December 2006

Analysis of Software Quality via a Goal Programming Approach

Young Chun
Louisiana State University

Analysis of Software Quality via a Goal Programming Approach

Young H. Chun
Department of Information Systems and Decision Sciences
E. J. Ourso College of Business
Louisiana State University
Baton Rouge, LA 70803, USA
chun@lsu.edu

ABSTRACT

Complex software systems often fail because of errors undiscovered in the design stage of the development process. Detecting these errors early in the process would eliminate many downstream problems. The so-called capture-recapture model used by biologists in estimating the size of wildlife populations has also been used to estimate the number of software design errors. However, one simplifying assumption in capture-recapture models is that the inspections performed by various inspectors are statistically independent from each other. In the paper, we propose a novel method that is based on the correlation matrix of multiple inspectors. In a numerical analysis, we show that our method outperforms other traditional models that are based on the independence assumption.

Keywords

Software reliability, software inspections, capture-recapture model, quality management.

INTRODUCTION

Sometimes complex software systems fail because of errors or faults introduced in the design stage of the development process. Design reviews can remove some of these errors, but often a few errors remain undetected until the software is developed further. It is important to eliminate these design errors at an early stage because they become much more expensive to fix as a software system proceeds through development.

In the design stage, software engineers usually produce documents that are reviewed by their peers. Document reviews serve as the gateway to the next stage in the process. The current industry practice is for each document to be reviewed by a small group of software engineers before it passes to the next stage. The engineers individually read the document and note “issues” that they believe should be resolved before the software feature is developed further. We refer to these issues as “errors” or “fault” in the paper. At the review meeting, data are collected showing which software engineers discovered which faults. These data are then used to estimate the number of errors that remain undetected. These estimates can then be used in managing the software quality at the design stage of the development process.

Since it is impossible to count the total number of errors in a software system before it has been in operation for a while, it is necessary to build estimation models of the number of defects in a software artifact. One of the techniques that have been widely used to estimate the software design faults is a class of statistical methods known as “capture-recapture models”.

This technique is most commonly used by biologists in estimating the size of wildlife populations in wildlife research.

LITERATURE REVIEW

Capture-recapture models with two inspectors

Suppose that animals are captured, marked, and released on several trapping occasions. If an animal bearing a mark is captured on a subsequent trapping occasion, it is said to be recaptured. Based on the number of marked animals that are recaptured, one can estimate the total population size using statistical models and their estimators. When many marked animals are recaptured, for example, one can argue that the total population size is small, and vice versa.

		Second Catch		
		Marked	Unmarked	
First Catch	Marked	n_{12}	n_1-n_{12}	n_1
	Unmarked	n_2-n_{12}	N-n_1-n_2+n_{12}	N-n_1
		n_2	N-n_2	N

Table 1. Capture-recapture history of fish in the lake

For example, consider the problem of estimating the number of fish in a lake (Schnabel, 1938). A commonly used approach is to first catch a number n_1 of fish, tag or mark them, and release them again into the lake. Some time later, after giving the tagged fish an opportunity to mix well with the remaining fish in the lake, a second catch of n_2 fish is made, and the number n_{12} of tagged fish is observed. What is the total number N of fish in the lake?

From Table 1, one can derive an intuitive estimator of the population size N based on the assumption that the ratio of marked to total animals in the second sample (n_{12}/n_2) should reflect the same ratio (n_1/N) in the population. Thus, the so-called Lincoln-Peterson estimator of N can be derived as follows (Seber, 1982):

$$\hat{N} = \frac{n_1 \times n_2}{n_{12}}. \tag{1}$$

From a formal statistical point of view, the Lincoln-Peterson estimator has the drawback of having an "infinite bias", because there is always a finite probability that $n_{12} = 0$ (i.e., no marked animals are caught on the second occasion). Chapman (1951) proposed an adjusted equation to circumvent this drawback:

$$\hat{N} = \frac{(n_1 + 1) \times (n_2 + 1)}{(n_{12} + 1)} - 1. \tag{2}$$

Now the idea behind using capture-recapture models for software engineering inspections can be illustrated by the following puzzle (Wells, 1992, p. 146):

> "Two proof readers are checking two copies of the same manuscript. The first finds thirty errors, and the second finds only twenty-four. When their completed proofs are compared, it turns out that only twenty errors have been spotted by both of them. How many errors would you suspect remain, not detected by either of them?"

