to the consumer.

Adapter: An entity that enables the interaction with an information consumer.

To demonstrate the publish/subscribe model, event distribution was applied to solve a situation of teacher-student interaction. The situation occurs when students have questions to ask teacher. WS-Notification was used in the scenario for the notifications from the student to the broker and from the broker to the teacher. This was done through a number of different steps:

1) The teachers registered their interest with the broker.

2) The students were either configured to send notifications to the broker or configured to register themselves as a publisher to the broker.

3) If students registered themselves as a publisher to the broker, the broker would ask for notifications from the students.

4) The students would send question notifications to the broker.

5) The broker would forward those question notifications to interested teacher.

In the situation, teacher is not only subscriber but also information consumer, and student is information producer. From user perspective, there are two interfaces: the student and the teacher. The student interface is responsible for allowing students to ask questions. The teacher interface allows teachers to monitor notifications as they arrive. The interfaces of the publish/subscribe model among the teacher, notificationbroker, NCPS and one of the students are following:

1) Subscribe: A subscriber, on behalf of the notification consumer, sends an XML subscription request message to notification broker service. This subscription message specifies transformation, distribution, and scheduling constraints for the information consumer, as well as the basic subscription constraints on information content from the producer that the consumer is interested in.

For a Web service to act as a NotificationProducer it must support the Subscribe message exchange – that is to say the WSDL for the Web service must be included in its portType definition an operation that contains the subscribe request and response messages defined by the WS-Notification specification. By implementing the Subscribe exchange, the producer service is required to send a notification to each notification consumer with a subscription registered whenever the producer has a message be sent and the filter conditions expressed in the subscription are satisfying. In the example, the teacher subscription request is a message sent from the teacher to the broker to request notifications be sent. It contains the type id for a question that the teacher wishes to answer it. In this subscription request, the *Teacher* requests for notifications for question type with id *C001*, as shown in the Listing 1.

Two of the elements of the above request message are of particular interesting in the Publish/Subscribe environment:

- The ConsumerReference is a WS-Addressing endpoint reference that identifies the location of the Notification Proxy Service Consumer service to which matching notification messages will be sent by the producer.
- The Filter element, and in particular the TopicExpression filter, is used to determine the specific subset of all available notification messages that should match this new subscription. This is important because the majority of notifications are of interest only for a small number of

Listing 1. SOAP body contents of a subscribe request for notification delivery

```
<wsnt:Subscribe xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2">
      <wsnt:ConsumerReference>
          <wsa:Address>

http://notify.yznu.cn:9080/Teacher/NCProxyServiceEndpoint
          </wsa:Address>
      </wsnt:ConsumerReference>
      <wsnt:Filter>
     <wsnt:TopicExpression Dialect="http://docs.oasis-
open.org/wsn/t-1/TopicExpression/Simple"

xmlns:typeID="http://www.ibm.com/xmlns/wsn/question/typeID">
                typeID:C001
          </wsnt:TopicExpression>
     </wsnt:Filter>
<wsnt:InitialTerminationTime>P0Y0M0DT1H0M0S</wsnt:InitialTer
minationTime>
</wsnt:Subscribe>
```

consumers, so we improve the efficiency of the system by avoiding sending unwanted notifications.

2) Notify: The publisher entity creates a notification message when it receives information from entities in the notification producer that monitor and detect a situation, such as put new questions that are of interest for consumers. The publisher sends the notification message to notification broker so it can be distributed to the appropriate consumers that have subscribed to that message.

3) Notification brokering: The notify message sent by the publisher is routed by the notification broker service to the appropriate notification consumer proxy service. The notification broker matches notification messages to the consumers that are subscribed to these notifications. The information consumer proxy service receives the notification message and performs whatever business logic it needs to do in order to create and aggregate the appropriate response to the notification consumer. For example, the consumer proxy service might need to access additional information from the information producers to get all the needed data associated with the change. As a result of applying its specific business logic, the notification consumer proxy service assembles a message or a set of messages to be sent to the consumer.

A NotificationBroker may be a WS-Resource, and if it is, it must support the required message exchanges defined by the WS-ResourceProperties specification and MUST also support message exchanges and may support Resource Property elements defined by the following interfaces:

- NotificationProducer
- NotificationConsumer
- RegisterPublisher

The NotificationBroker portType aggregates the three portTypes and it is not the only way to implement a broker. A distributed broker implementation can be achieved by hosting NotificationProducer, NotificationConsumer, or RegisterPublisher portTypes at one or more physical endpoints.

4) Apply constraints:

Transformation constraints: The notification message is applied against the transformation constraints to determine what transformation module or modules to apply to the message.

Scheduling constraints: The scheduling service applies the scheduling policy constraints if any were specified as part of the subscription. These constraints relate to the specified delivery schedule for the information consumer. Notifications that cannot be transmitted due to the conditions specified in the policy are queued by the scheduling service for delivery during the next available window.

Distribution constraints: The distribution service applies the distribution policy constraints if any were specified as part of the subscription. One distribution policy specifies the size limit of a notification message transmitted to the consumer. If a notification message exceeds this size, it will be broken into a sequence of pieces which are smaller than the size limit and transmitted to the consumer individually by the distribution service.

5) Deliver: Once the model satisfies the scheduling and distribution constraints mentioned in the previous step, it sends the message to the information consumer through an adapter service.

4. Conclusions

This paper discusses how to implement a general purpose publish/subscribe interface for a Service Oriented Architecture through the WS-Notification bundle of standards, WS-BaseNotification, WS-Topics, and WS-BrokeredNotification. We describe an adaptive, policy-driven notification architectural model that can support a generalized publish/subscribe interaction pattern. This model is based on the WS-Notification standards, a set of reusable integration services. We introduce the teacher-student interactive scenario to help to demonstrate the WS-Notification features and explain that the publish/subscribe model is the standard of choice for event distribution and processing.

5. Acknowledgements

This work is supported by the Natural Science Project of Chongqing Municipal Education Commission (Project No.KJ091305)

REFERENCES

[1] R. B. Chumbley and J. D. Eisinger, “Leveraging key WS-notification features in your business applications,” April 2009, http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-wsnotificationWAS7/ws-wsnotification

WAS7-pdf.pdf.

