

SOFTWARE ENGINEERING LABORATORY: DATA VALIDATION

Marvin V. Zelkowitz and Eric Chen
Department of Computer Science
University of Maryland
College Park, Maryland 20742

The need to validate the data being collected by the Software Engineering Laboratory is a primary prerequisite before analyses can be attempted. In terms of validation, three phases have been identified: (1) forms validation, (2) project validation, and (3) completeness and consistency validation.

Forms validation is a process that verifies that the data on the forms that are being collected is accurately transferred to the computerized data base. It is mostly a clerical process as the forms are typed into the computer. Minimal checking of data across forms is attempted – all checking is at the local level. In addition, once a project's forms has been entered, the data is rechecked against the original forms before being used in analysis.

Project validation tests whether the entire set of forms for a project is consistent. For example, does the number of hours specified on the resource summary (filled out by the project manager weekly for all project personnel) agree with the number of hours specified by each programmer on the component status report (giving the hours spent each week on each component)? What date is missing (e.g., which reports are not in the data base)? This is a relatively straightforward check on the total collected data from a project.

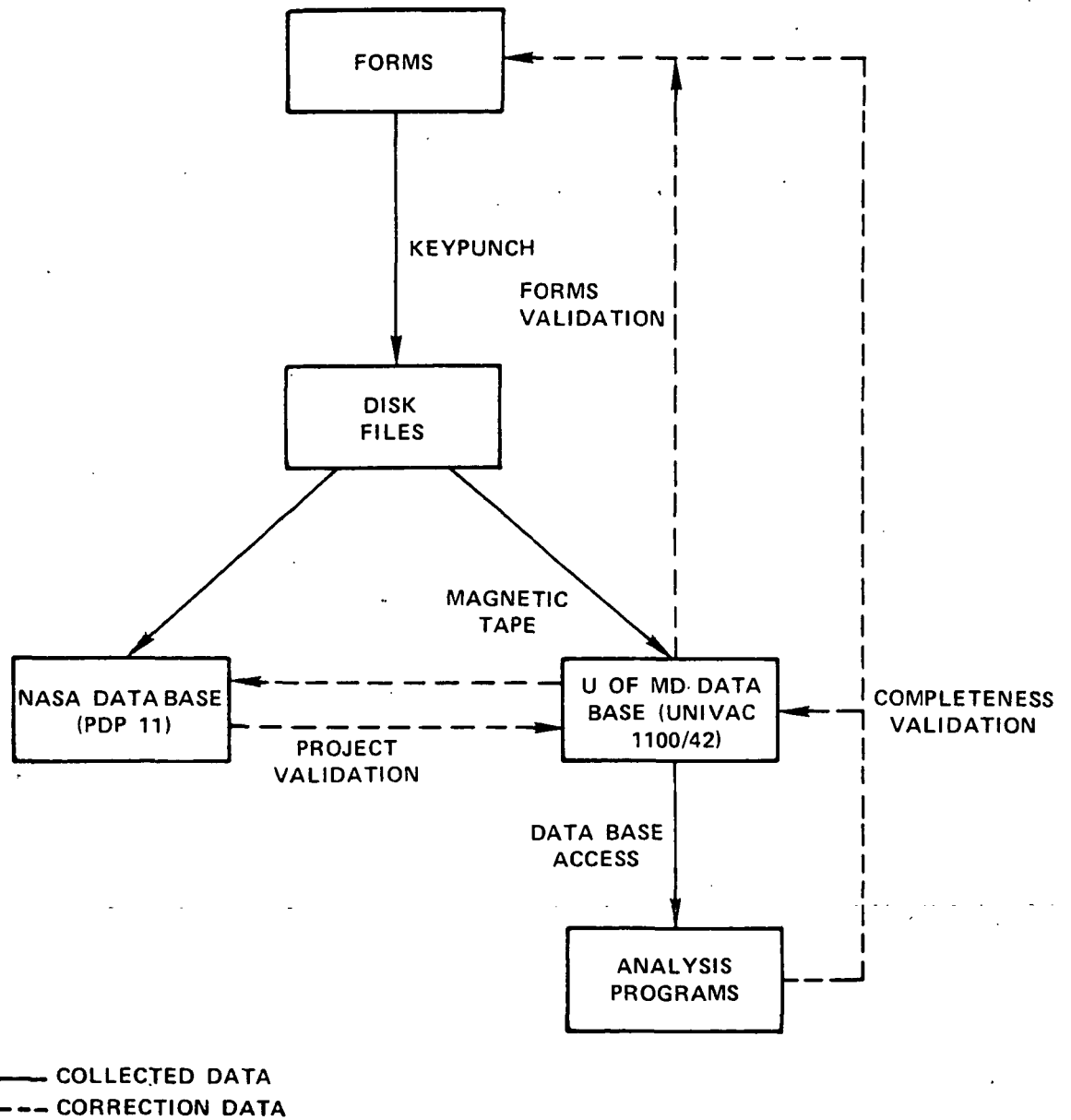
The more interesting question is completeness and consistency validation. This attempts to determine if there is any underlying structure or biasing in the ways forms are being filled out.

