THE BELL SYSTEM
TECHNICAL JOURNAL

VOLUME XLVI FEBRUARY 1967 NUMBER 2

Methods of Interpreting Diagnostic Data for Locating Faults in Digital Machines

By H. Y. CHANG and W. THOMIS

(Manuscript received August 29, 1966)

Several techniques for translating the results of diagnostic tests into specific fault identities are described. This translation can be difficult in large and complex machines. The amount of test data required to isolate faults, and the obscure symptoms some faults generate, preclude efficient manual test-by-test interpretations.

The additional observed fact that a significant number of faults yield inconsistent test results from diagnosis to diagnosis demands a flexible interpretation of data. Techniques are described for producing fault dictionaries which can be used by the maintenance craftsman to identify machine faults in a relatively short time. These techniques utilize multidimensional geometric representations of diagnostic results, methods for identifying and ignoring inconsistent tests, pseudo-random mappings, and other procedures for condensing and organizing the information contained in diagnostic test data.

The results of applying these techniques to data obtained from the Bell System's No. 1 Electronic Switching System are also discussed.

I. INTRODUCTION

The problem of locating faults in digital systems is becoming more acute with the increased complexity of these machines and their expanding use in real-time applications. The need for automatic testing techniques for locating faults by automatic programmed diagnostic

tests has been realized for some time and is being actively pursued in many areas.[1, 2] This need has particular urgency for a system designed to provide uninterrupted service. Such systems require extensive subsystem duplication as well as facilities for rapid fault isolation, location, and repair. The Bell System's No. 1 Electronic Switching System (No. 1 ESS), designed to control telephone switching functions, is an example.[3]

When machines were smaller and simpler, even the use of diagnostic tests to provide relevant trouble location symptoms was rare. An expert could usually locate the trouble by a quick survey of the behavior of the machine. This may be described as the "eureka" approach.

With large complex machines, however, analysis of symptoms by mere observation is lengthy and costly. Strange behavior sometimes occurs which even the expert is hard put to explain. Even the addition of test points and special diagnostic tools such as programmed testing may not immediately clarify the situation. Further, the sheer quantity of test data necessary to isolate a trouble to one of the myriad of components comprising the machine often demands preprocessing before presentation to the maintenance craftsman.

The techniques described in this paper evolved during the development of No. 1 ESS. They were devised in response to the need for translating the output of diagnostic programs into specific fault identities. Such translation techniques had to use a minimum of real time and memory while being accurate and rapidly applicable by relatively unskilled personnel. Our results were obtained by applying these techniques to the data generated on the No. 1 ESS. This system and particularly its central processor is sufficiently like other digital machines that we believe our conclusions have some general validity. For readers who are not familiar with No. 1 ESS, a brief description of its functions and maintenance plan is provided in Appendix A.

Section I reviews several conventional diagnostic data interpretation techniques—the so-called exact-match presentations, and points out some of their shortcomings. Section II reports some general facts concerning the inconsistency of test results and suggests a number of solutions to the problem. These solutions fall into two general categories: the phase dictionary approach and the cell dictionary approach. Section III describes the phase dictionary approach, its implementation in No. 1 ESS, its advantages and disadvantages, and also discusses some alternate techniques. Section IV introduces the cell dictionary approach and describes how it was implemented on No. 1 ESS. Evalua-

tion results, obtained by applying these techniques to data obtained from No. 1 ESS, are discussed in Section V.

1.1 *Terms and General Background*

A *diagnostic test* consists of the application of special inputs to a machine for the purpose of locating a possible fault. The corresponding responses are termed the *test results* or diagnostic data. A *fault* can be defined as a physical defect in a logical element which will cause incorrect machine operations. Test results are generally processed and interpreted, either automatically by programs or by other manual means, to give field maintenance personnel the necessary information to locate and identify the faulty circuit component or packages. A *circuit package* is the smallest replaceable module of a machine. In No. 1 ESS, it consists of a relatively small number of components mounted on a printed wiring board.[3]

There are at least two types of programmed procedures for fault diagnosis: the "combinational" approach and the "sequential" approach. In the combinational approach a fixed set of tests is applied to the machine, and the results analyzed to identify the fault. The identification process generally utilizes a *fault dictionary* which is a listing of the test results of known faults organized in a fashion convenient for look-up.[2] In the sequential approach, the set of tests applied to the machine is not fixed.[4] The result of each test is used as a basis for determining the next test to be applied. Each fault, or a group of faults giving identical diagnostic results, is then identified by a certain test sequence—no additional data analysis or dictionary look-up is necessary. It is noted that this distinction is to some extent academic. A sequential analysis can be performed on data generated by the combinational approach. In cases where faults cause large numbers of tests to give inconsistent results, this may be advantageous since the sequential approach will be costly in terms of memory storage. Only if the average running time of the sequential approach is significantly less than the combinational approach *and* time is at a premium will the sequential approach be a better choice. For these reasons, the combinational approach is used in No. 1 ESS.

The fault diagnostic information necessary for generating dictionaries can be obtained by two fundamentally different procedures. The two approaches are known as "program simulation" and "hardware simulation." In program simulation the logic description of a machine is compiled into a computer program which is designed to simulate the behavior of the object machine. A particular fault can then be

"introduced" into the object machine simply by appropriately changing the program description of the machine's logic. Subsequent logical simulation of the machine under control of its diagnostic program reveals the object machine's actions under test in the presence of the fault. In hardware simulation, faults are physically introduced into a real machine by replacing good circuit components with catastrophic failures such as shorts or opens. The diagnostic tests are performed and results recorded each time a fault is inserted. A fault dictionary can then be generated by processing test results obtained by either method.

1.2 *Straightforward Dictionary Presentations*

One method of presenting diagnostic data for dictionary use is simply to list for each fault only the failing tests. This method is quite efficient when, on the average, few diagnostic tests fail.[5] A sample page (with added comments) of such a listing appears as Fig. 1 (a).

In this example, the tests were grouped into so-called test phases such as phase A, B, . . . G, H, . . . etc. The tests in each phase are numbered sequentially. Ordering of entries in the dictionary is first by phases in alphabetical order, then by test numbers within the phase. This type of dictionary was employed in the first Electronic Central Office which was in commercial use at Morris, Illinois between 1960 and 1962. A detailed description of its format and implementation is contained in Ref. 5.

As can be seen, such a technique enables a relatively unskilled maintenance man to trouble-shoot by merely searching for matches of any particular pattern with entries in the dictionary. Furthermore, in this representation, the exact configuration of test results is preserved. This feature may be useful when dictionary look-up fails to locate the trouble. The maintenance man may be able to locate faults by a direct examination of the test pattern with the aid of other documents such as diagnostic program listings, logic flow charts, etc.

However, this type of presentation suffers from the disadvantage of bulk when the diagnosis is of any considerable size (i.e., more than about 1000 tests). Further, large numbers of diagnostic tests result in large and complicated patterns, which in turn increase the difficulty of finding matches with dictionary entries [see Fig. 1 (b)]. This technique is one of a class of methods called *exact-match* techniques since an exact match between a test pattern generated by a real fault and a pattern in the dictionary is required to identify the trouble.

