PERSONNEL PLANNING DATABASES AND MODELING: A SOFTWARE APPROACH

Luis B. Boza
Applied Statistics Department
Bell Laboratories, Holmdel, NJ

ABSTRACT

Two obstacles frequently encountered by model builders in personnel planning are the lack of data on the personal dynamics of organizations and of flexible software systems for execution of models tailored to particular organizations and issues. In this paper we present the main ideas behind the INTERACTIVE FLOW SIMULATOR, a software system constructed to avoid these difficulties. A procedure for building longitudinal personnel databases is outlined. An algorithmic view of discrete time models which encompasses a wide class of personnel planning and other models is presented.

I. INTRODUCTION

Personnel planning in large organizations is an interdisciplinary function which involves the study and the selection of courses of action to reduce the mismatch between needs for and availability of human resources, between organizational goals and personal aspirations, and between short and long term objectives.

One aspect of the planning function is the consideration of the ways in which the personnel dynamics of an organization affect, and are affected by personnel policies and environment factors. Flow models can play an important role in this regard. They permit the study of complex interrelationships, the representation of conflicts, the study of different scenarios, and the comparative evaluation of alternative policies. Goal-seeking flow models assist in the development of policies for approaching certain targets. Last, but not least, flow models enhance the understanding of the dynamics of the past by projecting the consequences of their continuation in the future.

Consulting experience with persons directly involved in model building for personnel planning, however, shows that many of them encounter data and software obstacles that limit the effectiveness of models as manpower planning tools. Historical data on the personnel dynamics of their organizations may not be readily available to them. Ad-hoc software for implementing their own models may have to be developed for each particular modeling task.

To eliminate these limitations a software system for discrete time flow modeling and related data analysis, called the INTERACTIVE FLOW SIMULATOR (IFS), was developed by the author. The main ideas behind this system are presented in this paper. The techniques for organizing past personnel records into databasis that provide information about the dynamics of an organization over time are described in Section II. In Section III we describe an algorithmic reconciliation of a wide class of discrete flow models into a model of a model. This reconciliation allowed the development of the modeling part of the above mentioned software system. Section IV illustrates the algorithmic reconciliation in the context of two simple, well-known examples. Detailed discussions of this algorithmic view of discrete flow models and of technical software points will be presented elsewhere.

The author believes that a careful second reading of the paper with each reader's own example in mind may provide the best illustration of this approach. Examples of databasis and of manpower planning models that have already been constructed using the IFS software system will, therefore, not be presented here but in a separate publication.

II. DATA ORGANIZATION

The first obstacle a model builder usually encounters is the lack of longitudinal data (i.e., life histories) with which to quantify the broad historical patterns of change in an organization. Without this kind of data the model building task may turn out to be either so time consuming that it becomes impractical, or, even worse, dependent on someone's preconceptions about reality rather than on reality itself.

A vicious cycle of underdevelopment is often encountered here: in organizations that do not have some longitudinal data available, it is difficult to present a cost/effectiveness argument in favor of the acquisition of such data. Fortunately the circle can be broken: a starting and very useful set of longitudinal data can usually be constructed with a minimum of effort and cost in a way analogous to the construction of motion-pictures from still photographs.

Most organizations have payroll or other computerized personnel files which they use in day-to-day operations. As time elapses these files are updated to keep track of internal moves, reclassifications, and entries or exits of employees. Typically, such organizations keep copies of old files - say, year-end files - in storage for a certain period of time.

These stored files contain records of the persons who were members of the organization at the given moments of time. The records contain information about each person's place in the organization (i.e., "organizational attributes" such as job, location, etc.) and about the person itself (i.e., "personal attributes" like education, age, etc.). Most importantly, these records contain an identifier such as a payroll number or social security number, through which each person can be traced over successive past files.

The construction process for an IFS longitudinal data base proceeds as follows. Depending on the planning issues of interest to the organization, certain fields of information (attributes) in the fixed-time files are selected for inclusion in the longitudinal data. Each such field is partitioned into a number of (non-overlapping and exhaustive) categories. The files are then merged according to the identifier, and the merged file is laid out so that each person is represented by a record covering the entire selected time frame. Appropriate codes for the attributes are used at those time points at which the person is not present in the original files. Records in this new file still carry the identifier. It is sometimes convenient, as a storage-saving technique, to condense records that are the same except for the identifier, into a single record which contains, in addition to the longitudinal data, the count of employees which share the corresponding data.