The initial approach is to use cluster analysis. Each of the forms is represented as a multidimensional vector of M dimensions. Each vector is projected onto a N -dimensional space using a subset of the M components as a basis. It is determined which forms cluster near one another in this N -dimensional space – such forms being considered related according to the basis chosen. Various regression techniques are being used to see if any of the other $(M-N)$ attributes are predictors of this clustering.

Some of the issues being initially investigated include: Is the programmer identification a predictor of the cluster? (It shouldn't be.) If so, then some of the programmers fill out the forms in certain characteristic ways which would show a biasing in the collected data. On the projects so far checked, this does not seem to be the case. Another question: Is the project name a good predictor when several projects are considered together? If so, then either there is biasing at the project level, or else different methodologies on different projects lead to different data being collected. If true, then the reasons will be investigated. A third initial question to be studied is: Are the clusters indicative of certain characteristic errors? Can clustering be used as an error classification?

While the work is still very preliminary, the use of such clustering techniques in this environment seems promising.

DATA FLOW THROUGH THE SOFTWARE ENGINEERING LABORATORY



FORMS COLLECTED

Resource Summary (by management)
hours/week/programmer

Component Status Report (by programmers)
hours/week/component/phase

Change Report Form (by programmer)
Each change or error, when found

Computer Run Analysis (by programmer)
Each computer run

General Project Summary (by management)
Each project

Component Summary (by programmers)
Each piece of system

DATA VALIDATION

1. **Forms Validation** – Each form is self-consistent, as it is entered into bases. Checks are both manual and automated.
2. **Project Validation** – Similar data on different forms for a project is analyzed, for missing or incomplete data.
3. **Consistency Validation** – Checks whether there is any systematic biasing of the set of collected forms between projects.

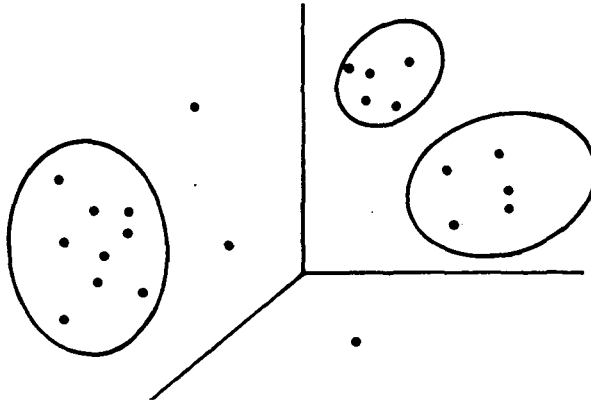
then –

Either:

- (a) Projects are not using same interpretation of instructions when filling out forms.
- or (b) Methodology used leads to characteristic differences in approaches to forms.

CONSISTENCY VALIDATION

- Basic approach uses cluster analysis.
- Each form a multidimensional vector of N dimensions.
- Choose M of those components.



- Objects near one another are “related” by M chosen elements are in same cluster.