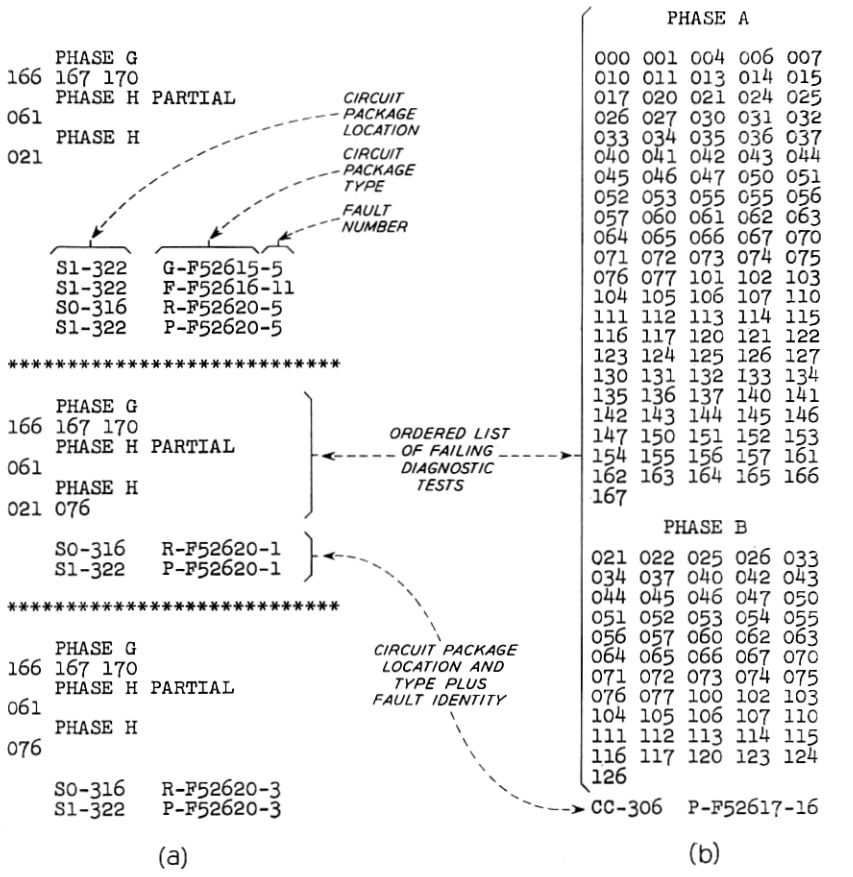


Fig. 1 — Sample page of fault dictionary.

1.3 *An Improved Exact-Match Technique*

In order to reduce the problem of dictionary bulkiness and the difficulty of manual look-up of large and complicated fault patterns, a "number generation" technique is used in No. 1 ESS. This technique is essentially a "hash" storage technique which effectively "reduces" each diagnostic to a smaller fixed-length decimal number by means of a psuedo-random mapping.[6] Conceptually, the probability of mapping two or more large binary numbers into the same decimal number of relatively small size can be made arbitrarily small by a proper selection of the decimal sample space. A detailed discussion on the particular technique used in No. 1 ESS is incorporated as Appendix B.

One interesting result is that if the mapping is truly random, the expected number of diagnostics that will map into the same random number is

$$E_r = k - N\left(1 - \left(1 - \frac{1}{N}\right)^k\right),$$

where N is the number of diagnostic test results to be mapped and k denotes the smallest integer greater than or equal to $\log_{10} N$, i.e., the number of digits required in the decimal number. Thus, for No. 1 ESS, in which the largest sample of fault patterns for any unit is no greater than 10^5, a 12-digit representation will suffice to insure that the number of replications, as a result of the data reduction mapping, will be less than one (if, of course, the mapping is truly random).

Fig. 2 shows a sample of a No. 1 ESS exact-match dictionary entries. Each of the 12-digit numbers was derived from a fault pattern by a pseudo-random manipulation of the pattern. As described in Appendix B, the results of this pseudo-random mapping method agreed very well with the theory.

It is estimated that a five-to-one reduction in bulk was thus achieved. In addition, it was found that the time required for look-ups was greatly reduced.

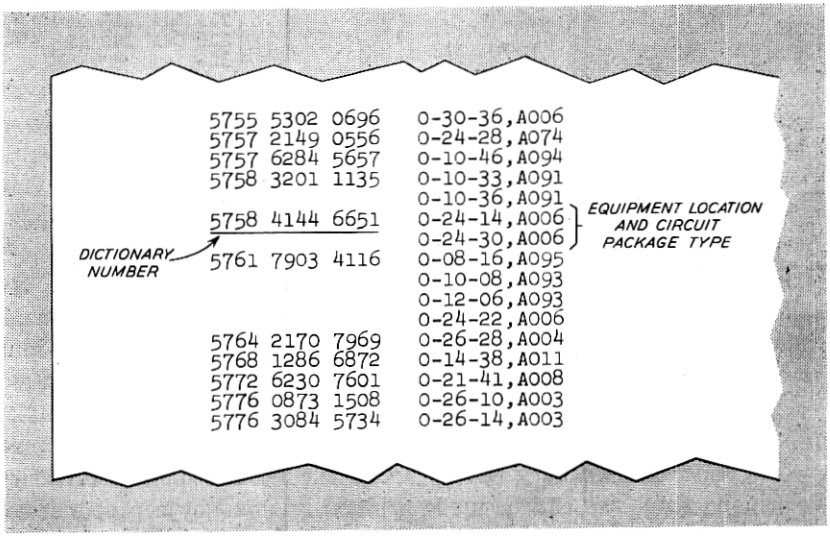


Fig. 2 — Sample format of No. 1 ESS dictionary entries.

1.4 *Shortcomings of "Exact Match" Techniques*

A major problem in using such approaches is that occasionally a diagnostic generated in the field does not exactly match a dictionary entry. This will arise if the results of the field diagnosis of a given fault differ from those yielded by a diagnosis on the same fault which was used to prepare the dictionary, i.e., the test results are *inconsistent.* There are many possible causes of inconsistent test results. Some of the more obvious ones are (*i*) improper machine initialization, i.e., if the effect of the fault is such that it prevents the machine under diagnosis from being properly initialized, the test results may then vary from time to time, depending on the state of memory elements at the time the fault occurs, (*ii*) presence of intermittent or marginal faults, i.e., faults that cause machine malfunction at some times but not at others, and (*iii*) other factors such as the presence of noise and/or variations in circuit component values, etc. Consequently, supplementary techniques are desirable.

II. GENERAL FACTS CONCERNING INCONSISTENT TEST RESULTS

Suppose one had the fault patterns for a number of sample faults that were simulated on a test model of a digital machine for purposes of producing a fault dictionary. Further, suppose that these same faults were introduced into a different but supposedly identical machine and the fault patterns collected. A comparison of the two sets of data is revealing.

Two such sets of data were collected using No. 1 ESS. A sample of faults was inserted in the central processing unit of a No. 1 ESS office in Chase, Maryland and compared with dictionary results obtained from another No. 1 ESS at the laboratory. The sample consisted of 302 faults selected so as to be both well distributed and representative of expected troubles. Of the 302 faults inserted, 58 produced printouts that could not be found in the exact-match dictionary. For various reasons, only 40 of these 58 inconsistent printouts could be analyzed.* This experiment and a comparison of the test results of these 40 faults with their corresponding dictionary patterns show that:

(*i*) About 15 to 20 percent of the data for corresponding faults disagree.

(*ii*) Among the diagnostics that are inconsistent, the majority differ in only a few bits.

---

* This experiment is reported fully in Section 5.1.

(*iii*) Furthermore, these differences generally cluster, i.e., a relatively small group of adjacent test bits are affected. Several groups may be affected but in general, the groups do not consist of more than about 25 tests (about $\frac{1}{2}$ percent of the overall diagnostic).

(*iv*) Within these clusters the diagnostics produced by the test model tend to have somewhat fewer failing tests.

(*v*) Only about 25 percent of the inconsistent diagnostics have extensive differences.