[2] Sams Publishing, WS Notification and WS Topics in the WS Resources Framework, July 2006, http://www.devarticles.com/c/a/Web-Services/WS-Notification-and-WS-Topics-in-the-WS-Resources-Framework/.

[3] K. Czajkowski, D. Ferguson, I. Foster etc., WS-Resource Framework, 9th June 2004, http://www.globus.org/wsrf/specs/ws-wsrf.pdf.

[4] W. Vambenepe, S. Graham, and P. Niblett, 9th October 2006, http://wsn-ws_topics-1.3-spec-c, http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-cs.

[5] B. Sotomayor, "The globus toolkit 4 programmer's tutorial," August 19th 2007, http://gdp.globus.org/gt4-tutorial/multiplehtml/ch08s02.html.

[6] W. Vambenepe, S. Graham, and P. Niblett, August 9th 2006, http://wsn-ws_base_notification-1.3-spec-cs.pdf, http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-cs-01.pdf.

[7] W. Vambenepe, S. Graham, and P. Niblett, August 9th 2006, http://wsn-ws_brokered_notification-1.3-spec-cs, http://docs.oasis-open.org/wsn-/wsn-ws_brokered_notification-1.3-spec-cs-01.pdf.

[8] A. Bou-Ghannam and M. Roberts, "GPASS: A generalized publish and subscribe solution using WS-Notification standards," August 2007, http://www.ibm.com/developerworks/websphere/library/techarticles/0708_boughannam/0708_boughan nam.html.

J. Software Engineering & Applications, 2009, 2: 301-307
doi:10.4236/jsea.2009.24039 Published Online November 2009 (http://www.SciRP.org/journal/jsea)

A Hybrid Importance Sampling Algorithm for Estimating VaR under the Jump Diffusion Model

Tian-Shyr Dai[1]; Li-Min Liu[2]

[1]Department of Information and Financial Management, Institute of Information Management and Institute of Finance, National Chiao-Tung University, Taiwan ,China; [2]Department of Applied Mathematics, Chung Yuan Christian University, Taiwan, China.
Email: d88006@csie.ntu.edu.tw

Received July 20th, 2009; revised August 12th, 2009; accepted August 14th, 2009.

ABSTRACT

Value at Risk (VaR) is an important tool for estimating the risk of a financial portfolio under significant loss. Although Monte Carlo simulation is a powerful tool for estimating VaR, it is quite inefficient since the event of significant loss is usually rare. Previous studies suggest that the performance of the Monte Carlo simulation can be improved by importance sampling if the market returns follow the normality or the distributions. The first contribution of our paper is to extend the importance sampling method for dealing with jump-diffusion market returns, which can more precisely model the phenomenon of high peaks, heavy tails, and jumps of market returns mentioned in numerous empirical study papers. This paper also points out that for portfolios of which the huge loss is triggered by significantly distinct events, naively applying importance sampling method can result in poor performance. The second contribution of our paper is to develop the hybrid importance sampling method for the aforementioned problem. Our method decomposes a Monte Carlo simulation into sub simulations, and each sub simulation focuses only on one huge loss event. Thus the performance for each sub simulation is improved by importance sampling method, and overall performance is optimized by determining the allotment of samples to each sub simulation by Lagrange's multiplier. Numerical experiments are given to verify the superiority of our method.

Keywords*: *Hybrid Importance Sampling, VaR, Straddle Options, Jump Diffusion Process

1. Introduction

Value at Risk (VaR) is an important tool for quantifying and managing portfolio risk. It provides a way of measuring the total risk to which the financial institution is exposed. VaR denotes a loss ℓ that will not be exceeded at certain confidence level $1-p$ over a time horizon from t to $t+\Delta t$. To be more specific,

$$P(V_{t+\Delta t} - V_t < \ell) = p\ ,$$

where V_τ denotes portfolio value at time τ. Typically, p is close to zero. For convenience, we define $V_{t+\Delta t} - V_t$ and $(V_{t+\Delta t} - V_t)/V_t$ as the portfolio gain and the return over the time span Δt. Some academic papers focus on a relevant problem: computes the probability p of a portfolio loss to exceed a given level ℓ [1], and our paper will focus on this problem.

VaR can not be evaluated by simple yet exact analytical formulas when the assumptions on the processes of the assets' values or the composition of financial portfolios are complex [2]. The asset in this paper is assumed to be stock for convenience. The Monte Carlo simulation is a flexible and powerful tool to estimate VaR since it is usually more easily to sample the stock prices from complex diffusion price processes than to estimate the distributions of the stock prices at a certain time point. We can repeatedly evaluate possible future values of a financial portfolio by sampling prices of stocks that compose the portfolio and the distribution of the portfolio gain can then be estimated. However, estimating VaR by the Monte Carlo simulation is very inefficient since the event that the portfolio loss exceeds ℓ is rare (note that p is close to zero) and a large number of samples is thus required to obtain an accurate probability estimate of this rare event. By assuming the market returns follow normal distributions, Glasserman *et al.* develop an efficient variance reduction method based on importance sampling that can drastically reduce the number of samples required to achieve accurate probabilities estimates of rare events [3]. In their method, the stock prices are sampled from a new probability measure where the event of significant loss is more likely to happen than in the original one. This new probability measure is selected to

"asymptotically minimize" the second moment of the estimator for estimating $P(V_{t+\Delta t} - V_t < \ell)$. (Details will be introduced in Section 2.)

Empirical studies claim that the stock returns observed from the real world markets show higher peaks and heavier tails than what is predicted by a normal distribution as illustrated in Figure 1 [4−6]. For estimating VaR, the heavy tail phenomenon must be taken into account since this phenomenon causes a significant loss of the stock price more likely to happen. To address this problem, Glasserman *et al.* extend their work by assuming that the stock returns follow t distributions [7]. Indeed, most financial papers address the aforementioned problem by assuming that the stock prices follow the jump-diffusion model [8], GARCH models [9], or the stochastic volatility model [10] instead of t distribution. The first contribution of this paper is to extend Glasserman *et al.* [7] to the jump diffusion model, which assumes that the stock returns and the jump sizes follow normal distributions and the arrival of jumps is modeled by a Poisson process. In this paper, the probability distributions of the stock returns, jump sizes, and the arrival of jumps are probably tilted to "asymptotically minimize" the second moment for estimating the probability of the huge loss event.