The resulting file contains, therefore, discrete-time, multi-attribute data. By using a retrieval language based on relational database concepts [3] the IFS software system can extract from this type of files either "static" or "dynamic" multi-dimensional tables. Exits, entries of employees, transition flows of many kinds, vacancy - filling patterns, as well as fixed-time population profiles, for instance, can be readily obtained. These tables may then be displayed and/or used for estimating parameters of the desired flow models.

III. A CONCEPT OF A MODEL OF DISCRETE FLOW MODELS

A second obstacle that a model builder usually faces is the lack of a flexible software system within which models reflecting the realities of his own organization could be quickly implemented. If software is not available the model builder may often have to use simplifying, but not too realistic, assumptions in order to have a mathematically tractable structure. Ad-hoc software could be built or borrowed. The former alternative usually implies time delays. In the latter case the model builder may have to accept software which contains rules or variations that do not quite represent the situation being modeled.

Fortunately, this software obstacle can also be overcome. A review of the literature on discrete (time and state-space) flow models in manpower planning, as well as in other areas, shows that many such models, while differing in field of application and in purpose, share a common algorithmic formulation. This common algorithmic formulation is, in a very real sense, a model of a model. A software system programmed to "understand" the model of a model can, thus, implement any particular model belonging to this class and formulated in this framework.

A model, in our algorithmic formulation, is completely specified by three kinds of descriptions (Figure 1)

First, the initial description which provides information about the organization being modeled as of some chosen initial time. Second, the within interaction description which provides information as to how the organization evolves from one time point to the next time point in the projection interval. Third, the description of between-interaction effects, which provides information about changes, if any, in the within iteration description from one iteration to the next.

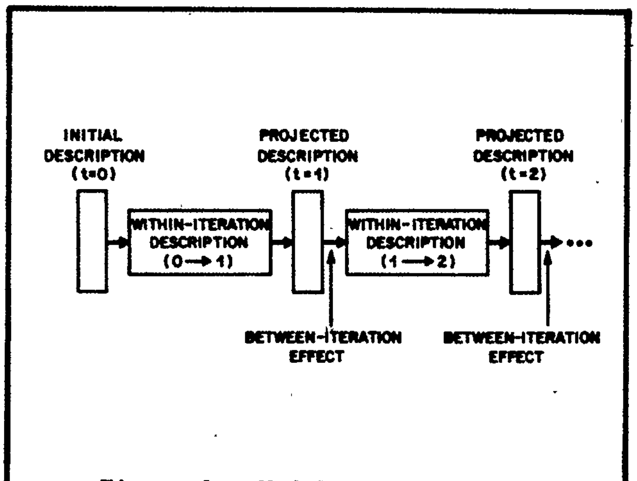
INITIAL DESCRIPTION (t=0)
PROJECTED DESCRIPTION (t=1)
PROJECTED DESCRIPTION (t=2)
WITHIN-ITERATION DESCRIPTION (0→1)
WITHIN-ITERATION DESCRIPTION (1→2)
BETWEEN-ITERATION EFFECT
BETWEEN-ITERATION EFFECT

Figure 1. Model Descriptions

These three elements of the algorithmic formulation are explored in the rest of this Section, and illustrated in Section IV, with two simple examples.

A - Initial Description

The model building task is that of creating a mathematical structure within which certain precisely defined issue(s) can be studied. The first step in that task is to state the description of the organization being modeled as of some chosen initial time. Four kinds of information form the initial description of an organization in our approach:

i - the set of types of the population elements,

ii - the initial population array

iii - the set of types of the vacancies

iv - the initial vacancy array

i - The set of types of the population elements

Once the issues to be studied via a particular flow model have been defined, the level of aggregation of the model population required to consider the issues has to be selected. In multitype branching processes terminology [4] the set of types of the population elements in the model has to be selected.

This is done by choosing attributes (dimensions) for inclusion in the model, and the categories of each such attribute. The set of categories for each attribute is called a set of marginal types. The set of types of the population elements is, then, the direct product of the chosen sets of marginal types. Since we may have attributes with just one category in the model, the set of types of the population elements can always be expressed, without any loss of generality, as the direct product of at least two sets of marginal types.

ii) Initial population array

The initial population array in the model is simply an array of numbers (P_0) over the set of types of the population elements. Each such number provides the count of employees of the corresponding type present in the organization as of some chosen initial time.

iii) The set of types of the vacancies

Most manpower models impose constraints on the number of employees allowed in the categories of one or more attributes. These models have to keep track of vacancies in the categories of such attributes so that the population can evolve over time satisfying the given size constraints. The set of types of the vacancies is, therefore, defined as the direct product of the sets of marginal types on which size constraints are to be imposed in the model. Without loss of generality, the set of types of the vacancies can always be represented as the direct product of at least one, but not all, of the sets of marginal types used for describing the population elements.