Applying the Lincoln-Peterson estimator to the data given in the problem, we see that the estimate of N is 36, and thus answer to the puzzle is 2.0 (=N-n_1-n_2+n_{12}). Note that the Chapman's modification of the Petersen-Lincoln estimator in (2) gives $\hat{N}$ =35.90.

Readers are referred to Pollock (1991) and Seber (1986) for more detail about the modeling of capture, recapture, and removal statistics in the estimation of demographic parameters of fish and wildlife populations, and to Briand, El Emam, Freimut, and Laitenberger (2000) for a comprehensive analysis of capture-recapture models in the field of software inspections.

Capture-recapture models with multiple inspectors

The basic methodology of capture-recapture models can be easily extended to the case in which there are more than two inspectors. Suppose that there are m inspectors, who work independently on identical copies of the product, or who work in some sequence on a single product, identifying, but not removing, the errors which they find. Suppose also that the inspection process is such that (1) each error present has the same probability of being detected by a given inspector, and (2) the probability that inspector i will find any given error is p_i, independent of previous errors found by inspector i or by any other inspector.

Let n_i denote the total number of errors detected by inspector i (i = 1, 2,…, m), and u_i be the number of errors not detected by inspector i but detected by any of the remaining inspectors. Likewise, let n_{ij} denote the number of errors detected by both inspector i and inspector j, and u_{ij} denote the number of errors detected by neither inspector i nor inspector j but detected by any of the remaining inspectors. Furthermore, let n_T be the total number of errors discovered by any of the inspectors and n_0 be the number of errors not discovered by any inspectors. Thus, the total number of errors N in the software design document is $n_T + n_0$ as shown in Table 2.

		Inspector j		
		Detected	Undetected	
Inspector i	Detected	n_{ij}	$n_i - n_{ij}$	n_i
	Undetected	$n_j - n_{ij}$	$u_{ij} + n_0$	$u_i + n_0$
		n_j	$u_j + n_0$	$N = n_T + n_0$

Table 2. Capture-recapture history in a multiple inspector case

The most commonly used estimator of N in the software design document is the maximum likelihood (ML) estimator $\hat{N}$, which maximizes the following likelihood (Eick *et al.*, 1993a):

$$L(N) = \binom{N}{n_T} \prod_{i=1}^{m} p_i^{n_i} (1 - p_i)^{N - n_i} . \tag{3}$$

The exact ML estimator of N does not exist in closed form. The relevant log-likelihood function for estimating the parameters $(N, p_1, p_2, \ldots, p_m)$ is given by

$$\ln L(N) = \ln \binom{N}{n_T} + \sum_{i=1}^{m} n_i \ln p_i + \sum_{i=1}^{m} (N - n_i) \ln(1 - p_i) . \tag{4}$$

Given the value of N, the ML estimators $p_i(N)$ are obtained by taking the first-order derivative of (4) with respect to p_i, setting it equal to zero, and solving it for p_i:

$$\hat{p}_i = \frac{n_i}{N}, \qquad i = 1, 2, \ldots, m. \tag{5}$$

By replacing p_i in (4) with $\hat{p}_i$ in (5), we have the following the log-likelihood function:

$$\ln L(N) = \ln \binom{N}{n_T} + \sum_{i=1}^{m} n_i \ln \left(\frac{n_i}{N} \right) + \sum_{i=1}^{m} (N - n_i) \ln \left(\frac{N - n_i}{N} \right)$$

$$= \ln N! - \ln(N - n_T)! + \sum_{i=1}^{m} (N - n_i) \ln(N - n_i) - Nm \ln N + \text{constant} . \tag{6}$$

The ML estimator $\hat{N}$ that maximizes (6) can be obtained numerically with the constraint $N \geq n_T$. For a large value of N, it is computationally more convenient to use the log-gamma function ln [$\Gamma(N+1)$] in place of ln $N!$ in (6), assuming that N is a continuous variable; In the numerical analysis, we used the log-gamma function, GAMMALN( ), in Microsoft Excel 2000.

