AN APPLICATION OF FUNCTIONAL DEPENDENCIES TO THE TOPOLOGICAL ANALYSIS OF PROTECTION SCHEMES

L. Jenkins, Member-IEEE
H.P. Khincha* Senior Member-IEEE
S. Shivakumart
P.K. Dash$

+Department of Electrical Engineering, Indian Institute of Science, Bangalore 560 012, India
+Department of Computer Science, University of Central Florida, Orlando, FL 32816, USA
$Department of Electrical Engg, Regional Engineering College, Rourkela, India

Abstract - The task of systematically choosing the settings of the relays in a protection scheme is facilitated by the identification of a suitable break point set and the corresponding relative sequence matrix. The concept of functional dependency is used as a means of developing algorithms for the accomplishment of these sub-tasks. These algorithms are seen to be computationally more efficient than the existing algorithms of the graph theoretic schemes.

INTROOUCTION

Protection engineers select the settings of the relays in a protection scheme on the basis of certain assumptions with regard to the power system topology and the load conditions. Whenever these assumptions lose their validity, such as after a drastic change in the load values, or the removal of a line from service, the relay settings may no longer be appropriate. Hence, it is desirable that these settings be reviewed before any major alterations in the power network are brought about. However, since the manual design of the relaying scheme is a tedious and time-consuming task, the relay settings are not modified as often as is warranted, and there is a consequent degradation in the performance of the protection scheme. The availability of computer-aided design tools has led to efforts in developing automated relay coordination techniques, which would take the tedium out of the design of protection schemes, thereby encouraging more frequent adjustments in the relay settings, with a consequent improvement in performance. Recent work in this direction is reported by Damborg et al [1], Rao and Rao [2], and Damborg and Venkata [3]. In this paper, we develop more efficient technique for the automatic coordination of directional overcurrent and distance relays.

The earlier work in the field of automatic relay coordination employed graph theoretic schemes [1,2,4,5]. Although these schemes do achieve their objective, they require a computation time which is an exponential function of the number of relays, and hence the computation cost is prohibitive when the number of relays is large [2]. In this paper, the concept of functional dependency will be used as a means for the development of algorithms which complete the same tasks in a computational time which is a polynomial function of the number of relays.

90 IC 559-5 PWRD A paper recommended and approved by the IEEE Power System Relaying Committee of the IEEE Power Engineering Society for presentation at the 1990 IEEE/PES International Power Meeting-India, New Delhi, India, October 28 - November 1, 1990. Manuscript submitted March 30, 1989; made available for printing July 13, 1990.

Functional dependency is a concept which arises in data base systems [6,7,8]. In the context of power system protection, the setting of each directional overcurrent or distance relay has a functional dependency on all the primary relays for which this relay serves as a backup relay. Hence, all the constraints on the relay settings can be expressed through a set of functional dependencies. This set of functional dependencies contains all the constraint information as applied to the relay settings, and it is possible to develop algorithms which use these functional dependencies as a means of facilitating automatic coordination. These algorithms will be described in the subsequent sections of this paper.

THE RELAY COORDINATION PROBLEM

The coordination of directional overcurrent relays involves a choice of relay settings such that for every fault in the system, there is a specified minimum coordination interval or time delay between the operation of the primary relay and that of the backup relay; this interval ensures that the backup relay operates only when the primary relay fails to perform its assigned task. Similarly, in distance relay coordination, the time delay zones 2 and 3 of the primary and backup relays must not be allowed to overlap unless there is a suitable delay in their time settings. Hence, when relays are being set, it is necessary to check all primary-backup relay pairs in the protection scheme.

A major consideration is that the same relay may serve as the primary relay in some primary-backup pairs, and as the secondary relay in some other pairs. Hence the setting of each relay must be chosen such that it operates ahead of some other relay for some faults, as well as operating after some other relay for some other faults. To visualize the problem, one could consider a set of relays rl, r2, ..., rm as belonging to a loop, with the corresponding set of primary-backup pairs being (rl, r2), (r2, r3), ..., (r(m-1), rm), (rm, rl). After an initial choice of the setting of the relay rl has been made, one would then set the relay r2 on the basis of the constraint given by the primary-backup pair (rl, r2). Similarly, the choices of settings of the relays r3, r4, ..., rm would be made on the basis of the constraints that correspond to the pairs (r2, r3), (r3, r4), ..., (r(m-1),rm) respectively. It would then be necessary to check whether the settings of the relays rm and rl satisfy the timing constraint that corresponds to the primary-backup pair (rm, rl). If they do not, then another iteration of choosing the settings of the relays in the loop must be undertaken. The coordination problem is a complicated one, because each relay in any loop can also belong to other loops, and its setting must satisfy a large number of constraints simultaneously. In the absence of a systematic approach to the problem, one would have difficulty in deciding on a suitable set of relays as a starting point in the coordination of the different relay loops, and many such choices may be unsuitable from the point of view of the rapid completion of the task of choosing the relay settings.