Glasserman's method performs poorly for portfolios of which huge loss is triggered by significantly distinct events. Take a portfolio, shorting straddle options (which will be introduced later), illustrated in Panel (a) of Figure 2 as an example. This portfolio suffers significant loss when the stock price increases or decreases drastically. Thus tilting the probability measure of the stock price to make one huge-loss event, says a significant decrease in the stock price, more likely to happen will make the other event (a significant increase in the stock price) much rarer. The numerical results in our paper show that naively applying Glasserman's method deteriorates the performance. Glasserman *et al.* argue that the aforementioned problem can be solved by the delta-gamma approximation [11,12] if the portfolio gain can be well approximated by a quadratic function of the stock price. But it is obvious that many portfolios, like the shorting straddle options and the three-minimum portfolio (see panel (b) of Figure 2) can not be well approximated by quadratic functions.

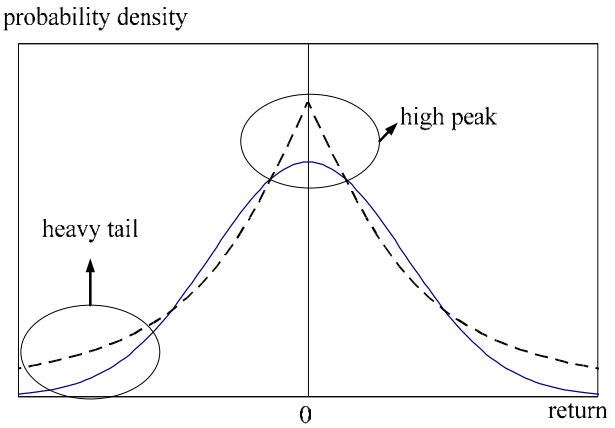


Figure 1. High peaks and heavy tails of stock returns

The solid line denotes the return modeled by a normal distribution and the dashed line denotes the return modeled by a t distribution, which is closer to the distribution of the real world market returns than the former distribution

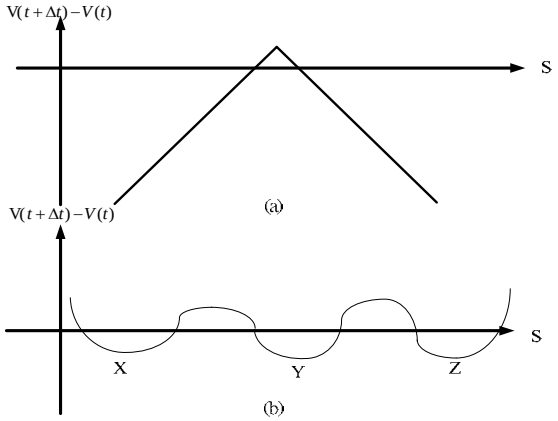


Figure 2. The relationship between the stock price and the portfolio gain

The *x*- and *y*-axis denote the stock price and the portfolio gain, respectively. Panel (a) denotes the case of shorting straddle options near the option maturity date. Panel (b) denotes the case of three-minimum portfolio mentioned in [2]. *X*, *Y* , and *Z* denotes there huge-loss events f this portfolio.

The second contribution of this paper is the hybrid importance sampling algorithm to solve the aforementioned problem. The hybrid importance sampling algorithm is composed of sub simulations; each sub simulation focuses on one significant loss event. For example, our algorithm for estimating the probability of huge loss for shorting straddle options can be decomposed into two sub simulations. On focuses on the significant decrease in the stock price and the other focuses on the significant increase in the stock price. The algorithm for the three-minimum portfolio can be decomposed into three sub simulations. These three sub simulations focus on huge-loss events X, Y, and Z, respectively. Each sub simulation tilts its probability measure of the stock price to "asymptotically minimize" the second moment for estimating the probability of the huge-loss event focused by that sub simulation. Finally, the computational resource allocated to each sub simulation is determined by Lagrange's multiplier to asymptotically minimize the second moment for estimating the overall huge loss probability.

Generally speaking, cross-discipline research, like bioinformatics and financial engineering, become more prevailing and important for both academics and practitioners. This paper merges the simulation technique from

applied mathematics and algorithm design and performance comparisons knowhow from computer science discipline to develop efficient numerical programs to solve finance problem. It plays a great platform to interchange the ideas, the challenges, and the techniques among the computer scientists, mathematicians, and financial experts.

The paper is organized as follows. The assumptions of the Merton's jump diffusion model, the Glasserman's importance sampling method, and the definitions of straddle options are introduced in Section 2. In Section 3, we will use shorting straddle options as an example to demonstrate how the probabilities of the jump diffusion process are tilted for each sub simulation and how the number of samples is allocated to each sub simulation to optimize the overall performance. Numerical results in Section 4 verify the superiority of our method. Section 5 concludes the paper.

2. Preliminaries

2.1 The Stock Price Process

Define *St* as the stock price at year *t*. Under the Merton's jump diffusion model, the stock price process can be expressed as

$$\frac{dS_t}{S_{t-}} = \mu dt + \sigma dW_t + (e^X - 1)dN_t \tag{1}$$

where W_t is the standard Wiener process, μ is the average stock return per annum, σ is the annual volatility, X is a normal random variable that models the jump size, and N_t denotes the Poisson process. We further assume that $X \sim N(\eta, \delta^2)$ and $P(dN_t = 1) = \lambda dt$. Define the stock return over the time horizon Δt as follows:

$$r_t \equiv \frac{S_{t+\Delta t} - S_t}{S_t} \approx \mu\Delta t + \sigma\sqrt{\Delta t}Z + \sum_{i=1}^{N_{\Delta t}} Z_i, \tag{2}$$

where $Z \sim N(0,1)$, $N_{\Delta t}$ denotes the number of jumps between time t and $t+\Delta t$, $Z_i \sim N(\eta, \delta^2)$. Note that the aforementioned model degenerates into the Black- Scholes lognormal diffusion process [13] when λ = 0 (i.e. $N_{\Delta t} = 0$ in Equation (2)).