Question: Is any one of the $(N-M)$ remaining components a predictor of cluster?

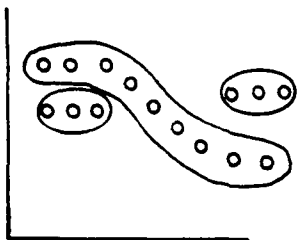
CLUSTERING ALGORITHM

1. Compute similarity between vectors (forms) I and J. Call it S_{ij} . S_{ij} will have a value between 0 and 1.
2. Choose some threshold B between 0 and 1.
3. If $S_{ij} > B$ then I and J are similar, so set $D_{ij} = 1$. Otherwise set $D_{ij} = 0$.
4. When viewed as a graph, $D_{ij} = 1$ represents that node I is connected by an arc to node J. Compute transitive closure $D^* = D + D^2 + \dots + D^n$.
5. $D^*_{ij} = 1$ if and only if nodes I and J are in the same connected subgraph. These connected subgraphs represent similar forms.

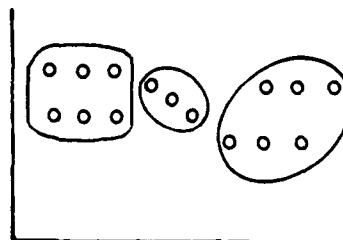
RESEARCH IDEAS


1. Vary B and measure effects on cluster sizes. The larger the B , then the fewer the vectors that will be similar. For the following graphs, $B = 0.950$.
2. Vary clustering algorithm. Current algorithm computes dot product of unit (normalized) vectors. Alternative strategy is to compute clusters as those vectors closest to some centroid instead of simply within the same connected subgraph.