Relay coordination may be systematically carried out through the identification of a break point set (BPS) and a relative sequence matrix (RSM). These two entities are now defined.

Definition 1. Let A be a subset of the set of relays, S. The relative sequence matrix of A is a column vector, each element of which is a disjoint subset of S, and is chosen as follows:
1. The first element is the set A
2. The second element, B, is chosen from the set (S-A) according to the following: a relay belongs to B if and only if it is a backup relay only to primary relays which belong to A, and is not a backup relay to any relay in (S-A).
3. Successive elements of the RSM are obtained by repeating 2, with the set (B+A) replacing A. The RSM is complete when no more elements can be so obtained.

The termination of step 3 takes place either because all relays have entered the RSM, or each relay that is not in the RSM serves as the backup to at least one other relay that is not in the RSM. In the former case, we say that A is a BPS.

Definition 2. Let A be a subset of the set of relays S. Then A is a break point set if its RSM contains every relay of S.

We can now see the significance of a BPS in relay coordination. If A is a BPS, then we begin coordination by assigning an initial choice of relay settings to the relays of A. Using these settings, we invoke primary-backup relationships to choose the settings of the relays which belong to the second element of the RSM. Similarly relays of each successive element are chosen on the basis of those of the previous elements. Once all the relays have been so chosen, it is necessary to check whether the initial choice of the settings of A is in agreement with the settings of every other relay with which they are associated as backups. If so, relay coordination has been achieved: otherwise a suitable adjustment is made to the initial settings of the relays of A, and a second choice is made of the other settings, through repeated use of the RSM, as before. Iterations are repeated till the relays of A are found to satisfy their primary-backup relationship.

If we use a set of relays which is not a BPS as the starting point of coordination, the RSM will not contain all the relays, and we cannot systematically coordinate the relays which are absent from the RSM. This demonstrates the significance of the BPS and the RSM.

The choice of BPS is not unique, and any set of relays which contains a BPS is itself a BPS. Since the settings of the BPS relays are initially chosen arbitrarily, the average number of iterations required for coordination increases with the size of the BPS. Hence, a BPS with less members is preferred. We now define the concepts of minimum and minimal BPS.

A BPS, U, of a protection network is a minimum BPS if there does not exist any BPS, V, of the network for which $|V| < |U|$, where $|U|$ denotes the number of elements, or cardinality, of the BPS U.

The difficulty in applying the concept of minimum BPS is that the identification of such a BPS is an NP-complete problem [9], and cannot be carried out in a polynomial time period. Hence, it is not feasible to develop algorithms which can find a minimum BPS. A more relevant concept is that of a minimal BPS, which will now be defined.

A BPS, U, of a protection network is a minimal BPS if there does not exist any proper subset of U, say V, such that V is a BPS.

The minimal BPS identification problem can be solved in finite time. In fact, all the graph theoretic algorithms find minimal BPS in time periods that are exponential functions of the number of relays, since in graphs the number of loops increases exponentially with the number of edges. In this paper, the concept of functional dependency is applied to the development of an algorithm which can find a minimal BPS in a time interval that is a polynomial function of the number of relays, and hence has an order of magnitude improvement over the graph theoretic schemes.

FUNCTIONAL DEPENDENCY

Functional dependency is a concept which has been applied to data base systems [8]. In a collection of data items, each item has a number of fields, which are known as its attributes. Functional dependency (FD) is a relation on these attributes, which is expressed in the form

$$F: (Ai, Aj, \ldots, Ak) \rightarrow Av$$

Here the implications is that the values of the attributes Ai, Aj, ..., Ak together determine the value of the attribute Av. For instance, if the fields in a population table are city, country, latitude, longitude and population, then the attribute country functionally depends on the attribute city. Similarly the attribute city functionally depends on the attributes latitude and longitude, since each combination of latitude and longitude values together determine the appropriate city.

Although the concept of functional dependency can produce very powerful general results, we will consider it in a more restricted sense, as applying directly to the study of relay coordination. In this context, the settings of the relays are viewed as the attributes of the protection scheme, and the functional dependencies are generated from the primary-backup pairs. For instance, if relay rv is the backup relay in exactly three such pairs, namely (ri, rv), (rj, rv), (rk, rv), then the setting of rv can be determined on the basis of the settings of ri, rj and rk, and one can write the FD.

$$f: (ri, rj, rk) \rightarrow rv$$

Obviously, if the system has n relays, with each relay serving as a backup to some other relays, then one will generate exactly n FD's by this approach. Example 1 is illustrative.