An alternative estimator of N is Darroch's approximate ML estimator $\hat{N}$, which is the unique solution of the following equation (Darroch, 1958):

$$1 - \frac{n_T}{N} = \prod_{i=1}^{m} \left(1 - \frac{n_i}{N} \right), \qquad \text{where } N \geq n_T. \tag{7}$$

The left side of (7) estimates the probability of not being able to detect an error using the fact that the group of inspectors as a whole discovered a total number of n_T errors. The right hand side of (7) estimates the same probability using the number of errors n_i detected by each individual inspector. It can be easily shown that, for the special case of m=2, Darroch's estimator in (7) reduces to the Lincoln-Peterson estimator in (1).

Both the maximum likelihood estimator and the Darroch's approximation method require a fast and iterative method, such as the Newton-Rapson method or the steepest-descent method, to find the optimal solution. However, Chao (1989) proposed an estimator of N that is in closed form as follows:

$$\hat{N} = n_T + \frac{\sum_{i=1}^{m}\sum_{j=i+1}^{m} n_{\{i\}} n_{\{j\}}}{s_2 + 1}, \tag{8}$$

where $n_{\{i\}}$ is the number of defects detected only by inspector i and s_2 is the total number of defects that have been detected by exactly two inspectors. For the special case of m=2, Chao's estimator in (8) reduces to the Chapman's modification of the Petersen-Lincoln estimator in (2).

Like all other mathematical models, capture-recapture models in software inspection as well as in biology make certain simplifying assumptions so as to render the mathematics tractable. One simplifying assumption is that the inspectors are working *independently* from each other. The statistical independence between two inspectors can be formally explained as follow.

For a certain error, let the random variable X_i be defined as follows:

$$X_i = \begin{cases} 1 & \text{if detected by inspector } i \\ 0 & \text{if undetected by inspector } i \end{cases} \tag{9}$$

Furthermore, let X_i be a Bernoulli random variable with parameter p_i as shown in Table 3. If and only if two inspectors are statistically *independent*, the joint probability p_{12} that the error is detected by both inspectors is expressed as $p_{12} = p_1\, p_2$. If two inspectors are working together to find many errors in common, then $p_{12} > p_1\, p_2$. If they have different specialties or educational backgrounds, then $p_{12} < p_1\, p_{2}$.

		Inspector 2		
		$X_2 = 1$	$X_2 = 0$	
Inspector 1	$X_1 = 1$	p_{12}	$p_1 - p_{12}$	p_1
	$X_1 = 0$	$p_2 - p_{12}$	$1- p_1 - p_2+p_{12}$	$1-p_1$
		p_2	$1-p_2$	1.0

Table 3. Probability table for two inspectors

Suppose that the data in Table 1 is observed from N Bernoulli trials. Then, the maximum likelihood estimator of p_{12}, p_1, p_2 are n_{12}/N, n_1/N, and n_2/N, respectively. If two inspectors are statistically independent, then we have $p_{12} = p_1\, p_2$ or

$$\frac{n_{12}}{N} = \frac{n_1}{N}\frac{n_2}{N}. \tag{10}$$

From (10), the Lincoln-Peterson estimator in (1) can be easily derived and the number of undetected errors n_0 is estimated as follows:

$$\hat{n}_0 = \frac{(n_1 - n_{12})(n_2 - n_{12})}{n_{12}}. \tag{11}$$

In many occasions, however, this independence assumption might be violated particularly when (a) inspectors collaborate (resulting is more defects found in common), or (b) specialize on certain defect types (resulting in fewer defects found in common). Collusion occurs when multiple inspectors work together, resulting in more common faults than would be expected. On the other hand, inspectors specializing in different areas may be intentionally chosen to ensure nothing is overlooked. Since each inspector may focus on certain parts of the document relating to his or her area of expertise, specialization may result in fewer common faults than would otherwise be expected. In either case, the cornerstone of capture-recapture models is no longer valid.