These observations are qualitative in nature and can only claim to be representative of No. 1 ESS. However, they serve to suggest a number of ways to attack the problem.

Observation (*iii*) above suggests that two differing patterns for the same fault could be made to match if the cluster of differing test bits were masked out. This idea resulted in the Phase Dictionary, the Phase Prime Dictionary, and the Test Group Dictionary (see Section III).

Observations (*ii*) and (*iv*) suggest that the two differing patterns are related and that their relationship could be expressed perhaps by some function of their Hamming distance. This idea resulted in various forms of "Cell" Dictionaries (see Section IV). We shall consider first the Phase Dictionary and some of its relatives.

III. PHASE DICTIONARY APPROACH

3.1 *Characteristics of Diagnostic Data*

The usual technique used to design diagnostic tests for a large and complicated machine is to divide the machine functionally into many small and disjoint (if possible) logic blocks. A logic block can be taken as a group of functionally related circuits, such as an order decoder or an index register, etc., whose input-output terminals are readily accessible. The tests are then designed to pinpoint faults which may exist in each logic block, assuming other logic blocks in the machine are faultless. The overall diagnostic is therefore composed of a concatenation of test results of many so-called *test phases,* each of which consists of tests that are aimed at testing a particular logic block.

Normally, when a fault occurs, it is expected that the fault will be detected by many of the tests that are specially designed to test that part of the circuitry where the fault lies. Thus, one would expect that in each overall diagnostic, the test failures would be roughly distributed over a certain number of test phases, rather than over all test phases. This is, indeed, the case as one examines, for example, the

diagnostic data of one unit of the No. 1 ESS central processor complex, the *Central Control*[3] (see Fig. 3). The Central Control has a total of 28 test phases; the overall diagnostic is about 5000 bits long. Out of 102,518 faults simulated, over 97 percent of them yield diagnostics which indicate some-tests-failed in only four or fewer test phases.

3.2 *General Description of Method*

The essential idea of the Phase Dictionary approach lies in the notion of identifying "phase diagnostics," i.e., the failure patterns of individual test phases. That is, in addition to the normal "exact-match" dictionary (as it is described in Section 1.3) one further prepares a supplementary *Phase Dictionary* which is divided into many chapters, each of which is produced by processing each diagnostic phase as if it were the whole diagnostic. Then a fault would be redundantly identified by a number of dictionary entries: one in the exact-match dictionary and many in the phase dictionary, as many as the number of test phases that failed in its overall diagnostic. For example, a fault, f_i, which has failed some tests in test phases 2, 3, 4, and 6 would be identified by an entry in the exact-match dictionary and four additional entries in the phase dictionary, one each in chapters designating test phases 2, 3, 4, and 6.

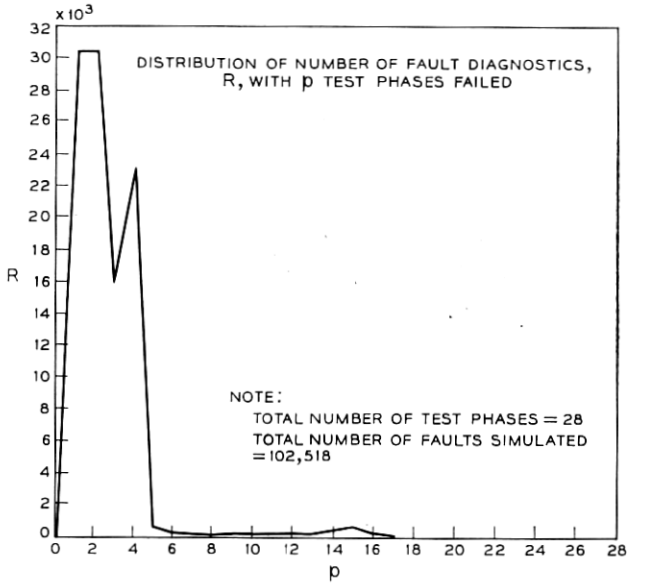


Fig. 3 — No. 1 ESS central control diagnostics.

Now suppose f_i is diagnosed in the field and its failure pattern obtained. Five numbers will then be generated: a normal exact match dictionary number, which is generated by the overall pattern of the diagnostic and four phase numbers, each of which results from manipulating diagnostics of test phases 2, 3, 4, and 6. The maintenance personnel normally will first try to find the overall number in the exact-match dictionary. If there is a match, he can simply replace the circuit package(s) indicated by the dictionary entry. If, on the other hand, he cannot find a match, he will then consult the phase dictionary by matching phase numbers with dictionary entries in chapters 2, 3, 4, and 6. Assume the inconsistency in the diagnostic is small and is confined to, for example, test phase 3 only. The maintenance man will probably discover that he can successfully match phase numbers for test phases 2, 4, and 6 with some entries in chapters 2, 4, and 6. He will not, however, find a match in chapter 3 because the field diagnostic of f_i which generated the phase number for test phase 3 *differs* from the diagnostic which was used to generate the dictionary entry. Nonetheless, the maintenance man can still examine the circuit package(s) indicated by the phase dictionary entries of test phases 2, 4, and 6 where he finds a match to determine which package(s) is to be replaced. Since he knows that the circuit package(s) associated with fault, f_i, ideally would be listed under all of these entries, he should, therefore, select the package(s) that has appeared the greatest number of times. This majority rule approach not only leads one toward replacing the proper circuit package(s) but also reduces the number of unnecessary package replacements.

A slightly more sophisticated form of the phase dictionary can eliminate the need for the somewhat laborious majority rule approach. The phase dictionary consists of entries formed by masking out all failing test bits except those in a single phase. The *Phase Prime Dictionary* consists of entries formed by masking out only a single phase at a time. Thus, if tests in phases 3, 4, and 5 failed during diagnosis, a number for phases 3 and 4, one for 3 and 5, and one for 4 and 5 would be produced. Then if the inconsistencies fall only in say phase 3, only the number produced for phases 4 and 5 could be matched. Note that the majority rule is not needed here as the dictionary for phases 4 and 5 has already performed that function automatically.

The phase prime dictionary, of course, is a little less general than the phase dictionary in that it is useful only if inconsistencies are confined to a single phase.

3.3 *Implementation in No. 1 ESS*

The general program flow for implementing the phase dictionary is shown in Fig. 4. The diagnostic data obtained through simulation are originally stored on magnetic tapes. Each diagnostic is read into core and is divided into many segments, one segment of each test phase. A phase dictionary number is then generated for each test phase by manipulating diagnostic data in that phase. The phase dictionary number computation is performed for each phase of all fault diagnostics. All phase dictionary numbers of the same test phase are grouped together and sorted. The sorted listings are then printed to form the various chapters of the phase dictionary.

3.4 *Advantages and Disadvantages*

The major advantage of the phase dictionary over the exact-match dictionary is its ability to locate faults whose diagnostics are inconsistent. That is, if the inconsistency of test results is confined to a small number of test phases, the phase dictionary can still locate faults by matching phase dictionary numbers. Further, the use of phase dictionary numbers also offers a possibility for identifying marginal faults. This can be done by repeatedly exercising (on-line) certain phase(s) of the diagnostic tests and then matching the phase numbers, since repeatedly exercising all diagnostic tests could be extremely costly in terms of system real time.