2.2 Glasserman's Importance Sampling Method

This subsection sketches Glasserman's importance sampling method [3] by assuming that the stock price process follows the log-normal diffusion process. Consider a portfolio which is composed of a stock. Let *A* denotes the event that the portfolio gain is less than ℓ:

$$\begin{aligned} A &\equiv \{S(t+\Delta t) - S(t) - \ell < 0\} \\ &= \left\{\frac{\text{S}(t+\Delta t)\text{-S}(t)}{S(t)} - \frac{\ell}{S(t)} < 0\right\} \\ &= \left\{r_t - \frac{\ell}{S(t)} < 0\right\} \qquad (3) \\ &= \{\text{-}r_t - r_p > 0\} \qquad (4) \\ &= \left\{\text{Z: f(Z)} \equiv \text{-}r_t - r_p = -\mu\Delta t - \sigma\sqrt{\Delta t}Z - r_p > 0\right\} \qquad (5) \end{aligned}$$

where we substitute Equation (2) into Equation (3) and (5), and $r_p \equiv \frac{-\ell}{S_t}$ into Equation (4).

To minimize the second moment for estimating the probability of event *A*, Glasserman samples *Z* from a new probability measure $\wp_\theta$ instead of the original probability measure $\wp$ (where $Z \sim N(0,1)$). The likelihood ratio for these two probabilities measures is

$$\frac{d\wp_\theta}{d\wp} = \exp\{\theta f(Z) - \Psi(\theta)\}, \tag{6}$$

where $\Psi(\theta) \equiv \log E\ [\exp\ (\theta f(Z))]$. Define $E_{\wp_\theta}$ as the expected value measured under $\wp_\theta$ and

$$A_\theta \equiv \{Z_\theta : f(Z_\theta) = -\mu\Delta t - \sigma\sqrt{\Delta t}Z_\theta - r_p > 0\},$$

where $Z_\theta \sim N(\theta\sigma\sqrt{\Delta t}, 1)$. Then we have

$$p = E(1_A) = E_{\wp_\theta}\left[1_{A_\theta} \exp(-\theta f(Z_\theta) + \Psi(\theta))\right].d*9$$

The second moment of the estimator is then

Second moment

$$= E_{\wp_\theta}\left[1_{A_\theta} \exp(-2\theta f(Z_\theta) + 2\Psi(\theta))\right] \le \exp(2\Psi(\theta)). \tag{7}$$

To asymptotically optimize the performance of the Monte Carlo simulation, a proper θ is selected to minimize exp ($2\Psi(\theta)$) by the following equation:

$$\Psi'(\theta) = 0 \tag{8}$$

$\wp_\theta$ is then determined by substituting θ (obtained from Equation (8)) into Equation (6).

2.3 Straddle Options

Stock options are derivative securities that give their buyer the right, but not the obligation, to buy or sell the underlying stocks for a contractual price called the exercise price *K* at maturity. Assume that the options mature at time $t+\Delta t$ then the payoffs of a call option and a put option at maturity are $\max(S_{t+\Delta t} - K, 0)$ and $\max(K - S_{t+\Delta t}, 0)$, respectively. Shorting straddle options denotes a portfolio that shorts D_1 units call options and

D_2 units put options with the same strike price. To be more specific, the portfolio gain at maturity is $-D_1 \max\left(S_{t+\Delta t}-K,0\right)-D_2 \max\left(K-S_{t+\Delta t},0\right)$. The portfolio gain is interpreted in terms of stock return r_t defined Equation (2) in Figure 3. Note that

$$r_0^* \equiv \frac{K-S_t}{S_t} \tag{9}$$

The portfolio gain can be expressed as follows:

$$\text{Portfolio Gain} = \begin{cases} r_0^* S_t \delta_2 + (r_t - r_0^*) S_t \delta_1, & \text{if } r_t \geq r_0^*, \\ r_t S_t \delta_2, & \text{if } r_t < r_0^*, \end{cases} \tag{10}$$

where $\delta_1 = -D_1$ and $\delta_2 = D_2$. The portfolio gain is less than ℓ if the stock return r_t is larger than r_1^* or lower than r_2^*, where

$$r_1^* = \frac{\ell - r_0^* S_t \delta_2 + r_0^* S_t \delta_1}{S_t \delta_1} \text{ and } r_2^* = \frac{\ell}{S_t \delta_2}.$$

3. Contributions

We will use shorting straddle options as an example to illustrate the major contributions of this paper. First, the huge loss events of this portfolio are identified. The Monte Carlo simulation is then decomposed into two sub simulations; each focuses on one huge loss event. Next, the probability distribution for each sub simulation is tilted to asymptotically minimize the second moment of the estimator under the jump diffusion assumption. Finally, the allotment of samples for each sub simulation is determined by Lagrange's multiplier to optimize the overall performance.

3.1 Identify the Huge Loss Events

In Figure 3, the portfolio gain of shorting straddle options is less than ℓ when the stock return r_t exceeds threshold r_1^* or is below r_2^*. For convenience, events A_1 and A_2 are used to denote the events $\left\{r_t > r_1^*\right\}$ and $\left\{r_t > r_2^*\right\}$, respectively, as follows:

$$A_1 \equiv \left\{r_t : f_1(r_t) \equiv -r_0^* \delta_2 - (r_t - r_0^*)\delta_1 - r^* > 0\right\}, \tag{11}$$

$$A_2 \equiv \left\{r_t : f_2(r_t) \equiv -r_t \delta_2 - r^* > 0\right\},$$

where $r^* = -\ell / S_t$ and formula $f_1(r_t)$ and $f_2(r_t)$ are derived from Equation (10). Since A_1 and A_2 are mutually exclusive, the probability that the portfolio gain is less than ℓ is

$$p = E\left[1_{A_1 \cup A_2}\right] = E\left[1_{A_1}\right] + E\left[1_{A_2}\right].$$

The Monte Carlo simulation for estimating p can be decomposed into two sub simulations; one focuses on event A_1, and the other one focuses on event A_2.