The inclusion of types of vacancies in the initial description is important and, to our knowledge, formally new. With few exceptions, vacancies have been used for a long time, either explicitly or implicitly, in discrete time flow models. White [5] brought them to the same level of importance as that of the population in the model. Even White, however, when considering the conflicts between population elements that arise in the process of filling vacancies, refers to "substrata in the population" rather that to types of vacancies. The change of emphasis that our terminology implies helps clarify the role of vacancies by making more apparent the population-vacancies duality that is so crucial in the description of this kind of models.

iv) Initial Vacancy Array

Finally, the initial vacancies in the model (V_0) have to be given as an array of numbers over the set of types of vacancies.

In summary, the initial description of a discrete flow model is given in our algorithmic formulation by two tables P_0 and V_0. P_0, the initial population, contains as many cells as types of population elements in the model. V_0, the initial vacancies, contains as many cells as types of vacancies in the model.

B - Within-iteration description

The next component of the algorithmic specification of this class of discrete flow models is the within-iteration description: the procedure by which, for each time-t to time-(t+1) iteration, the model transforms the population and vacancy arrays at time-t (P_t and V_t) into the corresponding pair of arrays at time (t+1) (P_{t+1} and V_{t+1}) (see Figure 2).

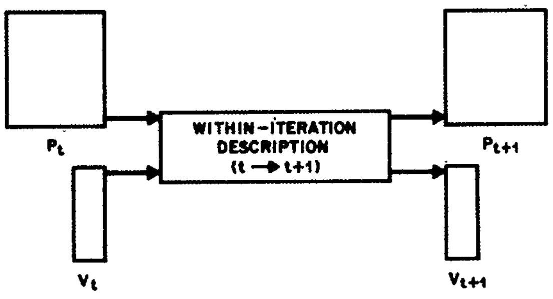

Figure 2. Within-Iteration Description

The within-iteration description, therefore, can be viewed as a mapping from the set of population and vacancy arrays onto itself. This mapping can be algorithmically described as the composition of an ordered sequence of parameterized primitive operations.

Primitive operation

A primitive operation O, the basic building block of within-iteration descriptions, is a class of mappings $\{o_a | a\varepsilon A(O)\}$. Each element o_a in this class is a mapping from the set of population and vacancy arrays onto itself. Elements in the class differ in their parameter, a, a certain vector with values in A(O), a parameter space for the operation O.

A basic set of primitive operations

A basic set of primitive operations, which allows the algorithmic formulation of within-operation descriptions of a wide class of models, contains only four operations which, for simplicity, will be called: GROWTH, LOSS, MARKOV and FILL. Loosely described the roles and parameterization of these four operations are:

GROWTH - it increases or decreases the vacancies as required by the size constraints, keeping the population array constant. The parameter of this operation is the set of growth rates and indicators for a simulation or deterministic execution of the operation.

LOSS - it depletes the population array by losses and increases the vacancies accordingly. The parameter for this operation is the set of loss rates for the various population types and indicators for the simulation or deterministic execution of the operation.

MARKOV - it moves population elements between categories of one or more attributes, and updates vacancies accordingly. This operation can be used to model flows of population elements which are not induced by existent vacancies (internal "push" flows in Bartholomew's [2] terminology). Obviously vacancies will not be affected by strictly markovian moves in attributes not included in the set of types of the vacancies, but may be affected otherwise. The parameter of this operation is the set of transition rates between categories of the attributes involved, and indicators for simulation or deterministic execution of the operation. This operation may be performed with substochastic or superstochastic matrices to represent a deterministic branching behavior. Deterministic versions of population interaction behavior, as in epidemic or predator-prey models, could also be performed by this operation. In this case the matrix involved may have some negative entries.