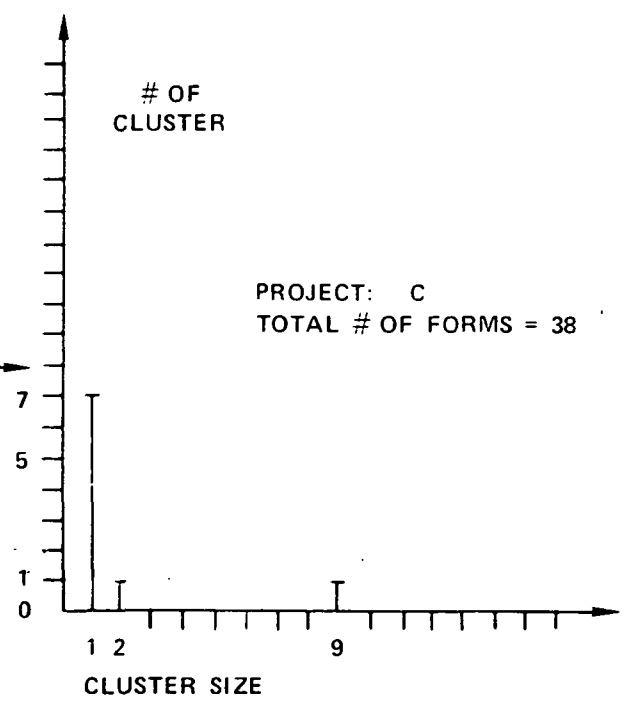
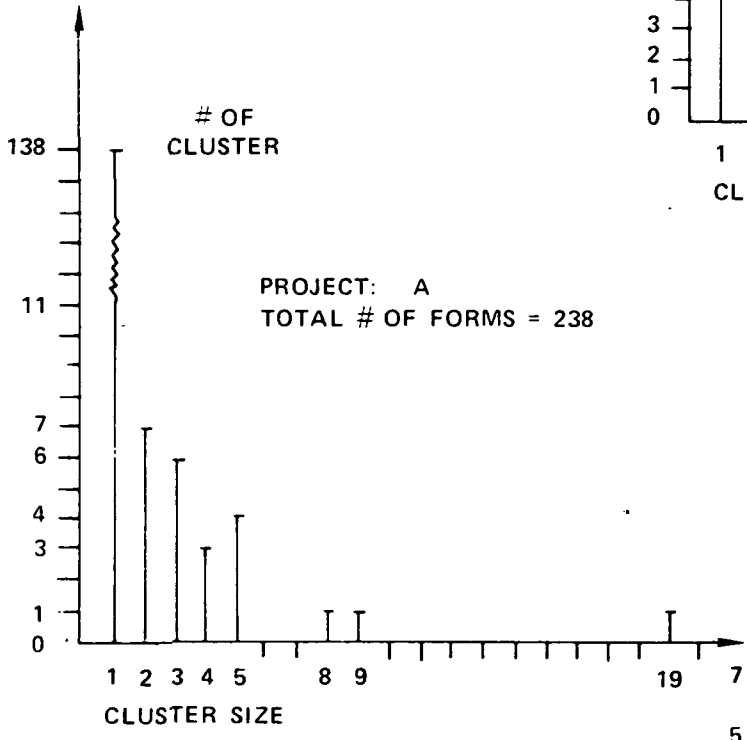
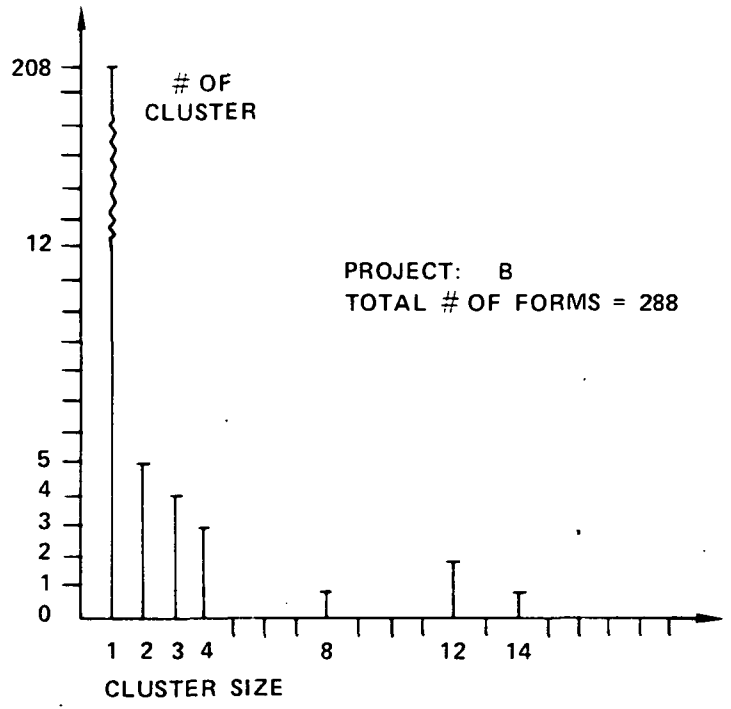
CURRENT ALGORITHM

