International Institute for Applied Systems Analysis
IIASA
www.iiasa.ac.at

Implementation of a Model of Service Location

Vetschera, R.

IIASA Working Paper

WP-81-030

March 1981

NOT FOR QUOTATION
WITHOUT PERMISSION
OF THE AUTHOR

IMPLEMENTATION OF A MODEL OF SERVICE LOCATION

R. Vetschera

March 1981
WP-81-30

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS
A-2361 Laxenburg, Austria

SUMMARY

This Working Paper describes the implementation of a model developed by R. Domanski and A. Wierzbicki. The first part contains a brief description of the model. The second part is directed to the user of the simulation system and contains all information required to produce simulation runs. The remaining parts are a technical documentation for those who want to modify the system or adapt it for different computers. The internal structure of the system is explained, hardware and software requirements for its installation are specified and possible modifications and extensions are indicated. An appendix contains the source listings of all programs.

CONTENTS

IMPLEMENTATION OF A MODEL OF SERVICE LOCATION

R. Vetschera

1. THE MODEL

1.1 Overview

The model is used to describe the development of services in a network of N villages. Different types of service with different times of introduction and different employment effects are analyzed. The development is also influenced by two other sectors of the economy: industry and intensive agriculture. The state of the model at time t is thus described by three variables:

$P_{x,t}$ .... population in village x at time t.

$E^{v}_{x,t}$ .... employment in industry ($v = 1$) or intensive agriculture ($v = 2$) in village x at time t.

$E^{u}_{x,t}$ .... employment in service u in village x at time t.

The model will start from an exogenously given state (P_0, E^v_0, E^u_0) at time $t = 0$. The dynamic development of the model is governed by three equations. Each of these equations determines the change of one state variable.

Changes in population are influenced by the natural growth rate of population r and migrations $M_{x,t}$ within the network:

$$(1.1\text{-}1) \qquad P_{x,t+1} = (1+r)P_{x,t} + M_{x,t}$$

Employment in industry and intensive agriculture depends on the employment in the previous period and changes $\Delta^{v}_{x,t}$ which are generated by an occupational employment mechanism:

$$(1.1\text{-}2) \qquad E^{v}_{x,t+1} = E^{v}_{x,t} + \Delta^{v}_{x,t}$$

Employment in services is modified in a similar way:

$$(1.1\text{-}3) \qquad E^{u}_{x,t+1} = E^{u}_{x,t} + \Delta^{u}_{x,t}$$

These three changes will be described in the following sections.

1.2 Population

The potential of migration $M^{*}_{x,t}$ in one village comprises all people affected by changes in employment and their dependants:

$$(1.2\text{-}1) \qquad M^{*}_{x,t} = k(\sum\Delta^{v}_{x,t} + \sum\Delta^{u}_{x,t})$$

This potential migration has to be modified to ensure that the sum of migration flows within the network is zero:

$$(1.2\text{-}2) \qquad M_{x,t} = M^{*}_{x,t} - \frac{\sum_{\bar{x}} M^{*}_{\bar{x},t}}{\sum_{\bar{x}}\left(P_{\bar{x},t} - M^{*}_{\bar{x},t}\right)} \cdot (P_{x,t} - M^{*}_{x,t})$$

This migration plus the natural growth rate r of population determine the size of population in the next period of time as defined in equation (1.1-1).

1.3 Employment in Other Sectors

Changes in the employment in intensive agriculture and industry correspond to the unemployed labour force at time t:

$$(1.3\text{-}1) \qquad \Delta^{v}_{x,t} = \delta^{v}_{x} \sum_{y}\left(\frac{P_{y,t}}{k} - \sum_{v} E^{v}_{y,t} - \sum_{u} E^{u}_{y,t}\right)$$

where δ^{v}_{x} is a parameter indicating the use of the total unemployed labour force for employment in sector v in village x. If δ^{v}_{x} is zero, the employment E^{v}_{x} will remain at the initial level $E^{v}_{x,0}$.

1.4 Employment in Services

There are three phases of development of services in the model:

(a) a service u is not introduced before a specific point in time t_{u0}. For $t < t_{u0}$, employment in service u is kept at zero level.

(b) in some villages the service is randomly introduced at time $t = t_{u0}$.

(c) for times $t > t_{u0}$, two possible cases must be considered:

(c1) a service u already exists in a village x and E^{u}_{x} is modified according to the demand for that service.

(c2) if the service u does not yet exist in village x, it might be newly opened.

So there are two basic mechanisms determining the employment in services: an opening mechanism, which is used in phase (b) and (c) and an employment modification mechanism, which is used in phase (c). Furthermore, there are differences in the opening of services in phase (b) and (c).