In the paper, we propose a novel approach that is distinct from the concept of capture-recapture models. The idea is to consider the correlation matrix of multiple inspectors. Since the correlation coefficients are expressed as a function of N, we can estimate N if the correlation matrix of multiple inspectors is given *a priori* or can be estimated based on past inspection history.

CORRELATION MATRIX OF MULTIPLE INSPECTORS

The correlation coefficient is a standardized measure of the relationship between two variables X_1 and X_2, which is defined as follows:

$$\rho_{1,2} = \frac{Cov(X_1, X_2)}{\sqrt{Var(X_1)\,Var(X_2)}}. \tag{12}$$

As shown above, the correlation coefficient is computed by dividing the covariance by the standard deviations of the two variables, X_1 and X_2. This division results in the correlation coefficient having a standardized range from -1 to +1. The correlation coefficient provides us with a good indication of relationship because it varies in the range from -1 to +1 and its value directly indicates (1) the strength and (2) the direction of the relationship between two variables.

For the binary random variables X_1 and X_2 in Table 3, the covariance between X_1 and X_2 is shown to be

$$Cov(X_1, X_2) = E[X_1X_2] - E[X_1]E[X_2] \ = p_{12} - p_1p_2, \tag{13}$$

and thus the correlation coefficient between the two binary variables is

$$\rho_{1,2} = \frac{p_{12} - p_1p_2}{\sqrt{p_1(1-p_1)\,p_2\,(1-p_2)}}. \tag{14}$$

From the contingency table in Table 1, the observed correlation coefficient is

$$r_{1,2}(N) = \frac{\frac{n_{12}}{N} - \frac{n_1}{N}\frac{n_2}{N}}{\sqrt{\frac{n_1}{N}\frac{(N-n_1)}{N}\frac{n_2}{N}\frac{(N-n_2)}{N}}} = \frac{n_{12}N - n_1n_2}{\sqrt{n_1(N-n_1)n_2(N-n_2)}}, \tag{15}$$

which is also expressed as a function of n_0 as follows:

$$r_{1,2}(n_0) = \frac{n_{12}n_0 - u_1u_2}{\sqrt{n_1(u_1+n_0)n_2(u_2+n_0)}}. \tag{16}$$

Since n_0 and $r_{1,2}$ in (16) are in one-to-one correspondence, we can estimate the total number of undetected errors n_0 for any given value of the correlation coefficient $\hat{\rho}_{1,2}$. In other words, there exists a *unique* solution because $r_{1,2}(n_0)$ is a *monotone* function of n_0. If the inspections are statistically independent as in capture-recapture models, for example, the correlation coefficient in (16) is assumed to be zero and the number of undetected errors is estimated as

$$\hat{n}_0 = \frac{u_1u_2}{n_{12}}, \tag{17}$$

which is consistent with (11). Thus, the Lincoln-Peterson estimator is a special case of (16) in which the correlation coefficient is zero.

Suppose that there are m inspectors. From Table 2, the observed correlation coefficient between inspector i and inspector j is obtained as follows:

$$r_{ij}(n_0) = \frac{n_{ij}(u_{ij}+n_0) - (n_i - n_{ij})(n_j - n_{ij})}{\sqrt{n_i(u_i+n_0)n_j(u_j+n_0)}}, \tag{18}$$

which is a function of n_0. For the $m \times m$ correlation matrix, there are $m(m\text{-}1)/2$ number of elements $r_{ij}(n_0)$, all of which will be changed simultaneously as we change the value of n_0.