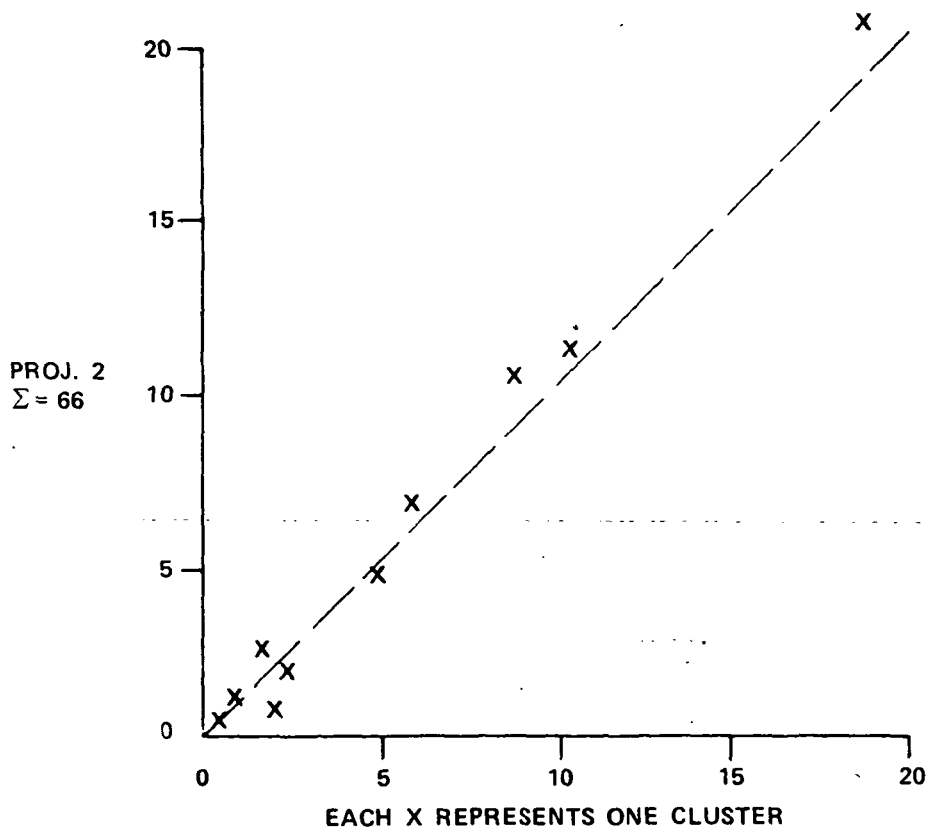
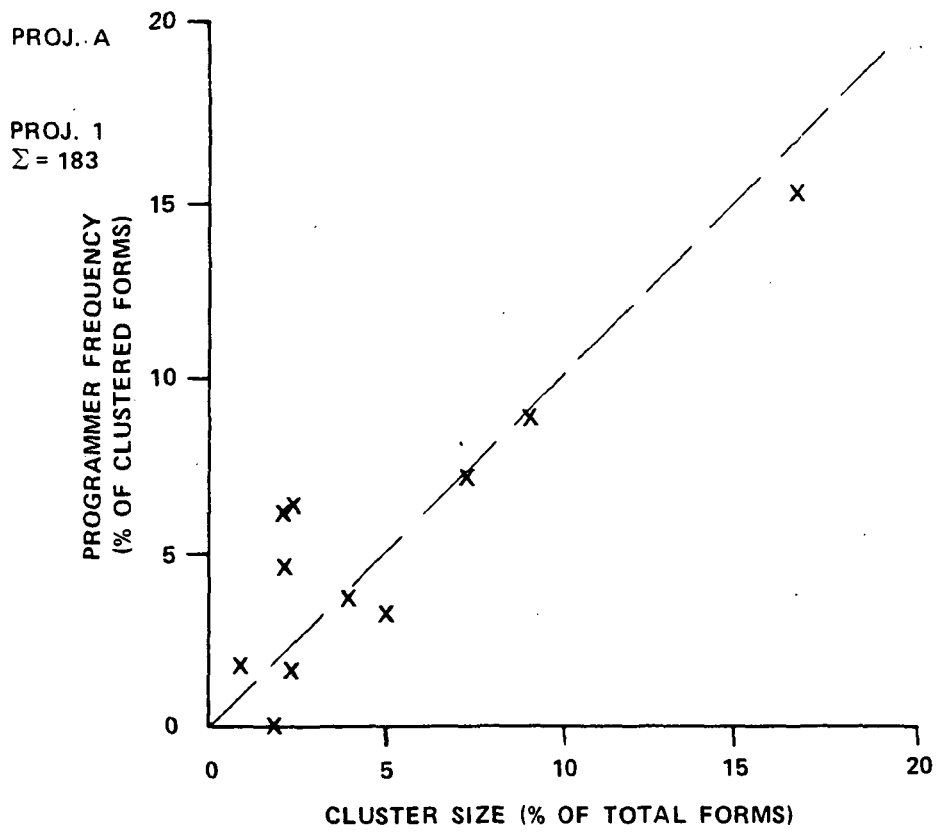