A new service will be introduced at time $t = t_{u0}$ (phase b) at a fixed level of employment E^{u0} in randomly chosen villages. The probability that a new service will be introduced in village x is

$$(1.4\text{-}1)\qquad p(E^u_{x,t+1} > 0) = \psi^u_{1x}\,\frac{(P_{x,t} + \gamma R_{x,t})N}{\sum_{\bar{x}\in X}(P_{\bar{x},t} + \gamma R_{\bar{x},t})}$$

where

γ and ψ^u_{1x} are parameters

N is the number of villages

X is the set of all villages ($\{1,\ldots,N\}$)

and

$$R_{x,t} = \sum_u E^u_{x,t}$$

is the existing level of other services in village x.

When an existing service is to be modified, (phase c1) the first step is to calculate the attractiveness of all services existing in village x for people living in all other villages, y as:

$$(1.4\text{-}2)\qquad A = \begin{cases} \dfrac{E^u_{x,t} + \sum_{\bar{u}\neq u}\lambda_{\bar{u}u}E^{\bar{u}}_{x,t}}{d^{\alpha}_{xy}} & \text{if } E^u_{x,t} > 0 \text{ and } d_{xy} \leq D_0 \\[2ex] 0 & \text{if } E^u_{x,t} = 0 \text{ or } d_{xy} > D_0 \end{cases}$$

where $\lambda_{\bar{u}u}$, D_0 and α are parameters and d_{xy} is the distance between x and y. This attractiveness is compared with that of other villages $\bar{x}$ to the same village y, giving the relative attractiveness

$$(1.4\text{-}3)\qquad \eta^u_{xy} = \frac{A^u_{xy}}{\sum_{\bar{x}\neq x}A^u_{\bar{x}y} + A^u_{xy}}$$

The number of people attracted from village y to village x is therefore $P_{y,t} \cdot \eta^u_{xy}$ and the total employment needed to satisfy the demand encountered in village x is

$$D^u_x = \sum_y P_y q^u \eta^u_{xy} \qquad (1.4-4)$$

where q^u is a parameter for service u.

Actual employment at time t+1 will differ from D^u_x due to two influences:

(1) It is assumed that employment will not exactly be changed by the difference required to reach D^u_x, but the changes in employment may over- or under-react to changes in demand thus:

$$E^u_{x,t+1} = E^u_{x,t} + a(D^u_x - E^u_{x,t}) \qquad (1.4-5)$$

where a is a parameter indicating the precise reaction $(a = 1)$, over-reaction $(a > 1)$ or under-reaction $(a < 1)$.

(2) There is a minimum level of employment for each kind of service which is defined as $b\,E^{u0}$. E^{u0} is the initial level of employment at which the service was started and b is a parameter. If demand falls below the level corresponding to minimal employment, the service is shut down completely and $E^u_{x,t+1}$ is set to zero.

The opening of a new service after time t_{u0} (phase c2) combines both the mechanisms described above. First, a hypothetical attractiveness is computed in a similar way to (1.4-2) and (1.4-3), using the starting size E^{u0} as a service size:

$$\tilde{A}^u_{xy} = \frac{E^{u0} + \sum_{\bar{u} \neq u} \lambda_{\bar{u}u} E^{\bar{u}}_{x,t}}{d^{\alpha}_{xy}} \qquad (1.4-6)$$

$$(1.4\text{-}7) \qquad \tilde{\eta}^u_{xy} = \frac{\tilde{A}^u_{xy}}{\sum_{\bar{x} \neq x} A^u_{\bar{x}y} + \tilde{A}^u_{xy}}$$

This estimated attractiveness is used to calculate the possible size of employment in a new service u:

$$(1.4\text{-}8) \qquad \tilde{D}^u_x = \sum_y P_y q^u \eta^u_{xy}$$

Opening a new service is considered worthwhile if $\tilde{D}^u_x > E^{u0}$. Whether the service will actually be opened is determined by a random procedure. The probability of opening the service is computed in a way similar to (1.4-1):

$$(1.4\text{-}9) \qquad p(E^u_{x,t+1} > 0) = \psi_2 \frac{(P_{x,t} + \gamma R_{x,t})N}{\sum_{\bar{x}} (P_{\bar{x},t} + \gamma R_{\bar{x},t})}$$

If the service is to be opened, employment in it is set to $\tilde{D}^u_x$.

2. USING THE SIMULATION SYSTEM

2.1 Components of the System

The simulation system, which implements the model described in chapter one consists of two sets of programs:

--The first set contains the actual simulation program. This program performs all the calculations specified in the model and stores the successive states of the model in a result file.

--The second set contains utility programs, which process the result file and print various reports.

The separation of actual simulation and the generation of output enables the user to produce more detailed outputs only for those runs for which they are really needed and without rerunning the simulation program for each type of output required.

2.2 Program Restrictions

Both the simulation program and the utility programs have been developed for a specific version of the model and are based on some assumptions of that version. In comparision with the general model described in chapter one, these assumptions lead to three restrictions of the system:

(a) The size of the model is limited to up to 36 villages and up to five services.