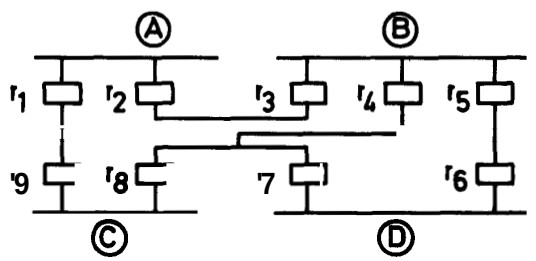


Fig.1 AN EXAMPLE SYSTEM

Example 1: The 9-relay protection scheme shown in Fig. 1 has the following set of primary backup pairs: (r1, r3), (r2, r9), (r3, r6), r3,r7), (r3,r8), (r4,r2), (r4,r6), (r5,r2), (r5,r7), (r5,r8), (r6,r4), (r6,r8), (r7,r5), (r8,r1), (r9,r4), (r9,r7). The corresponding set of FD's is given below:

f1 : (r8) → r1
f2 : (r4,r5) → r2
f3 : (r1) → r3
f4 : (r6,r9) → r4
r5 : (r7) → r5
r6 : (r3,r4) → r6
r7 : (r3,r5,r9) → r7
r8 : (r3,r5,r6) → r8
f9 : (r2) → r9.

IDENTIFICATION OF A MINIMAL BPS

As was explained earlier, the choice of the initial settings of the BPS relays is used to select the settings of the remaining relays of the protection scheme. Hence, in order that a set of relays form a BPS, the settings of every other relay is required to be functionally dependent on the settings of this set of relays. We will use the term 'relay rv is functionally dependent on the set of relays A' to denote that the setting of relay rv is functionally dependent on the settings of the relays of the set A.

Algorithm 1 will now be developed, to test whether any given subset of A, of the set of n relays, S, is a BPS. The algorithm examines the corresponding set, F, of n FD's of the relaying system, to determine whether every relay is functionally dependent on the set of relays A. Recall that the FD's are of the form

fp : (ri, rj, ..., rk) → rp p = 1,2,...,n

If rp is a member of A, then rp is functionally dependent on itself, and hence on the set of relays A. Therefore, the algorithm eliminates the corresponding FD from F, without the need of further processing. The remaining FD's, which correspond to those rp that belong to (S-A), can be split into two categories, on the basis of whether all their left hand side variables (ri,rj,...,rk) belong to A. If they do, then by definition rp is functionally dependent on A. If not, then rp will functionally depend on A if and only if each of the left hand side variables (ri, rj, ..., rk) of fp are either members of A or are functionally dependent on the relays of A. This follows from the reasoning that if a relay rp is functionally dependent on a relay rq, which in turn is functionally dependent on the relays of A, then the setting of the relays of A would enable one to select the setting of the relay rq, and follow this with the setting of the relay rp. This argument is referred to as pseudo-transivity, in Armstrong's axioms of functional dependency [10].

In Algorithm 1, the set $\hat{A}$ is assigned each relay that has been shown to be functionally dependent on A, and the set $\hat{F}$ contains those FD's which have not yet been eliminated from the consideration. Initially, $\hat{A}$ = A and $\hat{F}$ = F; then the FD's which correspond to members of A are removed from $\hat{F}$, since they play no further role in the study. In turn, each member of A is removed from the left hand side set (ri,rj, ...,rk) of every FD of $\hat{F}$ in which it occurs; it is then removed from $\hat{A}$. If the removal of a relay rq from the left hand side set of fp renders the set empty, then it is implied that rp is functionally dependent on A; hence, rp is added to $\hat{A}$, and fp removed from $\hat{F}$, before any attempt is made to carry out any more removals of rq. The process of deletion of relays from the FD's continues until either $\hat{A}$ or $\hat{F}$ is empty. If $\hat{F}$ is empty, it implies that all relays are functionally dependent on the relays of A, and hence A is a BPS. If $\hat{A}$ is empty, then no other relay can be shown to have a functional dependency on the relays of A; hence, the presence of FD's in $\hat{F}$ indicates that the corresponding relays are not functionally dependent on A, which is not a BPS.

Algorithm 1
Step 1. Set $\hat{A}$ = A, $\hat{F}$ = F
Step 2. Delete from P a l l FD's which correspond to members of A.
Step 3. Choose a member of $\hat{A}$, say rq. Delete rq from the left hand side set of every FD of $\hat{F}$ in which it occurs. If this deletion renders the set empty, remove the FD from $\hat{F}$, and add its right hand side relay to the set $\hat{A}$, before attempting the next deletion.
Step 4. Remove rq from $\hat{A}$. If $\hat{F}$ = Ø, then A is a BPS. Go to step 7.
Step 5. If $\hat{A} \neq 0$, go to step 3.
Step 6. A is not a BPS
Step 7. Terminate.