The phase dictionary in general tends to be bulkier than the exact-match dictionary. This is because each fault is multiply listed in the

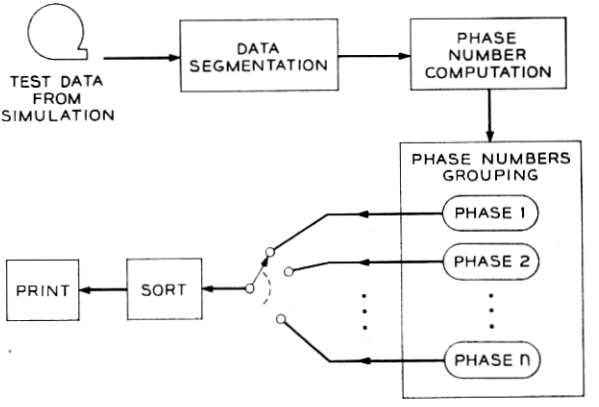


Fig. 4 — Program flow for phase dictionary implementation.

dictionary. For the particular example shown in Fig. 3, a phase dictionary is approximately $2\frac{1}{2}$ times the size of the exact-match dictionary, as can be determined by calculating the average number of failing test phases per diagnostic. Another shortcoming of the phase dictionary is that if the inconsistency of test results affects all test phases or if the phase which the inconsistency affects is the *only* failing phase in the overall diagnostic, the fault cannot be successfully identified with this approach. Although wide variations in test results for the same fault are less likely to arise in a good diagnostic design, they nevertheless represent a problem.

3.5 *The Test Group Dictionary*

A third form of dictionary that uses the idea of masking out inconsistent tests has been investigated. This dictionary was constructed by considering each test of the diagnosis independently and determining what faults caused this test to fail. Those faults would then be grouped and associated with that test. When this was done for all tests, the *Test Group Dictionary* was formed.

It was originally hoped that the list of faults associated with a given test would be small enough on the average so that a majority rule technique could be used to advantage when analyzing a field diagnostic result. Unfortunately, in all of the cases implemented so far, the average listing is well over 30 faults per test. This precludes its use as a manual tool. If stored on tape, however, and searched by machine, it gives promise as being a valuable laboratory tool. It might also be useful if the maintenance facilities for a number of machines were centralized.

IV. CELL DICTIONARY APPROACH

4.1 *Introduction*

The phase dictionary approach is a technique for finding exact matches between patterns by eliminating those portions of the patterns where differences are found. The "cell" dictionary approach, on the other hand, is not concerned with exact matches between patterns but with near matches. In the cell approach, the entire fault pattern is examined and a measure of its "similarity" to other patterns in the dictionary is made. On the basis of this measure, those faults in the dictionary associated with the most "similar" patterns are identified as being more likely to have caused the observed fault pattern.* This

* This assumption can only be justified ultimately by satisfactory results in its application.

then provides the maintenance man with a list in order of probability of faults to repair.

Clearly, the efficacy of this approach depends on (*i*) the way in which "similarity" is defined and measured, and (*ii*) the manner in which this measure is used.

4.2 *"Similarity" and "Dissimilarity" of Diagnostics*

If every test in a diagnosis was of equal significance in the task of isolating faults, one could say that two diagnostic test patterns were dissimilar in proportion to the number of corresponding tests in which the two patterns differ. If the diagnostic results were expressed as binary numbers, their dissimilarity could be expressed as having a direct relationship to their Hamming distance. The greater the Hamming distance, i.e., the more places in which two patterns differ, the less alike the patterns are. Conversely, the smaller the Hamming distance, the more similar two patterns are. Consequently, Hamming distance can be used a measure of both similarity and dissimilarity.

Hamming distance is a readily computable measure. And further, it can be easily modified to take into account the fact that diagnostic tests differ in *significance*. For the purposes of explaining the idea of significance, consider a small sample of faults and just those tests of the diagnosis which fail for at least one fault in that sample. Suppose a number of these tests always pass or fail together as a group for any given fault. Then the results of the entire group could be predicted by examining the result of any one of the tests. Here the information provided by each test in the task of locating a fault is the same but diluted by a factor equal to the number of tests in the group. One would say that the significance of these tests was low. On the other hand, a test whose result cannot be predicted by the results of other tests would have a higher significance. Intuitively speaking, the significance of a test should also be affected by its consistency. Less significance should be attributed to inconsistent tests.

A natural and satisfying way of expressing these considerations quantitatively would be to assign weights to tests in accordance with their significance. Section 4.9 gives a brief discussion of some mathematical techniques available to do this.[8]

Suppose now only that each test has been assigned a weight. Then similarity would be computed by summing the weights of the differing tests in the patterns. Similarity is, thus, now measured by a weighted Hamming distance. Whether or not test weights have been assigned, however, similarity is measured by an easily calculated number.

4.3 *A Problem*

It would be possible to store the dictionary (i.e., all the fault patterns) in a computer memory and to use the similarity measure just described to search the dictionary for patterns similar to some arbitrary pattern when desired. As a practical matter, however, the time required for the search and the storage required for the dictionary are prohibitive. In the case of No. 1 ESS, for example, it is estimated that about 2.5×10^7 36-bit words would be required to store the dictionary for the central control alone. The time required for a single search would be on the order of 10 minutes provided the dictionary were stored in core (rather than on tape or disk) and provided the central control did no other work. Since No. 1 ESS is a time-shared machine, no one job is run for longer than 10 percent of any extended interval. Consequently, a single search would run well over one hour. Clearly, a more sophisticated method is needed.

4.4 *A Geometric Model for the Dictionary*

Consider, for the time being, that diagnostic results are representable by binary numbers and that similarity is measured by simple Hamming distance. It is possible to construct a geometric analog of the binary number system in binary space. In this analog, each binary number represents a unique point in the space and the "distance" between points corresponds to the Hamming distance between the patterns representing those points. Thus, there are exactly 2^N distinct points or patterns in an N-dimensional space.

Suppose that all of the data in the dictionary were placed in some N-dimensional space. For example, the dictionary data for the No. 1 ESS central control consists of about 10^5 different patterns of order 5000 (i.e., N equals 5000). Thus, 10^5 points out of a possible $2^{5000} \approx 10^{1500}$ points would be taken up by dictionary data. (The space is very sparsely populated.) Now suppose some arbitrary pattern (produced by some real fault) is placed at point A in the N-space. If point A is already occupied by a dictionary pattern, then the two patterns match and the real fault is almost certainly the same as the dictionary fault associated with that pattern. If, however, point A is unoccupied, we have (presumably) the case of an inconsistency. Then it would follow, since Hamming distances are preserved in the analog, that those dictionary points that are closest spatially will be the most similar ones according to our definition. The faults associated with these nearby points would then represent those dictionary faults which could most probably have produced the inconsistent pattern.

4.5 *The Cell Dictionary Concept*

Imagine first that a number of points, $c_i (i = 1, 2, \cdots, m)$, in the N-space are selected arbitrarily. Now imagine that every point, x_k, in the space is associated with a c_i such that the Hamming distance $d(c_i, x_k)$ is least. This procedure would then produce a number of "cells" C_i with "centers" c_i, each containing all points x_k, such that $d(c_i, x_k) < d(c_j, x_k)$, for $i \neq j$. Applying this procedure to dictionary data will result in a *Cell Dictionary*. The dictionary would consist of an ordered list of those cells which contained diagnostic results together with the fault identities corresponding to those results. Each cell can be conveniently identified by its c_i. Such a dictionary could be used as follows:

(*i*) Place in memory a list of occupied cells (i.e., cells which have diagnostic results associated with them).

(*ii*) Given an arbitrary pattern, a search would consist of computing the cell containing it and finding the two or three closest occupied cells.

(*iii*) Likely faults could now be found by consulting the printed dictionary.

This method would be practical if such cell lists were relatively small. Unfortunately, No. 1 ESS data does not lend itself to small lists. Therefore, a different form of cell dictionary was adopted.