Suppose that all of ρ_{ij} in the correlation matrix are known *a priori* or can be estimated from the past inspection history. Then, there are "deviations" between the given correlation coefficients $\hat{\rho}_{ij}$ and the observed correlation coefficients $r_{ij}(n_0)$. The given correlation coefficients ρ_{ij} are considered multiple "goals" that should be achieved by changing n_0, and thus the problem of estimating the number of undetected errors n_0 can be formulated as a "goal programming" model in operations research. In the next section, we estimate the number of undetected errors via the goal programming approach.

GOAL PROGRAMMING APPROACH

For m inspectors, there are $m(m\text{-}1)/2$ number of goals $\hat{\rho}_{ij}$ to be achieved and the same number of attained values $r_{ij}(n_0)$

obtained from the inspection data. Note that $r_{ij}(n_0)$ are all functions of a single variable n_0. As we change n_0, all $r_{ij}(n_0)$ are changed simultaneously. Thus, each of the attained value $r_{ij}(n_0)$ may be lower or higher than a given goal $\hat{\rho}_{ij}$. Let $d_{ij} = \hat{\rho}_{ij} - r_{ij}(n_0)$ be a "deviational variable", which is the difference between the goal and the attained value. Using the deviational variable, we can formulate a goal program that minimizes the total cost incurred in deviating from the goals.

As in many other goal programming models, we use the *additive* objective function for its simplicity: the total cost is the weighted sum of individual costs as follows:

$$\min_{\{n_0\}} z = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} v_{ij}\ c(d_{ij}) \tag{19}$$

subject to

$$\hat{\rho}_{ij} - r_{ij}(n_0) = d_{ij}\,, \quad i = 1, 2, \ldots, m\text{-}1 \text{ and } j = 2, 3, \ldots, m \tag{20}$$

$$-\infty < d_{ij} < +\infty\,;\ n_0 \geq 0\,,$$

where $c(d_{ij})$ is a cost function and v_{ij} is its weight.

The weight v_{ij} should be determined such that the most important goal has the largest weight, and so on. In practice, we suggest that the weight v_{ij} be determined based on n_i, n_j, and n_{ij}; e.g.,

$$v_{ij} = n_i + n_j - n_{ij}\,. \tag{21}$$

For the cost function $c(d_{ij})$, we could use the one that minimizes the weighted sum of squared deviations,

$$c(d_{ij}) = d_{ij}^2\,, \tag{22}$$

or that minimizes the weighted sum of absolute deviations

$$c(d_{ij}) = \left|d_{ij}\right|\,. \tag{23}$$

In practice, the goals or targets, $\hat{\rho}_{ij}$, may not be known a priori. In that case, we recommend the following heuristic method. The attained correlation coefficient $r_{ij}(n_0)$ in (18) is a monotonically increasing function of n_0. Thus, the lower limit r_{ij}^{mim} is obtained when $n_0 = 0$,

$$r_{ij}^{\min} = r_{ij}(n_0 = 0) = \frac{n_{ij}u_{ij} - (n_i - n_{ij})(n_j - n_{ij})}{\sqrt{n_i u_i n_j u_j}}\,, \tag{24}$$

and the upper limit on r_{ij} is achieved when $n_0 \to \infty$,

$$r_{ij}^{\max} = \lim_{n_0\to\infty} r_{ij}(n_0) = \lim_{n_0\to\infty} \frac{n_{ij}(u_{ij} + n_0) - (n_i - n_{ij})(n_j - n_{ij})}{\sqrt{n_i(u_i + n_0)n_j(u_j + n_0)}}$$

$$= \lim_{n_0\to\infty} \frac{n_{ij}(u_{ij}/n_0 + 1) - (n_i - n_{ij})(n_j - n_{ij})/n_0}{\sqrt{n_i(u_i/n_0 + 1)n_j(u_j/n_0 + 1)}} = \frac{n_{ij}}{\sqrt{n_i n_j}}\,. \tag{25}$$