(b) A specific geographical structure of the network (as described in 3.1.3) is assumed and the distance between two villages is computed according to that structure.

(c) The initial state of the model is defined as

$P_{x,0} = P_0$ for all villages

$E^v_{x,0} = 0$ for all villages

$$E^u_{x,0} = \begin{cases} 0 & \text{if } t_{u0} > 0 \\ \text{generated according to (1.4-1)} & \text{if } t_{u0} = 0 \end{cases}$$

2.3 Running the Simulation Program

The simulation program SIM is used to generate simulation runs. It will read all parameters required for the run from a file called PAR and will produce a result file named OUT. Messages for the user will be written to the standard FORTRAN output unit (the terminal in the UNIX system).

2.3.1 *Input to the Simulation Program*

All input to the simulation program is contained in the parameter file PAR. This file consists of several lines. Most lines contain one or more parameters written in free format and separated by commas (lines requiring a special format are indicated in the following description). The contents of each line are:

line	contents
1	nr of villages N, nr of services U, nr of other sectors V
2	α, D_0, a, b
3	k, r, P_0 (initial population size of each village)
4	E^{u0} for all services
5	t_{u0} for all services
6	ψ_1, ψ_2, γ (if ψ_{1x}^u is to be specified for each individual village and service, a value < 0 for ψ_1 must be specified in this line. In this case, the line is followed by N additional lines, one for each village. Each of these lines contains U values specifying ψ_{1x}^u for the village and the U services. In numbering the following parameter lines, it is assumed that these lines are not present.)
7-11	$\lambda_{\bar{u}u}$ (each line contains the U values corresponding to 1 value of $\bar{u}$)
12	q^u for all services
13-14	These lines are used to set the ∂_x^v, line 13 for industry (v=1) and line 14 for intensive agriculture (v=2). They must be written in a fixed format. Their contents are:

column	content
1-05	value of ∂_x^v if sector v occurs in a village
6	blank
7-42	N indicator values (0 or 1) for the villages. If sector v is to occur in a village, 1 must be specified. If it is not to occur, 0.

line	contents
15	Starting value for the random number generator (a large integer number). Different starting values will lead to different results in the randomized opening of new services.
16	Number of periods to be simulated.

The program does not perform any checks on the parameters. When parameters or the structure of the parameter file are incorrect, computational errors may cause the program to terminate abnormally.

A sample parameter file is shown in figure 2-1.

```
36,5,2
2.0,100.0,1.0,0.5
3,0.01,500.0
5.0,10.0,15.0,20.0,25.0
0,5,10,15,20,25
0.166667,0.166667,100.0
0.2,0.1,0.0666667,0.05,0.04
0.2,0.1,0.0666667,0.05,0.04
0.2,0.1,0.0666667,0.05,0.04
0.2,0.1,0.0666667,0.05,0.04
0.2,0.1,0.0666667,0.05,0.04
0.004,0.007,0.012,0.017,0.020
0.005 0100000000010010000000100100000000100
0.002 0000000011010011000011100111000000000
896745231.0
50
```

Figure 2-1. Parameter file for the simulation program.

2.3.2 *Output from the Simulation Program*

Output from the program consists of two parts: a result file OUT and messages on the standard output device.

The result file contains information about parameter settings for the simulation run and the successive states of the model. Its contents are used by the utility programs for generating output listings of the simulation.

The simulation program will write the following messages on the standard output device:

PROCESSING TIME T = t

This message is issued every time before calculations for a new period t will begin.

MIGRATION ERROR, SUM = s

This message is printed when, due to rounding errors, the sum of all migration flows in the model exceeds a threshold value (normally 0.0000001). Calculations will proceed after the message is written.

2.3.3 *Time and Space Requirements*

On a CDC Cyber 170-720, the program needs about 16k words (CM 40000 octal) of main storage. The CPU time required for a simulation run of 50 periods is about 450 seconds (T 700 octal).

2.4 Utility Programs

2.4.1 *Types of Output*

Two types of output can be produced from the result file: maps and time paths.

A map shows the spatial distribution of one state variable at one point of time. The programs used to produce maps will print sets of maps for several variables and periods. As an example, a map of the population at time 50 is shown in figure 2-2.

A time path represents the development of one village and contains all state variables of the village for all periods of time. An example of a time path is shown in figure 2-3.

In addition, all utility programs produce a listing of the simulation parameters as shown in figure 2-4.

2.4.2 *Running the Utility Programs*

There are five utility programs which can be used to process the result file. Their names and functions are:

name	output produced.
ALLVL	time paths showing the individual development of each village.
LONG	maps showing the spatial distribution of population, employment in services and employment in other sectors.
SHORT	maps showing the spatial distribution of population and employment in other sectors.
SUMRY	time path for the whole system
VLIST	time path showing the individual development of one village

POPULATION AT TIME 50

519.333		1183.52		519.754		585.351		519.333		502.594		
	579.205		815.016		1147.44		1077.30		636.225		1917.66	
703.270		537.991		1511.16		847.800		634.418		519.333		
	602.900		807.733		936.077		772.968		1434.75		519.460	
642.079		1371.96		1346.73		815.381		1237.24		565.056		
	569.279		520.305		668.675		997.719		518.604		519.745	

Figure 2-2. Example for a map.