4.6 *Selection of Practical Cell Center Lists*

In this section, we shall discuss an algorithm which permits the generation of a list of cell centers (c_i's) that, given an arbitrary pattern, can be rapidly searched.

Consider all binary numbers of order N (i.e., having N bits) and a partitioning of the N bits into k equal parts (assuming N is divisible by k). Suppose that all bits in each part, P_i, of the partitioning are assigned like values—either all 0's or all 1's. Then the subset of all numbers of order N meeting the above requirements can be placed into one-to-one correspondence with the set of all binary numbers of order k. This subset can then be taken to form a set of 2^k cell centers which divides the N space into 2^k equal-sized (i.e., containing the same number of points) cells.

To show that every point is contained in some cell and every cell contains the same number of points, imagine an arbitrary pattern of order N is divided into k equal parts. Then all binary numbers having more 0's than 1's within part P_i of this pattern are closer Hamming distance-wise to a cell center whose P_i is all 0's. Similarly, all binary

numbers having more 0's (1's) than 1's (0's) within other parts, P_i's, of this pattern are closer to a cell center whose P_i's are all 0's (1's). It follows, therefore, from the definition of a cell that each point is contained in some cell. Moreover, if N/k is odd, each point will be contained in one and only one cell.

Since, in general, there are C_m^n possible binary patterns of order n with m 0's (or 1's),* the number of binary numbers having more 0's than 1's in P_i equals the number of binary numbers having more 1's than 0's. Since this holds for any P_i, and further it holds independently of any other P_i, it holds for the entire partition. Thus, each cell center will have exactly the same number of points closer to it than to any other cell center.

From the nature of the construction of cell centers, a binary number can be assigned to a cell merely by partitioning it and counting either the 0's or 1's in each part. This greatly simplifies and speeds up the assigning of patterns to cells. Furthermore, cell sizes can be varied at will merely by changing the partitioning. Very large cells will be generated if the P_i's are large and vice versa.

4.7 *The Multiple Cell Dictionaries Approach*

When these techniques were applied to No. 1 ESS, it was decided to produce a number of cell dictionaries, each corresponding to a different cell size. This approach evolved because of the problems encountered in apparently simpler implementations. For example, as was discussed in Section 4.5, one way of implementing cell dictionaries in the field would be first to produce only one cell dictionary from the laboratory data. Then, a list of those occupied cells (i.e., cells containing dictionary fault patterns) would be stored in the field ESS machine. This would enable the machine to take a pattern for a field trouble, compute the cell containing it, search the cell list, and print out a number of nearby occupied cells. In the case of No. 1 ESS, however, the list of occupied cells was very large (requiring on the order of 15,000 36-bit words of memory) and search times prohibitively long. The next possibility, which was tested and then discarded, was to print the cell dictionary (i.e., an ordered list of occupied cells) and search it manually. A search consisted of checking whether or not a cell containing a real fault was occupied. If it was not, then a check of the nearby cells was necessary. Thus, the search required that No. 1 ESS machine compute and print the cell containing the real fault and its neighboring cells in N-space. Unfortunately, the number of nearby cells in N-space can be enormous. For

* C_m^n = the number of combinations of n out of a total of m things.

example, there are, in general, C_d^k adjacent cells a distance dN/k from any given cell (N and k are as previously defined and $d = 1, 2, 3, \cdots$). Thus, if $N = 5001$ and $k = 1667$, the closest cells to the given $d = 1$ will be a distance of only 3 away and there will be 1667 of them. For $d = 2$, the number of cells will be about 1.4×10^6. As a result, unless the cell containing the fault pattern was occupied, the time for finding any nearby occupied cell could be extremely long. Consequently, a modification of this idea, the so-called *Multiple Cell Dictionaries* approach, was finally adopted.

Suppose the ESS machine computes the name of the cell that contains a real fault, and a check in the cell dictionary shows that the cell is unoccupied. Now suppose the machine repeats a similar calculation to obtain the name of a larger sized cell. Then, in general, a different cell name will result. In order to check whether or not this cell is occupied, a cell dictionary corresponding to this sized cell would have to be available and searched. Suppose the search was again unsuccessful; then the machine could compute another even larger cell and so on, for as many times as there are available dictionaries. Ultimately, as the cells grow larger, they must become occupied cells for some cell dictionary.

The computation of cells of different sizes for a real fault is relatively simple. In order to use the computed cell names, however, a number of corresponding cell dictionaries must be made available. Thus, the multiple cell dictionaries approach is a trade-off of bulk for search time. The increase in bulk, however, is not so great as might at first be suspected. This is because as the cell sizes increase the number of cells decreases and the number of faults in the "zero" cell* (which is not printed in general) increases. A qualitative evaluation of the results achieved will be presented in Section 5.1.

It should be noted that a partitioning such that each P_i consists of a a single bit will result in an exact-match cell dictionary. Thus, the cell approach can be extended to cover both the case of exact matches as well as the case of inconsistencies.

The form of cell dictionaries is exactly the same as the exact-match dictionary. This is achieved by scrambling cell center coordinates in a fashion similar to the procedure for scrambling diagnostic results for the exact-match dictionary (see Section 1.3 and Appendix B). The scrambled cell center coordinate serves as the cell name when printing the cell dictionary.

---

* The "zero" cell is one whose center has an all 0's pattern.

4.8 *Some Disadvantages*

Although the major advantages of the cell dictionary are fairly obvious, some of its disadvantages may not be.

First, the smallest practical cell sizes are formed with a partitioning such that each P_i consists of 3 bits. This means that any pattern that never groups at least two 1's within a single P_i will fall in the zero cell. Since the "all zeroes" diagnostic result represents the healthy machine, it could be expected that many faults that cause the machine to be "slightly sick" will fall into the zero cell or its neighbors. This is indeed the case as revealed in the No. 1 ESS diagnostic data. Thus, a fault falling in the smallest zero cell in many cases renders the cell dictionary useless because of its poor resolution. Fortunately, the situation is ameliorated by the fact that most inconsistencies occur with faults which cause many test failures.

A second disadvantage concerns the process of computing larger cells around the real inconsistent fault pattern. It is true that as the partitioning increases, larger cells containing the real fault are examined but two facts should be noted:

(*i*) The real pattern will not necessarily be close to the center of the cell, and

(*ii*) The centers of these cells will not usually coincide.

These facts mean that some faults not necessarily close to the real fault may occasionally be implicated and also that a cell may not be completely included in the next larger sized cell. Eventually, of course, as cell sizes increase, the smaller cells will be completely included but at the cost of a greater number of implicated faults.

Both of these considerations are affected by the algorithm used to select cell centers. A better algorithm might eliminate these problems. Also, an approach such as Kruskal's (see Ref. 8) can reduce if not eliminate them.

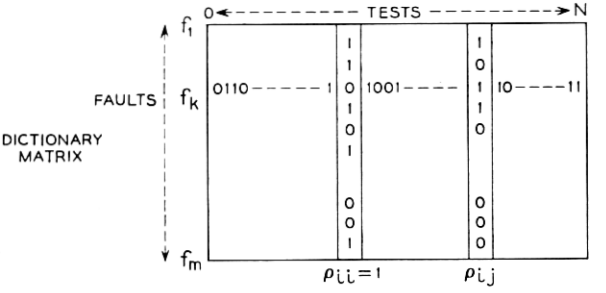
4.9 *Test Weighting*

The modification of similarity measurements by the inclusion of test weights should take the following considerations into account:

(*i*) The significance of a test relative to other tests in the task of isolating faults, and

(*ii*) The consistency of the test, i.e., what is the probability that on multiple diagnoses of the same fault, the test will give the same result as it gave on previous diagnoses.