FILL - it creates a cascade of vacancies on a given directed graph structure on the set of types of vacancies, with possible spills to the outside. At each step in the cascade, replacements to fill the vacancies are selected. Both the population and vacancy arrays are updated after each replacement. The FILL operation, in general, can be used in flow models to represent those flows of population elements that are caused by the need to fill existing vacancies ("pull" flows in Bartholomew's [2] terminology).

The parameter of this operation includes the graph of the cascade of vacancies, the rates of flow of vacancies to the "outside," methods for replacement selection, and rules for changes in the categories of attributes which are induced by the selection, or the nonselection of a population element to move. At each stage in the cascade of vacancies, methods for selecting replacements are specified as a sequence of partitions of the corresponding vacancies. Each partition is completely described by the attributes on which it operates, the rule used for the breakdown of the vacancies, and by numerical data needed for the execution of the given rule. Rules may take such forms as priority schemes, proportional allocations, simulation draws, and optimization mechanisms including goal seeking subalgorithms.

Construction of within-iteration descriptions

The within-iteration description of a model is constructed by choosing one or more primitive operations from the basic set, parameterizing them, and selecting their order of execution. A primitive operation may appear one or more times in a within-iteration description.

The primitive operation chosen for starting the within-iteration description (say O_1 in Figure 3) acts upon the population and vacancy arrays at the beginning of the iteration and produces a new population and vacancy array pair. These are acted upon by the second operation (O_2). The population and vacancy array pair output by the second operation are the inputs to the third, and so on. The arrays output by the last chosen primitive operation (O_n) are, by definition, the population and vacancy arrays at the end of the iteration.

C - Description of between - iteration effects

Between iteration effects are changes in the within-iteration description of a flow model that take place between the end of one iteration and the beginning of the next. If the model has a stationary within-iteration description, the within-iteration description of the first iteration is used for any subsequent iteration in the projection interval of the model.

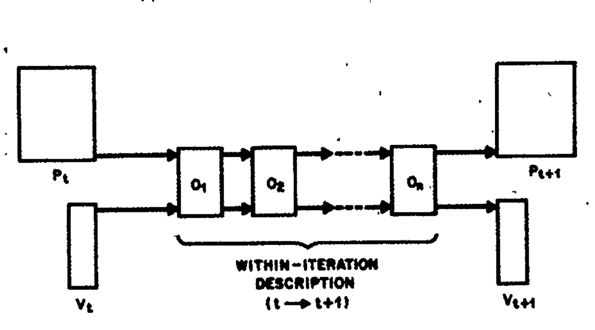

Figure 3. Within-Iteration Description and Primitive Operations

Otherwise, between iteration effects may be used to change the within-iteration descriptions as the model execution proceeds.

A few examples of between-iteration effects that are often of interest in planning models are:

i) Exogenous reparameterization of chosen primitive operations

If the GROWTH operation is used in a model, for instance, the growth rates parameter may be changed overtime as a function of exogenous projections of personnel size based, perhaps, on work-related or economic time series models. The between-iteration effect will, in this case, accomplish the readjustment of growth rates during the projection interval of the model.

ii) Endogenous respecification of the chosen primitive operations

In a model involving the LOSS operation for instance, loss rates may be changed over time as a function of, say, perceived promotion opportunities. If perceived promotion opportunities are defined based on the previous output of the model, the change in loss rates is a between-iteration effect of feedback type.

iii) Deletion, addition or reordering of primitive operations in within-iteration description

Suppose, for instance, that the organization being modeled contemplates an early-retirement plan for the third iteration of the model. The effect of this plan could be included in the model by a respecification of the loss rates or by adding a new LOSS operation immediately following the regular LOSS operation in the within-iteration description of the third iteration of the model. This new operation will act upon the population depleted by regular causes of loss, and will produce new losses and further readjustments of the population and vacancy arrays. The rates of loss parameters for this new operation will then be the conditional probabilities of loss due to the early-retirement plan, given that the employee has not left because of regular causes of loss. The new operation may be deleted from the within-iteration description of the fourth iteration if the early-retirement plan is a one-time period policy of the organization.

IV. EXAMPLES

Two examples of discrete-time flow models are presented in this Section for the purpose of illustrating the algorithmic description covered in Section III.

1 - Young and Almond [6]

A simple model for predicting distributions of staff among different grades in an organization was proposed by Young and Almond [6].