TIME PATH FOR VILLAGE 9

TIME	POPULATION	EMPLOYMENT IN SERVICE 1	2	3	4	5	SUM	EMPLOYMENT IN SECTOR 1	2
0	500.000	0.	0.	0.	0.	0.	0.	0.	0.
1	516.729	6.50000	0.	0.	0.	0.	6.50000	0.	1.19700
2	519.527	4.68113	0.	0.	0.	0.	4.68113	0.	2.37808
3	523.889	3.74012	0.	0.	0.	0.	3.74012	0.	3.57021
4	529.092	3.24576	0.	0.	0.	0.	3.24576	0.	4.77085
5	527.627	2.98399	0.	0.	0.	0.	2.98399	0.	5.97877
6	533.511	3.17872	0.	0.	0.	0.	3.17872	0.	7.17682
7	539.536	3.26736	0.	0.	0.	0.	3.26736	0.	8.37893
8	545.682	3.30918	0.	0.	0.	0.	3.30918	0.	9.58618
9	551.935	3.33286	0.	0.	0.	0.	3.33286	0.	10.7991
10	549.277	3.35174	0.	0.	0.	0.	3.35174	0.	12.0181
11	595.803	3.26239	0.	15.3387	0.	0.	18.6011	0.	13.2224
12	621.678	3.60304	0.	21.9824	0.	0.	25.5854	0.	14.4182
13	640.644	3.97307	0.	26.3034	0.	0.	30.2765	0.	15.6166
14	656.014	4.30724	0.	29.1519	0.	0.	33.4591	0.	16.8177
15	661.595	4.58912	0.	31.0907	0.	0.	35.6798	0.	18.0233
16	726.431	4.79257	0.	32.2838	31.5409	0.	68.6172	0.	19.2191
17	725.090	5.03270	0.	34.3926	23.1105	0.	62.5358	0.	20.3268
18	727.867	5.16366	0.	35.5230	18.8453	0.	59.5319	0.	21.4607
19	733.177	5.25041	0.	36.2398	16.7026	0.	58.1929	0.	22.6110
20	729.759	5.32132	0.	36.7957	15.6535	0.	57.7705	0.	23.7729
21	785.064	5.25700	0.	36.9433	15.0525	32.5293	89.7821	0.	24.9241
22	797.184	5.32831	0.	37.6340	15.9424	29.1384	88.0431	0.	25.9793
23	809.195	5.43449	0.	38.2186	16.5434	27.7909	87.9874	0.	27.0612
24	821.160	5.55344	0.	38.7794	16.9892	27.4467	88.7687	0.	28.1599
25	834.264	5.67429	0.	39.3424	17.3492	27.5794	89.9453	0.	29.2704
26	847.488	5.77777	0.	39.7394	17.6224	28.8138	91.9535	0.	30.3925
27	860.199	5.88955	0.	40.2477	17.8977	29.7393	93.7743	0.	31.5215
28	873.924	6.00134	0.	40.8049	18.1672	30.4815	95.4550	0.	32.6576
29	885.604	6.11483	0.	41.4138	18.4432	31.1387	97.1106	0.	33.8033
30	896.122	6.21920	0.	42.0067	18.6979	31.6868	98.6106	0.	34.9549
31	907.718	6.31395	0.	42.5628	18.9282	32.1625	99.9675	0.	36.1112
32	919.391	6.40729	0.	43.1276	19.1587	32.6239	101.318	0.	37.2747
33	931.167	6.50000	0.	43.7003	19.3895	33.0810	102.671	0.	38.4456
34	943.546	6.59256	0.	44.2791	19.6207	33.5379	104.030	0.	39.6241
35	955.383	6.68704	0.	44.8744	19.8576	34.0055	105.425	0.	40.8110
36	966.806	6.78022	0.	45.4625	20.0905	34.4675	106.801	0.	42.0052
37	979.135	6.87121	0.	46.0362	20.3166	34.9189	108.143	0.	43.2060
38	991.806	6.96558	0.	46.6249	20.5487	35.3819	109.521	0.	44.4152
39	1004.49	7.06100	0.	47.2290	20.7866	35.8558	110.933	0.	45.6330
40	1015.82	7.15615	0.	47.8400	21.0270	36.3341	112.357	0.	46.8592
41	1028.05	7.23977	0.	48.4207	21.2508	36.7868	113.698	0.	48.0914
42	1040.55	7.33141	0.	49.0078	21.4763	37.2442	115.060	0.	49.3315
43	1053.28	7.42851	0.	49.6051	21.7055	37.7090	116.448	0.	50.5798
44	1066.23	7.52908	0.	50.2130	21.9391	38.1817	117.863	0.	51.8365
45	1079.35	7.63178	0.	50.8306	22.1770	38.6621	119.301	0.	53.1019
46	1092.65	7.73577	0.	51.4569	22.4190	39.1497	120.761	0.	54.3759
47	1106.11	7.84058	0.	52.0911	22.6648	39.6442	122.241	0.	55.6588
48	1119.73	7.94598	0.	52.7324	22.9142	40.1453	123.738	0.	56.9506
49	1133.51	8.05190	0.	53.3805	23.1670	40.6528	125.252	0.	58.2516
50	1147.44	8.15835	0.	54.0352	23.4229	41.1666	126.783	0.	59.5617