3.2 Importance Sampling under the Jump Diffusion Assumption

Next, we will describe how to efficiently estimate $E\left[1_{A_1}\right]$ and $E\left[1_{A_2}\right]$ by importance sampling. Assume that the sub simulations for estimating $E\left[1_{A_1}\right]$ and $E\left[1_{A_2}\right]$ tilt their probabilities from $\wp$ to $\wp_{\theta_1}$ and $\wp_{\theta_2}$, respectively. Then θ_1 and θ_2 are derived as follows: Define $\Psi_1(\theta_1) \equiv \log E\left[\exp(\theta_1 f_1(r_t))\right]$ and $\Psi_2(\theta_2) \equiv \log E\left[\exp(\theta_2 f_2(r_t))\right]$. $E\left[\exp(\theta_1 f_1(r_t))\right]$ can be calculated as follows:

$$\begin{aligned} E\left[\exp(\theta_1 f_1(r_t))\right] &= E\left[\exp(\theta_1(-r_0^*\delta_2 - r_t\delta_1 + r_0^*\delta_1 - r^*))\right] \\ &= \exp(-\theta_1 r_0^* \delta_2 + \theta_1 r_0^* \delta_1 - \theta_1 r^*) E(\exp(-\theta_1 r_t \delta_1)). \end{aligned}$$

Note that

$$E(\exp(-\theta_1 r_t \delta_1)) = E\left[\exp(-\theta_1\delta_1(\mu\Delta t + \sigma\sqrt{\Delta t}Z + \sum_{i=1}^{N_{\Delta t}} Z_i))\right]$$

$$= \exp(-\theta_1\delta_1\mu\Delta t + 0.5\sigma^2\Delta t\theta_1^2\delta_1^2) E\left[\exp(-\theta_1\delta_1\sum_{i=1}^{N\Delta t} Z_i)\right],$$

$$E\left[\exp(-\theta_1\delta_1\sum_{i=1}^{N_{\Delta t}} Z_i)\right] = \sum_{n=0}^{\infty} \frac{e^{-\lambda_t}\lambda_t^n}{n!} E\left[\exp(-\theta_1\delta_1\sum_{i=1}^{n} Z_i)\right]$$

$$= \sum_{n=0}^{\infty} \frac{e^{-\lambda_t}\lambda_t^n}{n!} \exp(-n\theta_1\delta_1\eta + 0.5n\theta_1^2\delta_1^2\delta^2$$

$$= \sum_{n=0}^{\infty} \frac{e^{-\lambda_t}(\lambda_t \exp(-\theta_1\delta_1\eta + 0.5\theta_1^2\delta_1^2\delta^2))^n}{n!}$$

$$= \exp(\lambda_t e^{-\theta_1\delta_1\eta + 0.5\theta_1^2\delta_1^2\delta^2} - \lambda_t),$$

where $\lambda_t \equiv \lambda\Delta t$. Thus θ_1 can be obtained by numerically solving the equation $\Psi_1'(\theta_1) = 0$ (see Equation (8)) as follows:

$$\begin{aligned} \Psi_1'(\theta_1) &= -r_0^*\delta_2 + r_0^*\delta_1 - r^* - \delta_1\mu\Delta t + \sigma^2\Delta t\theta_1\delta_1^2 \\ &+ \lambda_{\Delta t}(-\delta_1\eta + \theta_1\delta_1^2\delta^2)\exp(-\theta_1\delta_1\eta + 0.5\theta_1^2\delta_1^2\delta^2) = 0. \end{aligned}$$

Similarly, it can be derived that

$$\begin{aligned} E\left[\exp(\theta_2 f_2(r_t))\right] &= \exp(-\theta_2\delta_2\mu\Delta t + 0.5\sigma^2\Delta t\theta_2^2\delta_2^2 \\ &- \theta_2 r^* + \lambda_t e^{-\theta_2\eta\delta_2 + 0.5\theta_2^2\delta^2\delta_2^2} - \lambda_t). \end{aligned}$$

θ_2 can also be solved numerically by the equation $\Psi_2'(\theta_2) = 0$ as follows:

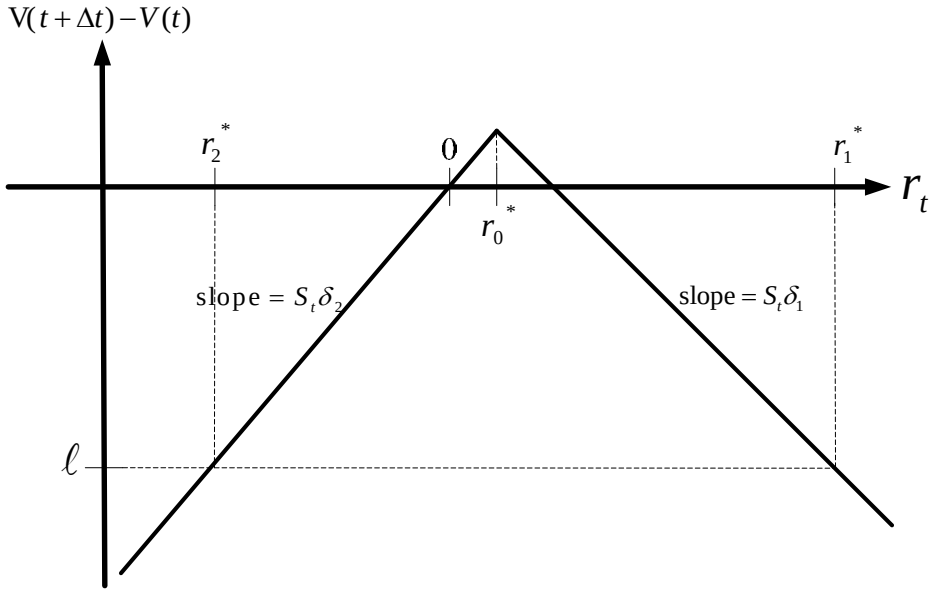


Figure 3. Shorting straddle options
The *x*- and *y*-axis denote the stock return and the portfolio gain, respectively.

$$\Psi'_2(\theta_2) = -\delta_2\mu\Delta t + \sigma^2\Delta t\theta_2\delta_2^2 - r^* + \lambda_t(-\eta\delta_2 + \theta_2\delta^2\delta_2^2)e^{-\theta_2\eta\delta_2+0.5\theta_2^2\delta^2\delta_2^2} = 0.$$