We illustrate Algorithm 1 through Example 2.

Example 2. Determine whether the following are BPS for the relaying scheme of Example 1 :- A1 = (r3,r4,r5), A2 = (r1,r2,r4)

Consider first Al. We set $\hat{A}$ = (r3, r4, r5). After deletion of f3, f4 and f5 from the set F of example 1, we get the set $\hat{F}$ as

f1 : r8) → r1
f2 : r4, r5) → r2
f6 : r3, r4) → r6
f7 : r3, r5, r9) → r7
f8 : r3, r5, r6) → r8
f9 : r2) → r9

Deleting successively r3, r4, r5 from the left hand side sets of these FD's, we eliminate f2 and f6 from $\hat{F}$. We get $\hat{A}$ = (r2, r6), and $\hat{F}$ as given by

f1 : r8) → r1
f7 : r9) → r7
f8 : r6) → r8
f9 : r2) → r9

Deleting successively r2 and r6, we eliminate f8 and f9. Now $\hat{A}$ = (r8, r9), and $\hat{F}$ is given by

f1 : (r8) → r1
f7 : (r9) → r7

Now deleting r8 and r9 gives $\hat{F}$ = 0, thereby establishing that A1 is a BPS.

Next take A2 = (r1, r2, r4). After the deletion of f1, f2, f4, the set $\hat{F}$ is given by

f3 : (r1) → r3
f5 : (r7) → r5
f6 : (r3, r4) → r6
f7 : (r3, r5, r9) → r7
f8 : (r3, r5, r6) → r8
f9 : (r2) → r9.

Deleting successively r1, r2, r4 from $\hat{F}$, one eliminates f3,f9. Hence, $\hat{A}$ = (r3,r9), and $\hat{F}$ is given by

f5 : (r7) → r5
f6 : (r3) → r6
f7 : (r3, r5, r9) → r7
f8 : (r3, r5, r6) → r8

Now, eliminating r3 and r9 removes r6 from $\hat{F}$, so $\hat{A}$ = r6 and $\hat{F}$ is given by

f5 : (r7) → r5
f7 : (r5) → r7
f8 : (r5, r6) → r8

The removal of r6 from the left hand side of these FD's does not eliminate any of them. One gets $\hat{A}$ = ∅, and? as given by

f5 : (r7) → r5
f7 : (r5) → r7
f8 : (r5) → r8

Since? is non-empty, if follows that A2 is not a BPS.

Algorithm 1 is invoked in Algorithm 2, which enables one to find a minimal BPS. In Algorithm 2, one initially chooses a known BPS, such as the entire set of relays, S; a relay is then removed from this BPS, and the remaining set of relays is tested through Algorithm 1, to check whether it is a BPS. If so, this new BPS becomes the starting point, and the process of eliminating an element from it is repeated. Otherwise, the relay that had been removed is deemed to be an essential member of the BPS; it is restored, and then an attempt is made to remove another element from the restored BPS. It is to be noted that if a collection of relays A is not a BPS, no subset of A can be a BPS. Hence, once a relay has been considered for elimination and then restored, it is never again considered. Algorithm 2 terminates after one attempt has been made to remove each one of the relays of the original BPS; the resultant BPS is a minimal BPS, since no relay can be removed from it, in order to produce a subset that is also a BPS.

Algorithm 2.

Step 1. Choose A as the set of relays, S
Step 2. Remove from A a relay which has not been removed earlier.
Step 3. Use Algorithm 1, to check whether A is a BPS. If so, go to step 5.
Step 4. Restore to A the relay which had been removed in step 2.
Step 5. If A contains a relay which had not been removed earlier, return to step 2.
Step 6. Terminate.

Although Algorithm 2 produces a minimal BPS, there is no guarantee that this BPS will be a minimum BPS, since the composition and the size of the minimal BPS that is obtained depend on the order in which relays are targeted for removal. Initially, when the BPS is the entire set of relays, one can usually remove any relay, and still have a BPS, but after a few relays are eliminated, some of the remaining ones become essential. In the absence of any specific information, one has to arbitrarily choose an order of removal of relays in step 2 of the algorithm, with the knowledge that the resultant BPS may not be a minimum BPS, though it is guaranteed to be a minimal BPS. This is a feature that the functional dependency approach shares with the graph theoretic techniques [2], and is to be expected, since the selection of a minimum BPS is an intractable problem, and cannot be solved in a realistic time interval [9].