Figure 2-3. Example for a time path.

```
SETTLEMENT NETWORK MODEL

SETTING OF PARAMETERS:

NR OF VILLAGES:           36
NR OF SERVICES:           5
NR OF OTHER SECTORS:      2

SPACE WEIGHT ALPHA:       2.
INFLUENCE RANGE D0:       5.
REACTION LEVEL A:         .5
CLOSING LEVEL B:          .5

FAMILY SIZE K:            3.
POPULATION GROWTH RATE    1. %
INITIAL POPULATION SIZE:500.

 INITIAL SERVICE SIZES:       5.00000          10.0000          15.0000          20.0000          25.0000
STARTING TIMES:           0 5 10 15 20

PROBABILITY LEVEL PSI1: .166667
PROBABILITY LEVEL PSI2: 1.E+30
CLUSTERING GAMMA:         100.
EMPLOYMENT/DEMAND UNIT:     .400000E-02     .700000E-02     .120000E-01     .170000E-01     .200000E-01
```

Figure 2-4. Listing of parameters.

All utility programs use a common set of subroutines, which are also included in the utility program file.

The utility programs process a result file and produce a listing on an output file. The names of both files may be specified by the user. The programs use FORTRAN units 5 and 6 for communication with the user. The following input will be requested interactively:

request	required response
RESULT FILE NAME	name of the 'OUT' file produced by the simulation program.
LIST FILE NAME	name of the file to receive the listing produced.
PRINT INTERVAL	time interval between two maps. Maps will be produced for those periods t for which t is an integral multiple of the interval specified, (e.g., if 5 is specified, periods 0,5,10,15,...will be printed). This input is required for programs LONG and SHORT.
VILLAGE NUMBER	number x of the village (1 to 36) for which a time path is to be printed. This input is required for program VLIST.

All responses may be entered in free format.

3. STRUCTURE OF THE SIMULATION PROGRAM

The simulation program consists of the following modules:

module name	purpose
SIM	main program, main simulation loop for t.
PARSET	input of simulation parameters.
INIT	generation of the initial state of the model.
STEP	controls the calculations for one period.
STOUT	output of the state for one period
GEMPSV	generation of employment in services for t + 1.

GEMPII	generation of employment in intensive agriculture and industry for t + 1.
GPOP	generation of population for t + 1.
UPDT	updating of status vectors from t to t + 1.
NEWSER	randomized generation of new services.
DIST	distance between two villages.

In addition, IMSLIB routine GGUBS is used to generate random numbers.

The interconnection between these routines is shown in figure 3-1.

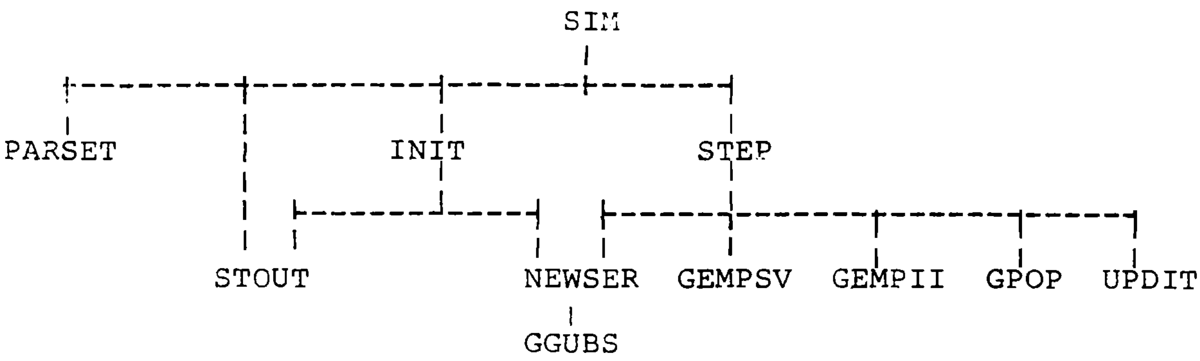

Figure 3-1. Interconnections of modules.