Item (*i*) can be derived from the test results obtained from dictionary production. Consider a matrix formed such that each row represents the binary diagnostic result for a fault (see Fig. 5). Then each column represents the results a given test yields for each of the faults. To calculate the significance of a test relative to other tests, first obtain the correlation coefficients ρ_{ij} of column i with every other jth



$$\text{CONSISTENCY FACTOR} = \frac{\text{NO. OF CONSISTENT FAULTS}}{\text{TOTAL NO. OF FAULTS}} = C$$

$$\text{TEST WEIGHT } W_i = \frac{C_i}{\sum_{j=0}^{N} \rho_{ij}^2}$$

$$\text{SIMILARITY} \approx \frac{1}{\text{WEIGHTED HAMMING DISTANCE}}$$

Fig. 5 — Test weights.

column using standard statistical definitions. The relative significance of the ith test which we will call σ_i is

$$\sigma_i = 1 \Big/ \sum_{j=0}^{N} \rho_{ij}^2 .$$

ρ is squared to obviate sign difficulties and down-weight small correlations.

Item (*ii*) consists of the ratio of the number of faults for which the test was consistent to the total number of sample faults.* The con-

* This ratio may be obtained practically by repeatedly performing the diagnosis on a given fault and observing the results of the tests. This could be done during physical simulation (at a considerable time cost). Otherwise, field data would have to be collected over a period of time. Kruskal has an idea (yet to be verified) of how to predict the inconsistency of a test theoretically. See Ref. 8.

sistency factor c_i of test i can be defined as:

$$c_i = \frac{\text{total number of faults for which the test } i \text{ was consistent}}{\text{total number of faults}}$$

A test weight w_i then is

$$w_i = f(\sigma_i , c_i).$$

A suggested function is

$$w_i = (1 + c_i \log c_i + (1 - c_i) \log (1 - c_i))\sigma_i .$$

The quantity $[1 + c_i \log c_i + (1 - c_i) \log (1 - c_i)]$ is the usual entropy function. It essentially makes $c_i = \frac{1}{2}$ the zero point (which is the value we would expect if the test was completely inconsistent) and spreads the intermediate factors appropriately.

This formulation of test weights is due to J. B. Kruskal. Ref. 8, Section VII contains an elegant discussion of test weights excluding inconsistency considerations.

Test weighting has not, as yet, been used with No. 1 ESS data. The process of calculating $5000 \times 4999 \div 2$ correlation coefficients (assuming a diagnosis of 5000 related tests in a diagnosis) is very time consuming. However, results obtained without test weights (i.e., every test given the same weight) appear quite satisfactory.

V. RESULTS AND CONCLUSIONS

5.1 *Dictionary Evaluation Results*

An experiment was conducted to evaluate the effectiveness of the No. 1 ESS exact-match dictionary, which is being used in the field, and the phase and cell dictionaries, which are being implemented. A sample of faults was inserted in the central control at a field No. 1 ESS office located in Chase, Maryland. The sample, which consists of 302 central control faults, was selected by persons who were not involved in the dictionary production project so as to reduce the possibility of bias in the selection. The faults selected were well distributed with respect to their types and locations, and were representative of expected troubles. Each fault was inserted when the office was running under a simulated traffic load and a diagnosis was performed. The corresponding diagnostic printout had three possible outcomes: (*i*) a printout that matched the correct dictionary number in the exact-match dictionary, (*ii*) a printout which did not match the dictionary number in the exact-match dictionary, or (*iii*) a printout indicating all-tests-passed.

TABLE I—EXACT-MATCH DICTIONARY EVALUATION OF FAULTS INSERTED IN NO. 1 ESS OFFICE AT CHASE, MARYLAND

Faults inserted		302	
(*i*)	Found in exact-match dictionary	216	(72.2%)
(*ii*)	Produced diagnostic printout, but not found in exact-match dictionary	58	(19.4%)
(*iii*)	Produced all-tests-passed printout	25	(8.4%)
(*iv*)	Produced invalid data for analysis	3	

Of the 302 faults inserted, there were 216 faults that were successfully located by the exact-match dictionary, 58 faults producing a printout not found in the dictionary, 25 faults producing all-tests-passed printouts, and 3 faults whose test results were either invalid or incomplete for this analysis due to errors made in the fault insertion procedure. The evaluation results for the exact-match dictionary are shown in Table I. Since the diagnostic programs were designed under a pressing time schedule with little opportunity for the feedback of evaluation results, we consider these figures quite gratifying.

Further analysis revealed that out of 25 faults producing all-tests-passed printouts, 5 were faults that could not be detected by programs because of inherent circuit redundancy and 20 were faults that were not detected due to test inadequacy. Out of 58 faults producing printouts which did not match any dictionary numbers in the dictionary, 9 were those diagnostic data had not been simulated during the dictionary production process and 9 were faults whose data were incomplete for the purpose of analysis with phase or cell dictionaries. Thus, only the data of the remaining 40 faults were analyzed to illustrate the feasibility and the effectiveness of phase and cell dictionaries. As shown in Table II, only 2 faults could not be located; all other 38 faults were found in either phase or cell dictionaries. In addition, 80 percent of the faults located by the cell dictionary were isolated to 10 or fewer circuit packages. (For this evaluation, only 5 sections of

TABLE II—PHASE AND CELL DICTIONARY EVALUATION OF FAULTS INSERTED IN NO. 1 ESS OFFICE AT CHASE, MARYLAND

Faults producing inconsistent test results		40
(*i*)	Found in phase ictiodnary	27
(*ii*)	Found in cell dictionary	35
(*iii*)	Found in either phase or cell dictionaries	38
(*iv*)	Found nowhere	2

the multiple cell dictionary were constructed. The partitioning of the diagnostic for each section was 3, 5, 11, 21, and 41 bits, respectively.) This is indeed a significant improvement. However, it must also be cautioned that this sample of faults producing inconsistent test results is too small to draw any meaningful quantitative conclusions. What can be said is that significant improvements over present exact-match techniques can be expected and that the cell dictionary approach may be somewhat superior to the phase dictionary approach.

The exact-match dictionary has been in use in No. 1 ESS office in Succasunna, New Jersey, since May 30, 1965. Its effectiveness has been more or less compatible with our evaluation results. The machine can usually be diagnosed and repaired within twenty minutes when the dictionary look-up procedure is successful. The phase and cell dictionaries will be implemented for the No. 1 ESS office in Beverly Hills, California, which will begin service sometime in the fall of 1966. The detailed field performance of all these dictionaries is not covered in the scope of this paper.

5.2 *Concluding Remarks*

The major advantages of these techniques, as a whole, are that they provide the maintenance craftsman with rapid methods for extracting the information from diagnostic test patterns for the purpose of faults location. The techniques require a very modest amount of machine time and memory. They can be quite effective especially if some care is taken during the fault simulation phase of dictionary production.

Each technique has its limitations, however. The "exact-match" dictionary will not handle inconsistencies. The phase dictionary will be of assistance if at least one phase is consistent, but at the cost of resolution and time consumed while manually searching for fault identities using the majority rule approach. The phase prime dictionary will eliminate the manual search but will work only if the inconsistency is confined to a single phase. The cell dictionary is ineffective when the fault falls into the zero cell.

We think that the results obtained will be fairly typical of what can be expected when implementing a maintenance dictionary approach on a digital machine. The combination of techniques is not perfect but is one of the most powerful for locating faults in real-time systems.