The organization is assumed to have n grades. An employee, who at time-t is in grade g, may leave the organization before time (t+1) with probability ℓ_g. Otherwise he may move to grade g' (g'=1,2,...,n) with probability $q_{g,g'}$. Employees are assumed to move independently of each other. Let d be the rate of growth of the organization. Vacancies created by losses and expansion during a (t,t+1] time interval, are filled, during the same interval, by new employees. These new employees enter the organization in a grade g, with probability $r_g, \Sigma\, r_g = 1$ independently of each other.

Let $P_t = (P_t(g))$ denote the distribution of employees by grades at an arbitrary time-t (t=0,1,2,...), $Q = (q_{g,g'})$ the transition probability matrix, and $L = (\ell_g)$ and $R = (r_g)$ the arrays of loss and recruitment probabilities. The deterministic iteration equation can be expressed as

$$E[P_{t+1}|P_t] = S_tQ + \left\{\sum_g P_t(g)\ell_g + d \sum_g P_t(g)\right\}R$$

where

$$S_t = (S_t(g)),\ S_t(g) = P_t(g)(1-\ell_g).$$

Algorithmic Formulation

i - Initial Description

Set of Types of Vacancies

The size constraint in this model is not specified by grades. Rather, it is specified for the organization as a whole. Vacancies, therefore, are a scalar in this model, or equivalently, form an array on a one-category attribute. We may, thus, take "organization" to be an attribute with a unique category "X", and let {X} be the set of types of vacancies.

Set of Types of the Population

The set of types of the population elements can be expressed as the direct product

{X}x{grade 1,...,grade n}

Initial Vacancy and Population Arrays

Let $V_0 = 0$ be the initial vacancies, an array on {X}, and P_0 be the initial population, an array on the set of types of the population.

ii - Within Iteration Description

Ordered Sequence of Primitive Operations

Four primitive operations are used in this model: GROWTH, LOSS, MARKOV and FILL. Growth computes the vacancies created by the required expansion in size, LOSS depletes the population, MARKOV creates the intergrade moves and FILL replaces vacancies. These operations are executed, within each iteration, in the above order.

Internal Specification of the Primitive Operations

Clearly, GROWTH is parameterized by d, the growth rate; LOSS is parameterized by L, the array of loss probabilities and MARKOV by Q, the transition matrix over grades. In FILL the graph of the cascade of vacancies has a unique node "X", from which all vacancies flow to the outside. The method for replacing vacancies is a partition of the vacancies in "X" over the different categories of the attribute grade. The partition is done via a proportional allocation rule which uses the entries of R (data for the rule) as proportions.

A deterministic execution of this within-iteration description implements the above mentioned equation. Taking the first iteration as an example, it is apparent that:

GROWTH: receives P_0 and V_0 and outputs $P' = P_0 : V' = d\sum_g P_0(g)$

LOSS: receives P' and V' and outputs

$P'' = (P'(g)(1-\ell_g))$ and

$V'' = V' + \sum_g P'(g)\ell_g$

MARKOV: receives P'' and V'' and outputs

$P''' = P''Q$ and

$V''' = V''$

FILL: receives P''' and V''' and outputs

$P_1 = P''' + V'''R$

$V_1 = 0$

In a random simulation mode for the operations LOSS, MARKOV and FILL the execution produces a realization of P_1 conditional on P_0.

iii - Between Iteration Effects

No between iteration effects are present in Young and Almond's original model since it has a stationary within-iteration description. A number of between-iteration effects of great practical importance could, however, be included. For instance, successive redefinitions of the transition and/or hiring rates to seek a certain desired distribution of employees by level (Bartholomew [2].)

2 - Bartholomew [1] -

The following model is a discrete time version of a multistage renewal process considered in Bartholomew [1].

The organization is assumed to have n grades. An employee, who at time t is in grade g and has seniority s within the grade may leave the organization before (t+1) with probability $\ell_{g,s}$.

Assume, for simplicity, that grades have constant population size through time. Vacancies created in (t,t+1] are filled, in the same interval, as follows: those in grade 1 are filled with employees in grade 2, in strict order of seniority; vacancies in grade 2 (losses plus moves into grade 1) are next filled from grade 3, again in strict seniority. The process continues until vacancies in grade n, the lowest grade, are filled from an outside market. Employees entering a grade start at the lowest seniority bracket in the grade.

Algorithmic Formulation

i - Initial Descriptions

Set of Types of Vacancies

Size constraints, and therefore vacancies, are defined by grades in this model. The set of types of the vacancies is

{grade 1, grade 2,...,grade n}.