3.1 Major Components of the Program

3.1.1 *Overall Structure of the Simulation*

All calculations in the model are based on the current state as specified by the variables P_t, E_t^u, E_t^v. It is therefore necessary to keep these variables unchanged while the new state vectors P_{t+1}, E_{t+1}^u, and E_{t+1}^v, are computed. In the simulation program, three additional vectors P1, EU1 and EV1 are used to store the new state values, and only after all calculations have been completed, these vectors are copied into the actual state vectors, thereby moving the model into the next period.

3.1.2 *Generation of Employment in Services*

There are two routines which are involved in changing the employment in services: GEMPSV and NEWSER.

Routine GEMPSV performs the deterministic part of the modifications. It will perform modifications in existing services (including the closing of services) and calculate the level at which a new service should be opened. The level of a new service is either E^{u0} for $t = t_{u0}$ or the estimated level $\tilde{D}^u_x$ for later periods. Both the modified levels of existing services and the levels at which new services can be opened are stored in EV1. (This is possible as the sets of villages involved in these two cases form a partition of the set of all villages).

Routine NEWSER randomizes the opening of new services. For each village, the probability of opening the service is computed and a random number between 0 and 1 is generated. If the random number is less than the probability, the service is opened. If it is greater, the service will not be opened and employment for $t + 1$ is set to zero. The routine can determine easily, by looking at the current state, whether a service is to be opened or modified in the village. If $E^u_{x,t}$ is zero, an opening might occur, otherwise an existing service is modified and the value for $E^u_{x,t+1}$ as computed by GEMPSV is left unchanged.

3.1.3 *Distance Between Two Villages*

It can be thought that the villages in the network are at the points of a grid as shown in figure 3-2. A move from one village to the next is equivalent to a relocation by one row and one column (type 1 move) or no row and two columns (type 2 move). Each type of move corresponds to a distance of 4 kilometers.

The distance d between two villages v1 and v2 is computed as the shortest path between the two villages. This shortest path is determined by the following method:

```
column      0   1   2   3   4   5   6   7   8   9  10  11

row
 0          +---1---+---2---+---3---+---4---+---5---+---6
            |   |   |   |   |   |   |   |   |   |   |   |
 1          7---+---8---+---9---+--10---+--11---+--12---+
            |   |   |   |   |   |   |   |   |   |   |   |
 2          +--13---+--14---+--15---+--16---+--17---+--18
            |   |   |   |   |   |   |   |   |   |   |   |
 3         19---+--20---+--21---+--22---+--23---+--24---+
            |   |   |   |   |   |   |   |   |   |   |   |
 4          +--25---+--26---+--27---+--28---+--29---+--30
            |   |   |   |   |   |   |   |   |   |   |   |
 5         31---+--32---+--33---+--34---+--35---+--36---+
```

Figure 3.2. Location of villages in a grid.

Let v1 *be located at* (x1,y1) *and* v2 *at* (x2,y2). *If* $|x1 - x2| \geq |y1 - y2|$ *then* v2 *can be reached from* v1 *by* $|x1 - x2|$ *moves of type* 1.

Proof. From the structure of the network it can be seen that

(3.1-1) $$|y1 - y2| = 2k + (|x1 - x2| \bmod 2)$$

where

$$k \leq \left[\frac{|x1 - x2|}{2} \right]$$

The distance $|y1 - y2|$ can be covered by 2k moves of type 1 in which the y-coordinate is changed by sign(y1 - y2), and $2(\left\lfloor \frac{|x1 - x2|}{2} \right\rfloor - k)$ moves of type 1, in which y is changed first by 1 and then by -1, leaving it unchanged after the two moves. If y1-y2 is odd, x1 - x2 is odd, too, and a final move of type 1 will change both the x and the y coordinate to the desired values.

This method will generate a path of the shortest possible length because it will take $|x1 - x2|$ moves and at least $|x1 - x2|$ are required to cross the distance in rows between v1 and v2.

If $|y1 - y2| > |x1 - x2|$, we may write

$$(3.1\text{-}2) \qquad |y1 - y2| = 2\left\lfloor \frac{|x1 - x2|}{2} \right\rfloor + (|x1 - x2| \bmod 2) + 2n,\ n > 0$$

as the distance between two neighbouring villages in a row is two columns. The distance of

$$2\left\lfloor \frac{|x1 - x2|}{2} \right\rfloor + (|x1 - x2| \bmod 2)$$

is covered by $|x1 - x2|$ moves of type 1 as shown above, the remaining distance in y is covered by

$$(3.1\text{-}3) \qquad n = 0.5 \times (|y1 - y2| - |x1 - x2|)$$

moves of type 2, giving a total of

$$(3.1\text{-}4) \qquad m = |x1 - x2| + n$$

moves of 4 kilometers each. □

4. UTILITY PROGRAMS AND RESULT FILE

The utility program set consists of five main programs and two common subroutines. The connection between subroutines and main programs is shown in the following table, where an x indicates that a subroutine is used by a main program.

main pgm.	subroutines	
	MAP	PRTPAR
ALLVL		x
LONG	x	x
SHORT	x	x
SUMRY		x
VLIST		x

Subroutine MAP is used to print a map of the network of villages on the output file which is referenced via unit number 2. The map contains six lines of numbers, separated by two blank lines. Above and below the map, two blank lines are written. The routine requires two parameters:

parameter	type	description
1	integer	number of villages to be printed, (usually 36).
2	(36) real	vector of values to be printed.