Finally, the new probability distribution $\wp_{\theta_1}$ sampled by the first sub simulation can be derived by Equation (6) as follows:

$$d\wp_{\theta_1} = d\wp\exp(\theta_1 f_1(r_t) - \Psi_1(\theta_1))$$
$$= \frac{1}{\sqrt{2\pi}}e^{-Z^2/2}\sum_{n=0}^{\infty}\left[\frac{e^{-\lambda_t}\lambda_t^n}{n!}\left(\frac{1}{\sqrt{2\pi\delta^2}}\right)^n e^{-\frac{\sum_{k=1}^{n}(Z_K-\eta)^2}{2\delta^2}}\right]\exp\left(\theta_1 f_1(r_t) - \Psi_1(\theta_1)\right)$$
$$= \frac{1}{\sqrt{2\pi}}e^{-\frac{\left(Z+\theta_1\delta_1\sigma\sqrt{\Delta t}\right)^2}{2}}\sum_{n=0}^{\infty}\left[\frac{e^{-\lambda_t e^{-\theta_1\delta_1\eta+0.5\theta_1^2\delta_1^2\delta^2}}\left(\lambda_t e^{-\theta_1\delta_1\eta+0.5\theta_1^2\delta_1^2\delta^2}\right)^n}{n!}\left(\frac{1}{\sqrt{2\pi\delta^2}}\right)^n e^{-\frac{\sum_{k=1}^{n}\left(Z_K-\left(\eta-\theta_1\delta_1\delta^2\right)\right)^2}{2\delta^2}}\right].$$

That is, the first sub simulation tilts the probability from $\wp$ to $\wp_{\theta_1}$, where the distributions of random variables defined in Equation (2) are changed as follows:

$$Z \sim N\left(-\theta_1\delta_1\sigma\sqrt{\Delta t},1\right)$$

$N_{\Delta t} \sim Poisson\left(\lambda_t e^{-\theta_1\delta_1\eta+0.5\theta_1^2\delta_1^2\delta^2}\right)$ and $Z_i \sim N\left(\eta-\theta_1\delta_1\delta^2,\delta^2\right)$.

Note that

$$E\left[1_{A_1}\right] = E_{\wp_{\theta_1}}\left[1A_1\exp\left(-\theta_1 f_1(r_t) + \Psi_1(\theta_1)\right)\right].$$

Thus $E\left[1_{A_1}\right]$ is estimated by sampling the unbiased estimator $1_{A_1}\exp\left(-\theta_1 f_1(r_t) + \Psi_1(\theta_1)\right)$ from $\wp_{\theta_1}$ in the first sub simulation. Similarly, the probability distribution $\wp_{\theta_2}$ used by the second sub simulation can be derived as

$$d\wp_{\theta_2} = d\wp\exp\left(\theta_2 f_2(r_t) - \Psi_2(\theta_2)\right)$$
$$= \frac{1}{\sqrt{2\pi}}e^{-\frac{\left(Z+\theta_2\delta_2\sigma\sqrt{\Delta t}\right)^2}{2}}\sum_{n=0}^{\infty}\left[\frac{e^{-\lambda_t e^{-\theta_2\delta_2\eta+0.5\theta_2^2\delta_2^2\delta^2}}\left(\lambda_t e^{-\theta_2\delta_2\eta+0.5\theta_2^2\delta_2^2\delta^2}\right)}{n!}\left(\frac{1}{\sqrt{2\pi\delta^2}}\right)^n e^{-\frac{\sum_{K=1}^{n}\left(Z_K-\left(\eta-\theta_2\delta_2\delta^2\right)\right)^2}{2\delta^2}}\right].$$

Note also that

$$E\left[1_{A_2}\right] = E_{\wp_{\theta_2}}\left[1_{A_2}\exp\left(-\theta_2 f_2(r_t)+\Psi_2(\theta_2)\right)\right].$$

The second sub simulation estimates $E\left[1_{A_2}\right]$ by sampling the unbiased estimator $1_{A_2}\exp\left(-\theta_2 f_2(r_t)+\Psi_2(\theta_2)\right)$ from $\wp_{\theta_2}$.

3.3 Allocation of Computational Resources to Each Sub Simulation

Finally, we try to minimize the upper bound of the second moment for estimating *p* given a constraint on computational resources. The number of stock return samples serves as a proxy of computational resources. Assume that we can only sample N stock returns, and the numbers of stock returns sampled in the first and the second simulations are n_1 and n_2, respectively. By Equation (7), the upper bounds of the second moment of the estimator $1_{A_1}\exp\left(-\theta_1 f_1(r_t)+\Psi_1(\theta_1)\right)$ and $1_{A_2}\exp\left(-\theta_2 f_2(r_t)+\Psi_2(\theta_2)\right)$ under the probability measure $\wp_{\theta_1}$ and $\wp_{\theta_2}$ are $\exp\left(2\Psi_1(\theta_1)\right)$ and $\exp\left(2\Psi_2(\theta_2)\right)$, respectively. The second moment for estimating *p* is then

$$\frac{\exp\left(2\Psi_1(\theta_1)\right)}{n_2}+\frac{2\exp\left(\Psi_2(\theta_2)\right)}{n_2} \qquad (14)$$

To minimize Equation (14) under the constraint $n_1+n_2=\mathrm{N}$, n_1 and n_2 can be solved by Lagrange multiplier as follows:

$$n_1 = \frac{\sqrt{\exp\left(2\Psi_1(\theta_1)\right)}}{\sqrt{\exp\left(2\Psi_1(\theta_1)\right)}+\sqrt{\exp\left(2\Psi_2(\theta_2)\right)}}\mathrm{N} \qquad (15)$$

$$n_2 = \frac{\sqrt{\exp\left(2\Psi_2(\theta_2)\right)}}{\sqrt{\exp\left(2\Psi_1(\theta_1)\right)}+\sqrt{\exp\left(2\Psi_2(\theta_2)\right)}}\mathrm{N} \qquad (16)$$

4. Numerical Results

Table 1 illustrates how the probability tilting mechanism proposed in this paper greatly improves the performance of the Monte Carlo simulation. Consider a portfolio which is composed of a stock. The probability that the portfolio loses more than 5% in 0.008 year is estimated with three different approaches: Original denotes the naive Monte Carlo simulation that samples the stock return from Equation (2). Lognormal denotes Glasserman *et al.* importance sampling method under the Black-Scholes lognormal diffusion assumption (see subsection 2.2). Jump diffusion denotes the importance sampling method derived in Equation (13). We do 100 estimations

Table 1. Estimating the huge loss probability under different probability measures

Probability Measure	Original	Lognormal	Jump Diffusion
$\hat{p}$	0.0336	0.0339	0.0338
Var($\hat{p}$)	2.49×10^{-6}	1.21×10^{-6}	3.69×10^{-7}

The stock price is assumed to follow Merton's jump diffusion process: The stock average annual return μ is 0.05, the annual volatility of the stock price σ is 0.3, the time span Δt is 0.008 year, the jump frequency λ is 6, the average jump size η is 0, and the standard derivation of jump size δ is 0.03. $\hat{p}$ and $Var\left(\hat{p}\right)$ denote the estimated probability and the variance, respectively.

for each Monte Carlo simulation method and each estimation samples 10000 stock returns. The probability for the portfolio to lose more than 5% is about 3.38%. Obviously, the method proposed in this paper reduces the variance at a ratio of $\frac{3.69\times10^{-7}}{2.49\times10^{-6}}\approx 1/7$, which is better than the Glasserman's method$\left(\frac{1.21\times10^{-6}}{2.49\times10^{-6}}\approx 1/2\right)$.