In Example 3, which illustrates Algorithm 2, the relays will be selected for elimination in descending order, beginning with r9.

Example 3. Find a minimal BPS for the relaying scheme of Example 1.

We begin by selecting A as (r1, r2, ..., r9), and then attempt to remove r9. To check whether A = (r1, r2, ..., r8) is a BPS, through Algorithm 1, we choose $\hat{A}$ = (r1, r2, ..., r8) and $\hat{F}$ is given by

f9 : (r2) → r9

Since the left hand side relay r2 is a member of $\hat{A}$, r9 is functionally dependent on the relays of A, and the set A is a BPS.

Similarly, when r8 is now deleted, to give A = (r1, r2, ..., r7), the set $\hat{F}$ consists of the two FD's

f8 : (r3, r5, r6) → r8
f9 : (r2) → r9

It is obvious that r8 and r9 are functionally dependent on the relays of A, which therefore is a BPS.

When r7 is removed, the set A = (r1, r2, ..., r6), and F is given by

f7 : (r3, r5, r9) → r7
f8 : (r3, r5, r6) → r8
f9 : (r2) → r9

It is clear that r8 and r9 are functionally dependent on the relays of A. Since the left of f7 consists of r9, along with members of A, it follows that r7 also is functionally dependent on A, which is a BPS. We can now eliminate r6, and similarly establish that (r1, r2, r5) is a BPS. Next, by removing r5, we get A = (r1, r2, ..., r4), and $\hat{F}$ is given by

f5 : (r7) → r5
f6 : (r3, r4) → r6
f7 : (r3, r5, r9) → r7
f8 : (r3, r5, r6) → r8
f9 : (r2) → r9

By removing r1, r2, r3, r4 from these FD's, we eliminate f6 and f9, and hence $\hat{A}$ = (r6, r9), while $\hat{F}$ reduces to

f5 : (r7) → r5
f7 : (r5, r9) → r7
f8 : (r5, r6) → r8

Next, r6 and r9 are removed, to give $\hat{A}$ = 0, and $\hat{F}$ as given by

f5 : (r7) → r5
f7 : (r5) → r7
f8 : (r5) → r8

Since A = 0 and F ≠ ∅, it follows that the set (r1, r2, r3, r4) is not a BPS, and hence r5 is restored. Next r4 is removed, to get the set A = (r1, r2, r3, r5). The application of Algorithm 1 establishes that this set is not a BPS, so r4 is restored, and r3 is eliminated, to give A = (r1, r2, r4, r5), with $\hat{F}$ as given by

f3 : (r1) → r3
f6 : (r3, r4) → r6
f7 : (r3, r5, r9) → r7
f8 : (r3, r5, r6) → r8
f9 : (r2) → r9

The removal of r1, r2, r4, r5 from these FD's eliminates f3 and f9, so $\hat{A}$ = (r3, r9), and $\hat{F}$ is given by

f6 : (r3) → r6
f7 : (r3, r9) → r7
f8 : (r3, r6) → r8

$$\begin{bmatrix} r1, r4\ r5 \\ r2, r3, \\ r6, r9 \\ r7, r8 \end{bmatrix}$$

It is to be noted that, unlike the technique of [1], this method of identifying the RSM does not explicitly utilize the listing of primary-backup pairs, but instead one obtains this same information directly from the set of FD's, F.

COMPUTATIONAL COMPLEXITY

We first consider Algorithm 1, which is involved repeatedly in the execution of Algorithm 2. In Algorithm 1, the major computational burden is the comparison of the relay rp, that is currently being deleted, with the set of relays on the left hand side of each FD in F. It is reasonable to assume that the relay entries occur randomly in the FD's, as well as in the order in which they are selected for removal. Hence, on an average each relay entry in the FD's takes part in $(n+1)/2$ comparisons before it is deleted. Hence, if the number of such entries is N, the total number of comparisons is $N.(n+1)/2$.

The number N is given by the product of the number of FD's, n, and the average number, k, of relays on the left hand side of each FD. The quantity k is the average number of primary relays for which each relay serves as a backup relay; it can be taken to be a constant over an entire class of relaying schemes.

Hence the number of comparisons is given by $k.n.(n+1)/2$, and the time complexity is of order n^2.

Algorithm 2 involves the removal of each relay in turn, followed by the execution of Algorithm 1, and perhaps the restoration of the removed relay. The major computational burden is the invocation of Algorithm 1, which takes place n times. Hence the identification of a minimal BPS involves a time complexity of n^3.

Algorithm 3 is very similar to Algorithm 1, in that the major computational effort consists of comparison and deletion of each relay entry on the left hand side of the functional dependencies. Similarly, it has a time complexity of n^2.