Subroutine PRTPAR opens the result file and the list file and prints the parameter values stored in the result file. Parameter q^u, which is used by program SUMRY, is stored in a common block called PARAM.

The names of the files are requested from the user. The result file is opened with unit number one, the list file with unit number two. Unit numbers five and six are used for communication with the user.

The result file contains 15 records with parameter values, followed by records containing the successive states of the model. The contents of records 1 - 15 are

record	content
1	nr of villages, nr of services, nr of sectors.
2	α, D_0, a, b.
3	k, r, P_0.
4	E^{u0}.
5	t_{u0}.
6	ψ_1, ψ_2, γ.
7-11	$\lambda_{\bar{u}u}$(record 7 for $\bar{u}=1$,...,record 11 for $u=5$).
12	q^u.
13-14	∂_x^v (record 13 for $v=1$, record 14 for $v=2$).
15	starting value for the random number generator.

Each of the following records contains a time index t and P_t, E_t^u and E_t^v for that period.

5. PORTABILITY CONSIDERATIONS

5.1 Software Requirements

All programs are written in ANSI FORTRAN 77 and should be acceptable to any processor supporting this standard. They rely quite strongly on some new features of that standard (particularly the IF-THEN-ELSE construct and some I/O operations), so considerable effort may be required to transform them to conform to the old (1966) standard.

A routine GGUBS from the IMSL program library is used to generate random numbers. If this library is not available, routine GGUBS can be replaced by any routine generating uniformly distributed random numbers in the (0,1) interval. The only call to that routine is located in subroutine NEWSER. Changes in the random number generator might also affect the usage of the starting value variable SEED, which is located in the common block PARAM.

Difficulties might arise from using FORTRAN unit numbers five and six for communication with the user in the utility programs. References to these unit numbers are made in subroutine PRTPAR and in main programs LONG, SHORT and VLIST. These should be changed if necessary.

5.2 Hardware

On the CDC Cyber system, which uses 60 bit arithmetic, rounding errors in the generation of migration were below the tolerance value of 1.e-7. On a system with a smaller wordlength rounding errors may exceed that value.

6. POSSIBLE IMPROVEMENTS OF THE SYSTEM

6.1 Computational Speed

Computing time of the model is considerable and might become unacceptable on slower systems. One possibility to increase the computational speed is changing the calculation of demand in routine GEMPSV. There, the relative attractiveness is calculated as specified in the model as

$$\eta_{xy}^{u} = \frac{A_{xy}^{u}}{\sum_{\bar{x} \neq x} A_{\bar{x}y}^{u} + A_{xy}^{u}}$$

Computing $\sum_{\bar{x} \neq x} A_{xy}^{u}$ for each pair (x,y) requires 0(N**3) additions. Instead, a vector $h_y = \sum_x A_{xy}^{u}$ could be computed once and used in the calculation of η_{xy}^{u}. In the calculation of $\tilde{\eta}_{xy}^{u}$, h_y can be substituted for $\sum_{\bar{x} \neq x} A_{\bar{x}y}^{u}$, as in this case $E_{x,t}^{u} = 0$, so $A_{xy}^{u} = 0$ and therefore

$$\sum_{x} A_{xy}^{u} = \sum_{\bar{x} \neq x} A_{\bar{x}y}^{u} + A_{xy}^{u} = \sum_{\bar{x} \neq x} A_{\bar{x}y}^{u}$$

6.2 Different Geographical Structure of the Network

The behaviour of the model is influenced by the geographical structure of the network via the distance d_{xy} between two villages. This distance is computed in routine DIST. To specify a different structure, routine DIST should be replaced by a new routine which returns the distance of villages in the new network. A simple solution would be a routine which keeps all distances in a table and uses a DATA statement to initialize that table.

To represent the new structure of the network in the outputs, routine MAP in the utility program set has to be changed to print a map of the new network. The programs which use this routine assume that two maps fit on a page and might have to be changed if the maps become too large.

If the new network is to contain more than 36 villages, the dimensions of arrays in all routines have to be changed accordingly.

6.3. Different Initial States of the Model

The initial state of the model as specified in 2.2 is set up by routine INIT. To obtain a more general version of the system, routine INIT should be modified to read the initial state from a file.