Since the unknown goal $\hat{\rho}_{ij}$ is somewhere between $r_{ij}^{\min}$ and $r_{ij}^{\max}$, we may set the goal or target $\hat{\rho}_{ij}$ as a weighted average of the lower bound $r_{ij}^{\min}$ and the upper bound $r_{ij}^{\max}$ as follows:

$$\hat{\rho}_{ij} = w\, r_{ij}^{\min} + (1 - w)\, r_{ij}^{\max}\,. \tag{26}$$

As a practical matter, we suggest that equal weight should be put on the lower and the upper bound because we lack any theory that suggests otherwise. In our empirical research, we also found that the weight w of 0.5 or 0.6 gives a good estimate of $\hat{\rho}_{ij}$.

PERFORMANCE EVALUATIONS

In our goal programming approach, the goal is a weighted average of the lower and the upper bounds of the observed correlation coefficients $r_{ij}(n_0)$. Thus, the weight w should be specified in our model. In addition, the form of the cost function $c(d_{ij})$ and its appropriate weight v_{ij} in the objective function should be also specified in advance. In the numerical analysis, we set w equal to 0.6, which is the most appropriate value that we found from our empirical research. The objective is to minimize the weighted average of the absolute deviations, in which the weight v_{ij} is determined from (21).

Suppose that there are exactly 100 (=N) errors in the system, among which 80 (= n_T) errors have been detected by m inspectors. For all the possible cases, we found the estimates of N using the SOLVER function in Microsoft Excel and then computed the mean square error,

$$MSE = \frac{\sum_{j=1}^{k} (\hat{N}_j - 100)^2}{k}, \tag{27}$$

and the mean absolute percentage error,

$$MAPE = \frac{\sum_{j=1}^{k} \frac{|\hat{N}_j - 100|}{100}}{k}, \tag{28}$$

where k is the number of possible cases considered in the numerical analysis.

First, we consider the situation in which there are two inspectors. As shown in Table 4, we consider 24 possible cases in which exactly 80 errors have been found by the two inspectors and 20 errors are still remaining in the system. In the first row in Table 9, for example, both Inspector 1 and Inspector 2 have detected 70, respectively. If their inspections are statistically independent, the number of common errors detected by both inspectors should be 49. Since they have detected 60 common errors, it implies that their inspections are positively interrelated with each other.

	n_1	n_2	n_{12}	MLE	Darroch (1958)	Chao (1989)	Chun
1	70	70	60	80.968	81.667	81.639	86.667
2	65	70	55	81.980	82.727	82.679	88.202
3	60	70	50	83.186	84.000	83.922	89.531
4	55	70	45	84.657	85.556	85.435	90.698
5	50	70	40	86.493	87.500	87.317	91.720
6	45	70	35	88.852	90.000	89.722	92.601
7	40	70	30	91.997	93.333	92.903	93.333
8	35	70	25	96.399	98.000	97.308	93.905
9	30	70	20	103.001	105.000	103.810	94.296
10	25	70	15	114.003	116.667	114.375	94.477
11	20	70	10	136.006	140.000	134.545	94.410
12	15	70	5	202.011	210.000	188.333	94.048
13	60	60	40	88.996	90.000	89.756	93.333
14	55	60	35	93.141	94.286	93.889	94.908
15	50	60	30	98.667	100.000	99.355	96.313
16	45	60	25	106.401	108.000	106.923	97.560
17	40	60	20	118.003	120.000	118.095	98.647
18	35	60	15	137.338	140.000	136.250	99.566
19	30	60	10	176.006	180.000	170.909	100.300
20	25	60	5	292.008	300.000	263.333	100.824
21	50	50	20	123.003	125.000	122.857	100.000
22	45	50	15	147.338	150.000	145.625	101.584
23	40	50	10	196.006	200.000	189.091	103.004
24	35	50	5	342.007	350.000	305.000	104.254
Mean squared error				5359.346	5817.843	4031.396	40.419
Mean absolute percentage error				42.408%	44.067%	38.134%	5.240%