Now we proceed to verify the superiority of the hybrid importance sampling algorithm in Table 2 and 3, where the stock price processes are assumed to follow the lognormal diffusion process and the Merton's jump diffusion process, respectively. The first column in these two tables denotes the probability measure of the stock return sampled by each Monte Carlo simulation method, where $\wp$ denotes the original probability measure defined in Equation (2), $\wp_{\theta_1}$ denotes the probability measure defined in Equation (12), and $\wp_{\theta_2}$ denotes the probability measure defined in Equation (13). Hybrid denotes the hybrid importance sampling algorithm that is composed of two sub simulations, which sample stock returns from $\wp_{\theta_1}$ and $\wp_{\theta_2}$, respectively. The numbers of samples allocated to these two sub simulations are determined in Equation (15) and (16), respectively. The second column $\hat{p}$ denotes the estimated probability for each Monte Carlo simulation, and the third column $Var\left(\hat{p}\right)$ denotes the variance of the estimated probability for each Monte Carlo simulation.

We first focus on Table 2. The event that the portfolio loss exceeds ℓ is about 0.0349. This event can be decomposed into two mutually exclusive events A_1 and A_2 (see Equation (11)). The event A_1 (with probability $P(A_1)\approx 0.0049$) is less likely to happen than the event A_2 (with probability $P(A_2)\approx 0.0299$). Although tilting the probability measure of the stock return from $\wp$ to $\wp_{\theta_1}$ makes the Monte Carlo simulation estimate $P(A_1)$ more efficiently and accurately, it also damages the accuracy

for estimating $P(A_2)$. It can be observed that this tilting produce inaccurate probability estimation (0.0139) with large variance. Similar problem applies to the Monte Carlo simulation that tilts the probability measure to $\wp_{\theta_2}$; this Monte Carlo simulation is inadequate to estimate $P(A_1)$. But tilting the probability measure to $\wp_{\theta_2}$ is better than tilting the probability to $\wp_{\theta_1}$ since the event A_2 constitutes the bulk of the event that the portfolio loss exceeds ℓ. Note that both important sampling methods mentioned above are less efficient than the primitive Monte Carlo simulation (that samples the stock return form $\wp$). On the other hand, the hybrid importance sampling algorithm performs better than the primitive Monte Carlo simulation. It produces accurate probability estimation and reduces the variance at a ratio of $1/15\left(\approx \frac{2.28\times10^{-7}}{3.55\times10^{-6}}\right)$. In other words, the sample size of the primitive Monte Carlo simulation should be 15 times the sample size of the hybrid importance sampling algorithm to make the former method achieve the same accuracy level as the latter method.

In Table 3, the probability that the portfolio loss exceeds ℓ is larger (about 0.0403) since the jumps in stock price make the huge loss events A_1 and A_2 more likely to happen. Naively tilting the probability measure of the stock return to P_{θ_1} (or P_{θ_2}) also performs poorly in this case. Note that the hybrid importance sampling algorithm still outperforms the primitive Monte Carlo simulation by reducing the variance at a ratio of $1/7.5\left(\approx \frac{5.44\times10^{-7}}{4.05\times10^{-6}}\right)$.

Table 2. Compare monte carlo simulations under the log-normal stock price model

Probability Measure	$\wp$	$\wp_{\theta_1}$	$\wp_{\theta_2}$	Hybrid
$\hat{p}$	0.0349	0.0139	0.0395	0.0349
Var($\hat{p}$)	3.55×10^{-6}	4.14×10^{-3}	2.52×10^{-3}	2.28×10^{-7}

Consider a short straddle option that contains a short position in 1 unit call option $(\delta_1 = -1)$ and 1 unit put option $(\delta_2 = -1)$. The probability for the portfolio to lose more than 5% of the stock price in 0.008 year is estimated in column 2 (i.e., $r^* = 0.05$ and $\ell \equiv -0.05S_t$). The stock average annual return μ is 0.05, the annual volatility of the stock price σ is 0.3, and r_0^* defined in Equation (9) is 0.01. Note that the jump frequency λ is 0 in this case.

Table 3. Compare monte carlo simulations under the merton's jump diffusion stock price model

Probability Measure	$\wp$	$\wp_{\theta_1}$	$\wp_{\theta_2}$	Hybrid
$\hat{p}$	0.0405	0.0363	0.0426	0.0403
Var($\hat{p}$)	4.05×10^{-6}	$1.97\times10^{-2}s$	6.21×10^{-4}	5.44×10^{-7}

The numerical settings follow the settings listed in Table 2 except that the jump frequency λ is 6, the mean of jump size η is 0, and the standard derivation of jump size δ is 0.03.

5 Conclusions

The paper improves the performance for estimating VaR. We first extend Glasserman's importance sampling method to Merton's jump diffusion process. Then we suggest a novel Monte Carlo simulation, the hybrid importance sampling algorithm, which can efficiently estimate the VaR of complex financial portfolios. Numerical results given in this paper verify that our method greatly improve the performance.

6. Acknowledgement

We thank Shi-Gra Lin, and Ren-Her Wang for useful suggestions.

REFERENCES

[1] S. K. Lin, C. D. Fuh, and T. J. Ko, "A bootstrap method with importance resampling to evaluate value-at-risk," J. Financial Studies, Vol. 12, pp. 81–116, 2004.

[2] H. G. Fong and K. C. Lin, "A new analytical approach to value at risk," J. Portfolio Management, Vol. 25, pp. 88–97, 1999.