A logical continuation of this work would probably involve:

(*i*) A study of why tests are inconsistent. This would permit a better technique for eliminating inconsistencies from test patterns

when generating phase dictionaries. It would also permit modifications of the measure of "similarity" between fault patterns.

(*ii*) Development of better cell center algorithms for the particular form of cell dictionary described. An "ultimate" cell dictionary is probably an implementation of Kruskal's ideas.[8]

(*iii*) Develop criteria for establishing figures of merit for dictionary techniques which take into account: (*a*) the percentage of real troubles located (as compared to simulated faults), (*b*) the resolution, (*c*) the speed or facility with which dictionary look-ups can be made, and (*d*) the machine time and memory processing requirements.

Some progress has already been made on items (*ii*) and (*iii*) but considerably more is needed.

VI. ACKNOWLEDGMENTS

The authors would especially like to acknowledge the work of J. B. Kruskal whose ideas were the stimulus for much of the work on the cell dictionary. We are also indebted to F. M. Goetz for the initial suggestions on "pseudo-random" mapping methods and the phase prime dictionary. The programming skills of R. E. Archer and J. L. Kodner were invaluable, as well as the comments of R. L. Campbell, R. W. Downing, L. S. Tuomenoksa, and many of our colleagues.

APPENDIX A

No. 1 ESS System Organization and Maintenance Plan

No. 1 ESS is a general purpose electronic telephone switching machine which employs for its control a time-shared multiple-program computer operating in real time.[3] Functionally, the system can be divided into a central processor and a peripheral system (see Fig. 6). The *central processor,* which operates with $5\frac{1}{2}$ microsecond cycle time, provides the data processing facility for telephone, maintenance and administrative functions. It consists of program stores, call stores, and a central control. The *program store,* which is a read-only type of semi-permanent memory, contains the stored program and translation information that are needed to switch calls and provide services as well as maintenance programs. The *call store,* which is a temporary ferrite sheet memory, stores all transient information for processing calls, such as the digits dialed by the subscriber or the busy-idle states of lines and trunks. The *central control* consists mainly of wired logic. Its duty is to coordinate and command all system operations.

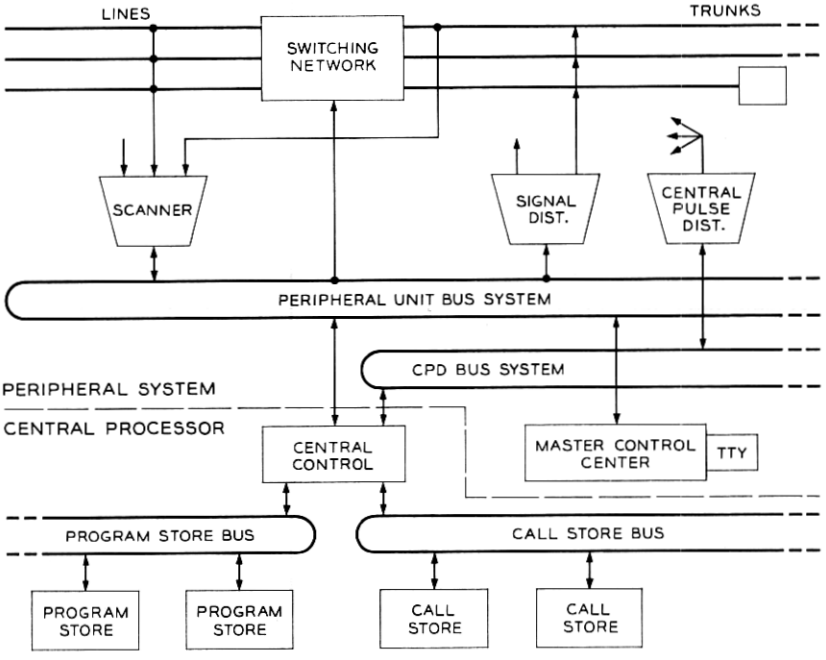


Fig. 6 — No. 1 ESS block diagram.

The *peripheral system* consists of a ferreed switching network, scanners and distributors, and a master control center. The *switching network* provides the connections between lines, trunks, and service circuits (i.e., auxiliary devices such as tone sources, signal receivers, and signal transmitters). The *scanners* are used to collect information from lines, trunks, and points internal to the system. The *distributor* is made up of two units: the *central pulse distributor*, which operates at machine cycle speed, is used for signaling logic circuits, whereas, the *signal distributor* is used for controlling slower devices such as relays in trunk circuits. The various subsystems are interconnected with balanced ac coupled bus systems. The *master control center* includes a teletypewriter for input-output, a panel for manual testing of lines and trunks, and some controls and displays.

The No. 1 ESS was designed for a high degree of maintainability and dependability. These objectives require that telephone service is not interrupted even in the presence of internal component failures.[2] To achieve these objectives, the major units of the central processor are duplicated, and circuit and program facilities are provided for

detecting troubles, locating the faulty subsystems and re-establishing an operational configuration without interrupting telephone service. Once a faulty subsystem is identified, the diagnostic programs are called in to analyze the trouble. These programs are executed (on-line) by the operational configuration; they are segmented and interleaved with the main call processing program to avoid interference with the normal system operation. The diagnostic programs carry out a fixed sequence of tests by observing the normal outputs of the faulty subsystem or by monitoring some special test points strategically embedded within the unit. The combinational testing approach is used to conserve program storage space and to simplify the processing of test results. The pass or fail test data are recorded and processed using the exact-match techniques discussed previously (in Section 1.3 and Appendix B) to produce a compact diagnostic printout on the teletypewriter. The translation of a diagnostic printout into the location of the replaceable faulty circuit package(s) is then accomplished with the aid of a dictionary. The data for the dictionary are generated through hardware simulation, i.e., by actually inserting almost every possible simple hard fault sequentially into the unit and then recording its diagnostic. The general approach of dictionary production is similar to the one on Morris machine[5] and therefore, will not be described here. A sample format of exact-match dictionary entries is illustrated in Fig. 2.

APPENDIX B

Pseudo-Random Number Generation

In No. 1 ESS, the diagnostic test results of each fault is represented by an n-bit binary number where each bit or a sequence of bits designates the pass or fail result of a particular test(s). The number of bits n is usually very large, e.g., $n \approx 5000$. The "hash" technique used to reduce No. 1 ESS diagnostic data to a smaller fixed length number employs a "pseudo-random" function which manipulates an arbitrary and large pattern of test results to produce a number with relatively few digits. The reduction procedure is pseudo-random in the sense that the mapping is deterministic but approximates the process of assigning one truly random number to each fault pattern.

This process is analogous to the problem of selecting numbers at random from an urn. Assume the urn has N distinct numbers. A total of k numbers are selected, one at a time with replacement, from the

urn. The probability P that all these k numbers are distinct is

$$P = \prod_{i=1}^{k-1} \left(1 - \frac{i}{N}\right)$$

$$\therefore \quad P \cong 1 - \frac{\sum_{i=1}^{k-1} i}{N} = 1 - \frac{k(k-1)}{2N}, \quad \text{for} \quad k \ll N.$$

Thus, suppose k denotes the total number of distinct fault patterns and $d = [\log_{10} N]$ denotes the number of decimal digits in the diagnostic printout (the symbol $[x]$ denotes the smallest integer greater than or equal to x). The probability of generating, at least conceptually, *all distinct* d-digit numbers from a pseudo-random number generator can be made arbitrarily large by increasing d. However, in practice the number of digits d in the printout should be kept reasonably small so as to simplify the look-up process and to reduce the dictionary's bulkiness. Hence, for a sample of 10^5 distinct fault patterns, a 12-digit representation would probably suffice, since it yields a probability of 0.995 that all resultant numbers will be distinct.