Initial Vacancy Array

Let V_0 be an array on the set of types of vacancies with all its entries equal to zero.

Set of Types of the Population

Let $(s_1,s_2...,s_m)$ be the finite set of categories chosen for the attribute "seniority in grade." The set of types of the population is then

$\{\text{grade } 1,\ldots,\text{grade } n\}x\{s_1,s_2,\ldots,s_m\}$.

Initial Population Array

Let P_0 be an array on the set of types of the population.

ii - Within-Iteration Description

Ordered Sequence of Primitive Operations

Three primitive operations are used in this model: LOSS, MARKOV and FILL. LOSS is used for depletion, MARKOV for seniority shifts and FILL for replacing vacancies. These operations, are executed, within each time iteration, in the above order.

Internal Specification of the Primitive Operations

LOSS is specified by $L = (\ell_{g,s})$ the array of loss probabilities. MARKOV is specified by a matrix Q of transition probabilities between categories of the "seniority in grade" attribute. The form of this matrix is a function of the categories chosen for the attribute and of their order. Moves induced by this operation occur in the "seniority in grade" attribute. Vacancies, which are defined on grades, are not affected.

The graph of the cascade of vacancies of FILL has n nodes (the grades) and links: grade 1 $\rightarrow$ grade 2,...,grade (n-1) $\rightarrow$ grade n. Vacancies can flow to the outside only from grade n. For grades 1 through (n-1) the method for selection of replacement is the same: vacancies are partitioned by grades - a trivial split since there is only one grade in the pools, then they are partitioned on the "seniority in grade" attribute according to a priority rule. The data given for the priority rule is an ordered sequence of categories of the "seniority in grade" attribute. In Bartholomew's model it is the strict seniority sequence.

As mentioned in Section III, the selection to move sometimes causes changes in attributes. For instance a selection to move from grade 2 to grade 1 causes a shift to the lowest bracket in the "seniority in grade" attribute. The detailed description of this transition is also part of the internal specification of FILL.

Finally, vacancies in grade n are filled by selecting replacements from an inexhaustible labor market into the "lowest" seniority status in the grade.

iii - Between-Iteration Effects

No between-iteration effects are present in the original model. A number of interesting between-iteration effects could, however, be also included.

IV. CONCLUSIONS

The ideas summarized in this paper were developed during an initial research phase on data organization and reconciliation of models. Once this phase of the work was completed the INTERACTIVE FLOW SIMULATION software system was constructed. It permits:

- retrieval of historical data from longitudinal information of the kind described in Section II;

- statistical analysis of the retrieved data, either for the study of the past, or as a step towards modeling;

- construction of user-defined flow models of the kind described in Section III;

- execution of these models and analyses of their output.

Full interactive capabilities were included to allow the free exercise of the user's initiative in these four functions of the system. Moreover, these functions were not considered in isolation, but as parts of a whole process. Users, therefore, may go from data retrieval and analyses to model building and execution, and perhaps back to the start of the modeling cycle, without leaving the IFS interactive environment.

A detailed description of our experience to date in the use of the system in both research and planning will be given elsewhere. It should be pointed out, however, that longitudinal data bases have been constructed and have helped in the identification of important planning issues. Descriptive and goal-seeking models have been built and their results have enhanced the organizations' ability to study complex human resources planning alternatives. Each result, to be sure, has raised new planning questions. These questions indicate that the use of flow models and related data analyses in human resources planning is a growing and an exiting field, full of opportunities for research and applications.

Acknowledgments

This author was fortunate to have the assistance of W. M. Brelsford, P. A. Groll, S. B. Guthery, C. T. Schlegel, and A. T. Seery at various stages of programming in the project. Their collaboration is gratefully acknowledged.

REFERENCES

1. Bartholomew, D. J., (1963), "A Multistage Renewal Process," J. R. Statist. Soc., B, Vol. 25, pp 150-168.

2. Bartholomew, D. J., (1973), "Stochastic Models for Social Processes," Wiley, 2nd edition.

3. Codd, E. F., (1970), "A Relational Model of Data for Large Shared Data Banks," Comm. A.C.M., 13, 6, pp 377-387.

4. Harris, T. E. (1963), "The Theory of Branching Processes," Prentice-Hall.

5. White, H., (1970), "Chains of Opportunity," Harvard University Press.

6. Young, A., and Almond, G., (1961), "Predicting distributions of staff," Comp. J., 3, pp. 246-250.