Since the computation of a minimal BPS requires a time complexity of n^3, and the generation of the RMS requires n^2, the functional dependency approach to relay coordination has a polynomial time complexity. This compares favourably with the graph theoretic schemes, which have an exponential time complexity [2]. In typical protection systems, the value of n is large, and the saving in computation time is significant.

OTHER ASPECTS OF FUNCTIONAL DEPENDENCY

While the most significant gain from the use of functional dependency in relay coordination is its polynomial time complexity, it has other benefits, which will now be described.

(i) Although there is no means of ensuring that the minimal BPS which is identified through Algorithm 2 will also be a minimum BPS, it is possible to modify the algorithm, so as to have a high probability that it identifies a BPS which has comparatively few members; this can be referred to as a near-minimum BPS. The modification is that one uses a suitable criterion to select the order in which the relays are chosen for elimination, since this ordering has a direct bearing on which relays comprise the minimal BPS. Instead of arbitrarily selecting the relays, for instance in descending order of index, as was the case in Example 3, one selects them on the basis of the number of occurrences of their attributes on the left hand sides of the FD's. While an analytic characterization is difficult, heuristically one has a good chance of obtaining a near-minimum BPS if the retained relays are such that the settings of many other relays are dependent on their settings, i.e., if these relays occur on the left hand side of many FD's. Hence, one maintains a count of the number of left hand side occurrences of each relay in F, and selects these relays for elimination in order of increasing count, beginning with the smallest value. For instance, for the FD's of Example 1, the order would be first the members from (r1, r2, r7, r8), followed by those from (r4, r6, r9), followed by (r3, r5). This involves only a minor change in Algorithm 2, and significantly improves its overall performance.

(ii) Quite often, the protection engineer has certain preferences with regard to which relays to retain in the BPS. This arises from operational considerations, and is a reflection on the relative advantage of making an initial choice of the setting of one relay, rather than beginning with some other one. This preference can easily be met by allocating higher priority to the relays which one prefers to retain in the minimal BPS, and then modifying Algorithm 2, so that one considers attributes for elimination in the order of increasing priority ranking of the corresponding relays. Such preferential selection of relays is not possible in graph theoretic schemes. Hence, the functional dependency approach offers considerable flexibility to the protection design engineer.

(iii) The functional dependency approach is versatile enough to handle special protection configurations, without any need for a modification in the algorithms. Suppose, for instance, one has radial lines in the network. Since these lines are not part of a loop, the graph theoretic schemes must treat such a network as a special case, and care must be taken to handle it. On the other hand, once the FD's that correspond to the radial lines have been written, the algorithms of the functional dependency technique can handle these FD's in exactly the same manner as they handle any other FD's. Similarly, if the directionality of a relay of Example 1 is reversed, its constraints cannot be captured by graph theoretic schemes, because it does not get into either the clockwise or the counterclockwise loops. However, the functional dependency algorithms can be used in this case, after the appropriate FD's are written.

CONCLUSIONS

The concept of functional dependency has been applied to the problem of relay coordination in protection systems. An algorithm has been developed for the identification of a minimal break point set of relays of a protection topology. This algorithm is an improvement over existing algorithms, in that it identifies a minimal BPS within a time period that is a polynomial function of the number of relays, while the earlier algorithms had exponential time behaviour. In the case of large protection schemes, the saving in computation costs is considerable. An algorithm has been developed for the selection of a relative sequence matrix; this algorithm also has polynomial time

Next r3 and r9 are removed from $\hat{F}$, thereby eliminating f6 and f7, to give $\hat{A}$ = (r6, r7), and $\hat{F}$ as follows:

f8 : (r6) → r8

The removal of r6 from this FD gives $\hat{F} = \emptyset$, thereby establishing that (r1, r2, r4, r5) is a BPS. From this BPS, the relay r2 is removed; it can be shown that (r1, r4, r5) is a BPS. Now, r1 is removed, and Algorithm 1 is applied to the set (r4, r5). We examine the set $\hat{F}$ of seven FD's

f1 : (r8) → r1
f2 : (r4, r5) → r2
f3 : (r1) → r3
f6 : (r3, r4) → r6
f7 : (r3, r5, r9) → r7
f8 : (r3, r5, r6) → r8
f9 : (r2) → r9

Deleting r4 and r5 from these FD's, we eliminate f2, to get $\hat{A}$ = (r2), and $\hat{F}$ as the set

f1 : (r8) → r1
f3 : (r1) → r3
f6 : (r3) → r6
f7 : (r3, r9) → r7
f8 : (r3, r6) → r8
f9 : (r2) → r9

Deleting r2 eliminates f9, so $\hat{A}$ = (r9). However, the deletion of r9 from the FD's does not eliminate any more FD's, so Algorithm 1 terminates with $\hat{F} \neq \emptyset$, and it is concluded that (r4, r5) is not a BPS. Hence, r1 is restored. Since an attempt has already been made to eliminate each of the relays r1, r4 and r5, Algorithm 2 terminates, and (r1, r4, r5) is a minimal BPS of the relaying scheme.