Table 4. Numerical analysis for two inspectors: *N* = 100

In this case, the total number of errors is estimated to be 80.968, 81.667, and 81.639, respectively, by the maximum likelihood estimator, Darroch's approximate maximum likelihood estimator, and Chao's estimator. Note that Darroch's estimate and Chao's estimate are the same as the Lincoln-Peterson estimate and Chapman's estimate, respectively, when m = 2. Our estimate of N based on the goal programming approach is 86.667, which is slightly closer to the parameter N = 100. In terms of the MSE and MAPE criteria, our method performs much better than the traditional capture-recapture methods as shown in the last two rows in Table 9. Particularly when the inspectors are negatively interrelated as in case 24, our goal programming approach outperforms the traditional methods remarkably.

The performance of our method is reaffirmed in the three inspector cases. As shown in Table 5, we considered 42 possible cases of m=3, systematically changing the number of errors detected by each inspectors to simulate various situations. As shown in last two rows in Table 10, the MSE and the MAPE of our method are much smaller than those of the traditional capture-recapture models.

We did not perform the numerical analysis for the cases of more than three inspectors, because there are too many possible permutations. For m inspectors, for example, we need to simulate 2^m -1 different inspection history. However, we expect that our method would also outperform the traditional models that are based on the independence assumption.

	n_1	n_2	n_3	n_{12}	n_{13}	n_{23}	n_{123}	MLE	Darroch (1958)	Chao (1989)	Chun
1	65	65	65	55	55	55	50	81.081	80.583	84.688	90.000
2	50	65	65	40	40	55	35	81.824	81.250	82.419	94.439
3	35	65	65	25	25	55	20	82.673	82.026	81.630	97.593
4	65	50	65	40	55	40	35	81.824	81.250	82.419	94.439
5	50	50	65	25	40	40	20	83.475	82.791	81.630	94.439
6	35	50	65	10	25	40	5	85.577	84.756	81.230	97.593
7	65	35	65	25	55	25	20	82.673	82.026	81.630	97.593
8	50	35	65	10	40	25	5	85.577	84.756	81.230	97.593
9	65	65	50	55	40	40	35	81.824	81.250	82.419	94.439
10	50	65	50	40	25	40	20	83.475	82.791	81.630	94.439
11	35	65	50	25	10	40	5	85.577	84.756	81.230	97.593
12	65	50	50	40	40	25	20	83.475	82.791	81.630	94.439
13	50	50	50	25	25	25	5	87.370	86.498	81.230	100.000
14	65	35	50	25	40	10	5	85.577	84.756	81.230	97.593
15	65	65	35	55	25	25	20	82.673	82.026	81.630	97.593
16	50	65	35	40	10	25	5	85.577	84.756	81.230	97.593
17	65	50	35	40	25	10	5	85.577	84.756	81.230	97.593
18	50	50	65	40	40	40	35	83.475	82.791	94.063	94.439
19	35	50	65	25	25	40	20	85.577	84.756	87.258	97.593
20	20	50	65	10	10	40	5	88.408	87.385	84.891	98.942
21	50	35	65	25	40	25	20	85.577	84.756	87.258	97.593
22	35	35	65	10	25	25	5	89.765	88.674	84.891	97.593
23	50	20	65	10	40	10	5	88.408	87.385	84.891	98.942
24	50	50	50	40	25	25	20	87.370	86.498	87.258	100.000
25	35	50	50	25	10	25	5	92.921	91.752	84.891	104.334
26	50	35	50	25	25	10	5	92.921	91.752	84.891	104.334
27	50	50	35	40	10	10	5	92.921	91.752	84.891	104.334
28	35	35	65	25	25	25	20	89.765	88.674	103.438	97.593
29	20	35	65	10	10	25	5	96.484	94.887	92.097	98.942
30	35	20	65	10	25	10	5	96.484	94.887	92.097	98.942
31	35	35	50	25	10	10	5	105.185	103.302	92.097	104.334
32	35	50	50	25	25	25	20	92.921	91.752	117.500	100.000
33	20	50	50	10	10	25	5	101.746	100.000	99.355	100.000
34	35	35	50	10	25	10	5	105.185	103.302	99.355	104.334
35	35	50	35	25	10	10	5	105.185	103.302	99.355	104.334
36	20	35	50	10	10	10	5	130.927	126.974	140.938	104.334
37	50	50	35	25	25	25	20	92.921	91.751	117.500	100.000
38	35	50	35	10	10	25	5	105.185	103.302	99.355	104.334
39	50	35	35	10	25	10	5	105.185	103.302	99.355	104.334
40	50	50	20	25	10	10	5	101.746	100.000	99.355	100.000
41	35	35	35	10	10	10	5	138.546	134.223	155.000	110.000
42	35	50	20	10	10	10	5	130.927	126.973	140.938	104.334
Mean squared error								228.996	225.176	350.847	16.288
Mean absolute percentage error								13.049%	13.129%	15.414%	3.281%