ALTERNATIVE ALGORITHM



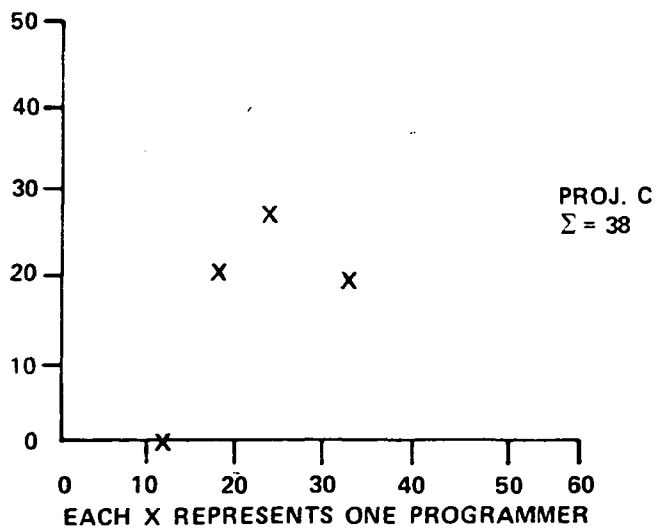
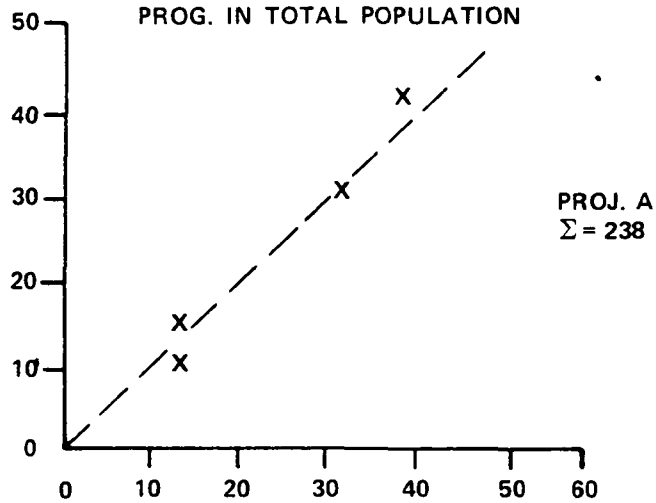
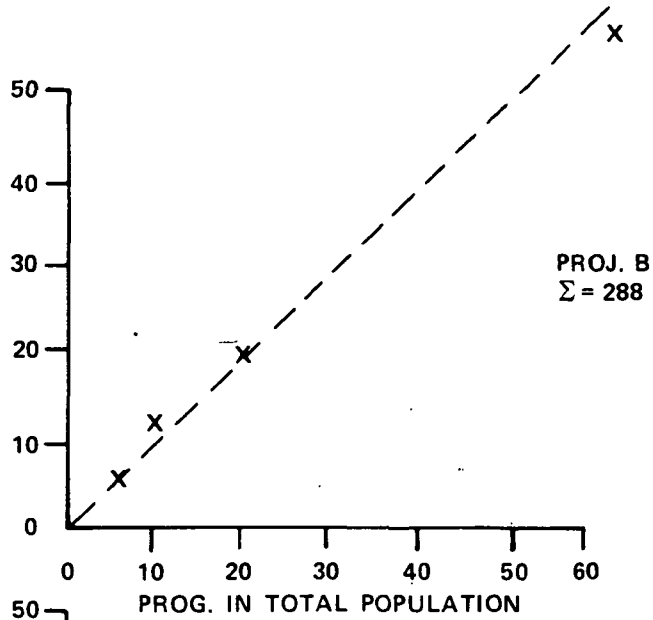
 CLUSTERS





A GIVEN PROGRAMMER APPEARING IN A GIVEN CLUSTER IS PROPORTIONAL TO CLUSTER SIZE - SHOWING EVEN DISTRIBUTION OF FORMS IN EACH CLUSTER

PROG. IN SINGLE
CLUSTER GROUPS



EACH X REPRESENTS ONE PROGRAMMER

FORMS THAT DO NOT CLUSTER ARE PROPORTIONAL TO NUMBER OF FORMS
BY A GIVEN PROGRAMMER - SHOWING EVEN DISTRIBUTION