SELECTION OF RELATIVE SEQUENCE MATRIX

As explained earlier, the RSM of any BPS, A, is a column vector, each element of which is a subset of the set of relays, S. The first element of this vector is the set A, and the second element is the set of relays B, with every member of B being a relay whose setting is specified directly in terms of the settings of the relays of A alone. Subsequent elements of the RSM are similarly specified, with the settings of the relays of each element being specified directly in terms of the settings of the relays in the elements which precede it. Since A is a BPS, every relay of S belongs to exactly one element of the RSM.

Algorithm 3, which obtains the RSM of any BPS, A, is very similar to Algorithm 1. Both these algorithms extract information from the set of FD's, F. While Algorithm 1 checks whether every relay is functionally dependent only on relays that are functionally dependent on the relays of some set A, Algorithm 3 determines the order in which relays may be established to be functionally dependent on relays of some BPS, A.

Algorithm 3 initially allocates the relays of the BPS, A, to the first element of the RSM; the FD's which correspond to these relays are no longer required, and are deleted; let $\hat{F}$ be the set of FD's which remain. Next, the relays of A are deleted from the left hand side of every FD of $\hat{F}$ in which they occur. If any relay rp belongs to the second element of the RSM, then its setting is directly dependent on only the relays of A, and all the relays on the left hand side of the FD fp will be deleted. Therefore the second element of the RSM is selected as the set of relays whose FD's have all their left hand side entries deleted; these FD's are discarded from $\hat{F}$. In the next iteration, the relays whose FD's have been deleted from $\hat{F}$ are removed from the remaining FD's, and the relays whose FD's have all their left hand side entries so deleted are included in the third element of the RSM. This operation is repeated until all relays have entered the RSM.

Algorithm 3

Step 1. Set K = 1, $\hat{F} = F$, $\hat{A} = A$. Allocate $\hat{A}$ to the first element of the RSM.
Step 2. Increment K. Delete from the left hand side of each FD in $\hat{F}$ the relays belonging to $\hat{A}$.
Step 3. Choose $\hat{A}$ as the set of relays in whose FD's in $\hat{F}$ all the left hand side entries have been deleted. Remove these FD's from $\hat{F}$.
Step 4. Allocate $\hat{A}$ to the Kth element of the RSM.
Step 5. If $\hat{F} \neq \emptyset$, return to step 2.
Step 6. Terminate.

We illustrate Algorithm 3 through Example 4.

Example 4. Find the RSM from the BPS (r1,r4,r5) which was obtained in Example 3.

The set $\hat{F}$ is obtained by deleting the three FD's f1, f4 and f5 of Example 1, to give $\hat{F}$ as

f2 : (r4,r5) → r2
f3 : (r1) → r3
f6 : (r3,r4) → r6
f7 : (r3,r5,r9) → r7
r8 : (r3,r5,r6) → r8
f9 : (r2) → r9

The set $\hat{A}$ is given by (r1,r4,r5). Deleting these relays from F gives

f2 : ( ) → r2
f3 : ( ) → r3
f6 : (r3) → r6
f7 : (r3,r9) → r7
r8 : (r3,r6) → r8
r9 : (r2) → r9

Since f2 and f3 have their left sides empty, the second element of the RSM is (r2,r3), and $\hat{F}$ reduces to

f6 : (r3) → r6
f7 : (r3,r9) → r7
f8 : (r3,r6) → r8
f9 : (r2) → r9

Deleting r2 and r3 gives

f6 : ( ) → r6
f7 : (r9) → r7
f8 : (r6) → r8
f9 : ( ) → r9

Hence the third element of the RSM is (r6,r9), and $\hat{F}$ reduces to

f7 : (r9) → r7
f8 : (r6) → r8

Deleting r6 and r9 from these FD's reduces them to

f7 : ( ) → r7
f8 : ( ) → r8

The fourth element of the RSM is (r7, r8). Since, at this stage, the last two FD's have been removed, the algorithm terminates. The RSM is given by

complexity. The functional dependency approach is seen to be more flexible and more powerful than the graph theoretic schemes, and holds out much promise for the development of efficient computer-aided design tools for the protection engineer.