Table 5. Numerical analysis for three inspectors: *N* = 100

CONCLUDING REMARKS

In the paper, we proposed a new method that estimates the number of undetected errors in software design documents. The idea is to use the correlation matrix of multiple inspectors and to formulate the estimation problem as a goal program. To use our method in practice, the goal or target ρ_{ij} must be given in advance or should be estimated based on the past data or experience. If the goal ρ_{ij} is not available, we may first find, using the observed data, its lower and upper bounds, and then estimate the goal as a weighted average of the two bounds. The optimal weight could be obtained from the training data, or we suggest w=0.5 or 0.6 if there is no prior information at all.

We also illustrated our method with an empirical data and compared its performance with those of traditional capture-recapture models that are based on the independence assumption. Although our method outperformed other capture-recapture models, it would be more informative to use our method along with other traditional models in a specific inspection environment.

The capture-recapture model was first introduced by ecologists as a means of estimating the size of wildlife populations. However, the basic methodology has been used in different scientific areas such as demography, engineering, and epidemiology. We expect that our method can be successfully applied to any areas in which we suspect the independence assumption among inspectors.

REFERENCES

1. Briand, L. C., El Eman, K., Freimut, B. G. , and Laitenberger, O. (2000) A comprehensive evaluation of capture-recapture models for estimating software defect content, *IEEE Transactions on Software Engineering*, 26, 6, 518-540.
2. Chao, A. (1989) Estimating population size for sparse data in capture-recapture experiments, *Biometrics*, 45, 427-438.
3. Chapman, D. G. (1951) Some properties of the hypergeometric distribution with applications to zoological sample censuses, *University of California Publications in Statistics,* 1, 131-160.
4. Darroch, J. N. (1958) The multiple-recapture census: I. Estimation of a closed population, *Biometrika*, 45, 343-359.
5. Eick, S. G., Loader, C. R. , Long, M. D., Votta, L. G., and Wiel, S. V. (1993a) Estimating software fault content before coding, *Proceedings of the 15th International Conference of Software Engineering*, Melbourne, Australia.
6. Eick, S. G., Loader, C. R., Vander Wiel, S. A., and Votta, L. G. (1993b) How many errors remain in a software design document after inspection? *Proceedings of the 25th Symposium on the Interface*, 195-202.
7. Pollock, K. H. (1991) Modeling capture, recapture, and removal statistics for estimation of demographic parameters for fish and wildlife populations: past, present, and future, *J. American Statistical Association,* 86, 225-238.
8. Schnabel, Z. E. (1938) The estimation of the total fish population of a lake, *American Mathematics Monthly,* 45, 348-352.
9. Seber, G. (1982) The estimation of animal abundance and related parameters, 2nd edition, Charles Griffin & Company.
10. Seber, G. (1986) A review of estimating animal abundance, *Biometrics*, 42, 267-292.
11. Wells, D. (1992) *The Penguin Book of Curious and Interesting Puzzles*, Penguin Books, New York.