When the ratio of k to N is not exceptionally small, a few duplicated (and replicated) numbers could result. Thus, it is also necessary to compute the expected number, E_r, of replicated pseudo-random numbers when k fault patterns are assigned random values from a sample space N. Suppose E_e represent the expected number of distinct numbers generated, then the probability that a particular number is selected, at least once, from the urn in k selections is $1 - (1 - 1/N)^k$. This probability is also equal to E_e/N. Thus,

$$E_e = N\left(1 - \left(1 - \frac{1}{N}\right)^k\right).$$

The expected number of replicated numbers becomes,

$$E_r = k - E_e = \sum_{i=1}^{k-1} (-1)^{i+1} C_{i+1}^{k} \left(\frac{1}{N}\right)^i.$$

Hence, for a sample of 10^5 distinct fault patterns, the expected number of replicated numbers in a 6-digit representation is about 4837 whereas the expected number of replicated numbers in a 12-digit representation is less than one.

An experiment was performed to verify the hypothesis that this method of diagnostic data reduction is analogous to the problem of selecting numbers at random from an urn. A sample of 964 fault patterns each consisting of approximately 1000 bits of test results was

used. Initially, the reduction process just used the addition and rotation instructions to produce a 6-digit number, nine duplicated resulted. On the subsequent attempts, shift and other peculiar instructions were added to better scramble the data; the number of duplicates finally decreased to zero. From the urn analog, the expected number of duplicates in assigning a 6-digit pseudo-random number to 964 distinct patterns is 0.5, and the probability of no duplication is 0.61.

The experiment demonstrates the feasibility of number generation schemes. An effective method can readily reduce each pattern of a collection of diagnostics to a smaller fixed-length number with very little loss in resolution. Since the greatest possible total number of fault patterns for any No. 1 ESS subsystem is about 10^5, the diagnostic printout uses a 12-digit representation. As mentioned earlier, this representation is amall enough so that the dictionary look-up process is easy, yet large enough so that the probability of all generated numbers being distinct is quite high and the expected number of duplicates is very small.

Basically, each fault pattern undergoes two stages of reduction process. In the first stage, each binary fault pattern of n bits is ANDed (bit-wise) with each member of a set of m preselected reference vectors R_1, R_2, $\cdots$ R_m, and the resultant "bit-sums" S_1, S_2, $\cdots$, S_m are collected ($m \ll n$). That is, suppose the binary fault pattern of fault F is $T_1T_2 \cdots T_n(T_i = 1 \text{ or } 0)$, and the pattern of reference vector R_i is $r_1^i r_2^i \cdots r_n^i (r_i^i = 1 \text{ or } 0)$. Then the bit-wise ANDing operation will generate a "bit-sum" S_i where,

$$S_i = \sum_{i=1}^{n} T_i \cdot r_i^i$$

and $j = 1, 2, \cdots, m$. To further reduce the size of the fault pattern, each set of bit-sums S_1, S_2, $\cdots$, S_m undergoes three independent stages of "data scrambling" manipulation, each resulting in a 4-digit number. Each stage is a pseudo-random number generation procedure based on the shift, rotation, and addition orders.

The final diagnostic printout is, therefore, a 12-digit number, formed by a concatenation of three 4-digit numbers. Fig. 7 shows the general program flow of the final reduction process.

The final three stage reduction process used was evolved through experimentation. The resolution of the dictionary so generated was quite high, high enough so that most entries in the dictionary associate with only four or fewer circuit packages. For example, in the case of the central pulse distributor,[7] which has 3312 simulated faults and whose

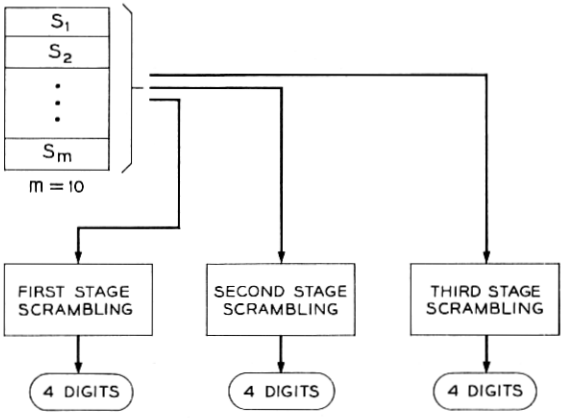


Fig. 7 — Data reduction.

TABLE III — DICTIONARY RESOLUTION STUDY
(CENTRAL PULSE DISTRIBUTOR)

i	$N(i)$		
	1st stage	1st & 2nd stage	All three stages
1	436	910	924
2	300	653	663
3	101	68	67
4	65	26	26
5	18	4	3
6	49	12	10
7	5	4	2
8	54	1	1
9	8	1	1
10	7	1	1
11	3	1	1
12	3	0	0
13	0	0	0
14	1	0	0
15	0	0	0
16	1	1	1
17 AND UP	0	0	0

$N(i)$ = Number of dictionary numbers having i associated packages.

fault pattern has 1150 bits, 15 reference vectors were used. Initially, only the *first stage* of the final reduction process (rotation and addition) was used to produce a 4-digit number. In the resultant dictionary only 59 percent of the entries were associated with four or fewer circuit packages and the mean was 2.56 packages per entry (see Table III). As the *second stage* was added to better scramble the data, 90 percent of the entries in the dictionary associated with four or fewer circuit packages and the mean was improved to 1.62 packages per entry. The *third stage* was added and the resultant dictionary had 95 percent of its entries associating with four or fewer circuit packages and a mean of 1.58 packages per entry. Further experimentation with the addition of a fourth stage reduction process did not provide significant improvement in resolvability. Thus, a three-stage reduction process was adopted. The dictionary is rather compact; it has only 52 (8-$\frac{1}{2}$ by 11) pages (for the central pulse distributor). A sample of the format printout is shown in Fig. 2. Sample evaluation results of this type of dictionary are discussed in Section V.

REFERENCES

1. Doyle, R. H., Meyer, R. A., and Pedowtiz, R. P., Automatic Failure Recovery in Digital Data Processing System, IBM J. Res. Dev., *3,* January, 1959, pp. 2–12.
2. Downing, R. W., Nowak, J. S., and Tuomenoksa, L. S., No. 1 ESS Maintenance Plan, B.S.T.J., *43,* September, 1964, pp. 1961–2020.
3. Keister, W., Ketchledge, R. W., and Vaughan, H. E., No. 1 ESS: System Organization and Objectives, B.S.T.J., *43,* September, 1964, pp. 1831–1844.
4. Brule, J. D., Johnson, R. A., and Kletsky, E., Diagnosis of Equipment Failures, IRE Trans. Rel. Qual. Contr., *RQC-9,* April, 1960, pp. 23–24.
5. Tsiang, S. H. and Ulrich, W., Automatic Trouble Diagnosis of Complex Logic Circuits, B.S.T.J., *41,* July, 1962, pp. 1177–1200.
6. McIlroy, M. D., A Variant Method of File Searching, Commun. ACM, March, 1963, p. 101.
7. Freimanis, L., Guercio, A. M., and May, H. F., No. 1 ESS Scanner, Signal Distributor and Central Pulse Distributor, B.S.T.J., *43,* September, 1964, pp. 2255–2282.
8. Kruskal, J. B. and Hart, R. E., A Geometric Interpretation of Diagnostic Data from a Digital Machine, Based on a Study of the Morris, Illinois Electronic Central Office, B.S.T.J., *45,* October, 1966, pp. 1299–1338.