[3] P. Glasserman, P. Heidelberger, and P. Shahabuddin, Vaniance Reduction Technique for Estimating Value-at-Risk, Management Sci., Vol. 46, pp. 1349–1364, 2000.

[4] E. Eberlein, U. Keller, and K. Prause, "New insights into smile, mispricing, and value-at-risk: The hyperbolic model, J. Business, Vol. 71, pp. 371–406, 1998.

[5] J. R. M. Hosking, G. Bonti, and D. Siegel, "Beyond the Lognormal," Risk, Vol. 13, pp. 59–62, 2000.

[6] K. Koedijk, R. Huisman, and R. Pownall, "VaR-x: Fat tails in financial risk management," J. Risk 1, pp. 47–62, 1998.

[7] P. Glasserman, P. Heidelberger, and P. Shahabuddin, "Portfolio value-at-risk with heavy-tailed risk factors," Math. Finance, Vol. 12, pp. 239–269, 2002.

[8] R. C. Merton, "Option pricing when underlying stock returns are discontinuous," J. Financial Econ., Vol. 3, pp. 125–144, 1976.

[9] J.-C. Duan, "The GARCH option pricing model," Math. Finance, Vol. 5, pp. 13–32, 1995.

[10] S. Heston, "A closed-form folution for options with stochastic volatility with applications to bond and currency options," Rev. Financial Studies, Vol. 6, pp. 327–343, 1993.

[11] P. Jorion, Value at risk, McGraw-Hill, New York, 1997.

[12] C. Rouvnez, Going Greek with VaR, Risk 10 (1997) 57–65.

[13] F. Black and M. Scholes, "The pricing of options and corporate liabilities," J. Political Econ., Vol. 81, pp. 637–659, 1973.

Call for Papers

International Conference on Computational Intelligence and Software Engineering (CiSE)

计算智能与软件工程国际会议征文

December 10~12, 2010 Wuhan, China

http:// www.ciseng.org/2010

The 2nd International Conference on Computational Intelligence and Software Engineering (CiSE 2010) will be held on December 10~12, 2010 in Wuhan, China. This conference will cover issues in **computational intelligence, information security, multimedia and graphics technologies, and software engineering.** **The conference proceedings will be published by IEEE, and all papers accepted will be included in IEEE Xplore and indexed by EI Compendex.**

TOPICS

- Artificial Intelligence
- Automated Reasoning
- Autonomous Systems
- Cloud Computing
- Cluster Computing
- Cognitive Science
- Communication Networks and Protocols
- Computational Biology
- Computational Chemistry
- Computational Neuroscience
- Computational Physics
- Computer Graphics
- Data Modeling
- Database Mining
- Database Technology
- Evolvable Hardware
- Expert Systems
- Fuzzy Systems
- Geographical Information System
- Grid Computing
- Hardware Implementation
- Human-computer Interaction
- Hybrid Systems
- Image Processing
- Image Understanding
- Machine Learning
- Multi-Agent Systems
- Natural Neural Systems
- Neural Genetic Systems
- Neural-Fuzzy Systems
- Numerical Algorithms
- Operating Systems
- Pattern Recognition
- Programming Methodology
- Project Management
- Real Time Control
- Requirement Analysis
- Simulation and Modeling
- Software Engineering
- Software Maintenance
- Software Standards and Design
- Software Testing and Quality Control
- Symbolic Mathematics
- Technological Forecasting
- Visualization
- Web-based Services

IMPORTANT DATES

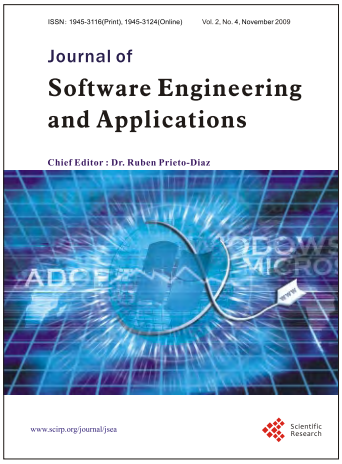
- Paper submission due: Jun. 21, 2010
- Acceptance notification: Aug. 10, 2010
- Conference: Dec. 10-12, 2010

CONTACT INFORMATION

Website:http://www.ciseng.org/2010

E-mail:info@ciseng.org

Call For Papers



Journal of Software Engineering and Applications (JSEA)

ISSN 1945-3116 (print) ISSN 1945-3124 (online)
www.scirp.org/journal/jsea

JSEA publishes four categories of original technical articles: papers, communications, reviews, and discussions. Papers are well-documented final reports of research projects. Communications are shorter and contain noteworthy items of technical interest or ideas required rapid publication. Reviews are synoptic papers on a subject of general interest, with ample literature references, and written for readers with widely varying background. Discussions on published reports, with author rebuttals, form the fourth category of JSEA publications.

Editor-in-Chief

Dr. Ruben Prieto-Diaz, Universidad Carlos III de Madrid, Spain

Subject Coverage

- Applications and Case Studies
- Artificial Intelligence Approaches to Software Engineering
- Automated Software Design and Synthesis
- Automated Software Specification
- Component-Based Software Engineering
- Computer-Supported Cooperative Work
- Software Design Methods
- Human-Computer Interaction
- Internet and Information Systems Development
- Knowledge Acquisition
- Multimedia and Hypermedia in Software Engineering
- Object-Oriented Technology
- Patterns and Frameworks
- Process and Workflow Management
- Programming Languages and Software Engineering
- Program Understanding Issues
- Reflection and Metadata Approaches
- Reliability and Fault Tolerance
- Requirements Engineering
- Reverse Engineering
- Security and Privacy
- Software Architecture
- Software Domain Modeling and Meta-Modeling
- Software Engineering Decision Support
- Software Maintenance and Evolution
- Software Process Modeling
- Software Reuse
- Software Testing
- System Applications and Experience
- Tutoring, Help and Documentation Systems

Notes for Prospective Authors

Submitted papers should not have been previously published nor be currently under consideration for publication elsewhere. All papers are refereed through a peer review process. For more details about the submissions, please access the website.

Website and E-Mail

Website: http://www.scirp.org/journal/jsea E-Mail: jsea@scirp.org

J. Software Engineering and Applications, **2009, 2**
Published Online November 2009 in SciRes (http://www.SciRP.org/journal/jsea).

TABLE OF CONTENTS

Volume 2, Number 4 **November 2009**

9771945311006 05