REFERENCES

[1] M.J. Damborg, R. Ramaswami, S.S. Venkata, J.M. Postforoosh, "Computer Aided Transmission Protection System Design", IEEE Transactions on PAS, vol. 103, no. 1, pp. 51-59, January 1984.

[2] V.V. Bapeswara Rao and K. Sankara Rao, "Computer Aided Coordination of Directional Relays: Determination of Break Points", IEEE Transactions on PWDR, vol. 2, pp. 638-645, December 1987.

[3] M.J. Damborg and S.S. Venkata, "Specification of Computer-Aided Design on Transmission Protection Systems", EPRI Report RP. 1766-6, January 1984.

[4] M.H. Dwarakanath and L. Nowitz, "An Application of Linear Graph Theory for Coordination of Directional Overcurrent Relays", Electric Power Problems - The Mathematical Challenge, SIAM, pp. 104-114, 1980.

[5] R. Ramaswami, S.S. Venkata, M.J. Damborg, J.M. Postforoosh and A.K. Jampala, "Enhanced Algorithms for Transmission Protective Relay Coordination", IEEE Transactions on PWDR, vol. 1, pp. 280-287, January 1986.

[6] C. Beeri and P.A. Bernstein, "Computational Problems Related to the Design of Normal Form Relational Schemes", ACM Transactions Database Systems, vol. 4, pp. 30-59, March 1979.

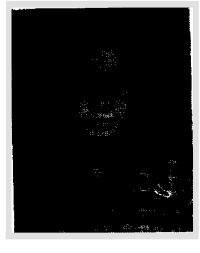
[7] D. Meier, "The Theory of Relational Databases", Computer Science Press, New York, 1983.

[8] J.D. Ullman, "Principles of Database Systems", Computer Science Press, New York, 1984.

[9] M.R. Garvey and D.S. Johnson, "Computers and Intractability", Freeman, San Francisco, 1978.

[10] W. Armstrong, "Dependency Structure of Database Relationships", Proceedings JFIP-74, North Holland, Amsterdam, 1974.

[11] D. Meier, "Minimum Covers in the Relational Database Model", Journal of the ACM, vol. 27, pp. 664-674, October 1980.



Jenkins [illegible] was born in Kolar [illegible] India, in 1947. He received the [illegible] degree in [illegible] from the Indian Institute of Technology, Madras in 1970 and the M.S. and the Ph.D. degree in Electrical Engineering from the University of [illegible] in 1972 and 1976 respectively.

Between 1976 and 1979, he served as an Assistant Professor at PSG College of Technology, Coimbatore. Since 1979, he has been a faculty of the Department of Electrical Engineering, at the Indian Institute of Science, Bangalore. He was a Visiting Associate Professor at Purdue University during 1985-86. He is the author of the text-book 'Digital Computer Principles' (Wiley Eastern Publishers, 1987). His current research interests are in parallel processing, fault tolerance, power system dynamics and real time simulation.

Dr. Jenkins is a member of the IEEE Computer Society. He is presently the Secretary of the IEEE Bangalore Section.

Khincha H P (S'67-M'69-SM'88) was born in Bangalore, India, in 1946. He received the B.E. degree in Electrical Engineering from Bangalore University in 1966, the M.E. and the Ph.D. from Indian Institute of Science, Bangalore, in 1968 and 1973 respectively.

From 1973, he has been a faculty of the Department of Electrical Engineering at Indian Institute of Science, Bangalore, and is presently Professor and Head of the Department. He has published over hundred papers in national and international journals/conferences. His research interests are in the areas of computer applications to power systems.

Professor Khincha is a member of the Power Engineering Society, and Computer Society. He has also been a Chairman of the Bangalore Section of IEEE, and is presently Chairman of IEEE India Council.

Shivakumar S (S'83) was born in Bangalore, India in 1962. He received his B.E. in Electronics from BMS College of Engineering, Bangalore University in 1984, and his M.Sc.(Engg.) degree in Electrical Engineering from the Indian Institute of Science, Bangalore in 1988. He is currently a Ph.D. student at the Department of Computer Science, University of Central Florida. His current research interests include parallel processing, computer aided design and data.

Dash P K received his B.E. in Electrical Engineering in 1962, and his M.E. in Electrical Engineering in 1964, from the Indian Institute of Science, Bangalore. He received his Ph.D. in 1971, from the University of Sambalpur, Orissa, India. Since 1965, he has been on the faculty of the Regional Engineering College, Rourkela, where he holds the rank of Professor. He has held visiting appointments at the University of New Brunswick, University of Calgary, Memorial University, University of Manitoba and BBC Brown Boveri. His research interests are in expert systems applications to power, HVDC, digital protection and microprocessor applications.