APPENDIX

Program listings.

```
      PROGRAM SIM
C     MAIN PROGRAM FOR SETTLEMENT NETWORK SIMULATION
      INTEGER T,FIN
      REAL P(36),EU(5,36),EV(2,36),EU1(5,36),EV1(2,36)
      REAL ALPHA,A,B,K,TAU,PO,EUO(5),PSI1(5,36),PSI2,LAMBDA(5,5),QU(5),
     *      DEL(2,36),DO,GAMMA
      INTEGER N,NU,NV,TUO(5)
      DOUBLE PRECISION SEED
      COMMON /PARAM/ SEED,N,NU,NV,ALPHA,A,B,K,TAU,EUO,TUO,
     *               PSI1,PSI2,LAMBDA,QU,DEL,DO,PO,GAMMA
      SAVE /PARAM/
C     READ SIMULATION PARAMETERS
      CALL PARSET(FIN)
C     GENERATE INITIAL STATE OF MODEL
      CALL INIT(P,EU,EV,EU1,EV1)
C     MAIN SIMULATION LOOP FOR ALL PERIODS
      DO 5 T=1,FIN
         WRITE(*,*) 'PROCESSING TIME T=',T
C     COMPUTATIONS
         CALL STEP(T,P,EU,EV,EU1,EV1)
C     OUTPUT
         CALL STOUT(T,P,EU,EV)
5        CONTINUE
      STOP
      END
```

```
      SUBROUTINE PARSET(FIN)
C   THIS ROUTINE READS THE PARAMETER FILES AND STORES THE PARAMETERS
C   IN COMMON BLOCK /PARAM/
C   MOST PARAMETERS ARE ALSO WRITTEN TO OUTPUT FILE OUT FOR
C   LATER PRINTING
      CHARACTER*8 F1,F2,F3
      INTEGER I,J,DH(36)
      REAL W
      REAL ALPHA,A,B,K,TAU,PO,EUO(5),PSI1(5,36),PSI2,LAMBDA(5,5),QU(5),
     *       DEL(2,36),DO,GAMMA
      INTEGER N,NU,NV,TUO(5),FIN
      DOUBLE PRECISION SEED
      COMMON /PARAM/ SEED,N,NU,NV,ALPHA,A,B,K,TAU,EUO,TUO,
     *                PSI1,PSI2,LAMBDA,QU,DEL,DO,PO,GAMMA
      SAVE /PARAM/
      OPEN(UNIT=1,FILE='PAR')
      OPEN(UNIT=2,FILE='OUT',FORM='UNFORMATTED')
      READ(1,*) N,NU,NV
      WRITE(2)  N,NU,NV
      READ(1,*) ALPHA,DO,A,B
      WRITE(2)  ALPHA,DO,A,B
      READ(1,*) K,TAU,PO
      WRITE(2)  K,TAU,PO
      READ(1,*) (EUO(I),I=1,NU)
      WRITE(2)  (EUO(I),I=1,NU)
      READ(1,*) (TUO(I),I=1,NU)
      WRITE(2)  (TUO(I),I=1,NU)
      READ(1,*) W,PSI2,GAMMA
      IF(W.GE.0) THEN
         DO 3 I=1,NU
            DO 3 J=1,N
               PSI1(I,J)=W
3              CONTINUE
      ELSE
         DO 4 I=1,N
            READ(1,*) (PSI1(J,I),J=1,NU)
4           CONTINUE
            W=-1.0
      END IF
      WRITE(2)  W,PSI2,GAMMA
      DO 5 I=1,NU
         READ(1,*) (LAMBDA(I,J),J=1,NU)
         WRITE(2)  (LAMBDA(I,J),J=1,NU)
5        CONTINUE
      READ(1,*) (QU(I),I=1,NU)
      WRITE(2)  (QU(I),I=1,NU)
      DO 15 I=1,NV
         READ(1,'(F5.3,1X,36I1)') W,(DH(J),J=1,N)
         DO 10 J=1,N
            DEL(I,J)=W*DH(J)
10          CONTINUE
         WRITE(2)  (DEL(I,J),J=1,N)
15       CONTINUE
      READ(1,*) SEED
      WRITE(2) SEED
      READ(1,*) FIN


      RETURN
      END
```

```
      SUBROUTINE INIT(P,EU,EV,EU1,EV1)
C   GENERATE THE INITIAL STATE OF THE MODEL
      REAL P(36),EU(5,36),EV(2,36),EU1(5,36),EV1(2,36)
      INTEGER X,U,V
      REAL ALPHA,A,B,K,TAU,PO,EUO(5),PSI1(5,36),PSI2,LAMBDA(5,5),QU(5)
     *     DEL(2,36),DO,GAMMA
      INTEGER N,NU,NV,TUO(5)
      DOUBLE PRECISION SEED
      COMMON /PARAM/ SEED,N,NU,NV,ALPHA,A,B,K,TAU,EUO,TUO,
     *                 PSI1,PSI2,LAMBDA,QU,DEL,DO,PO,GAMMA
      SAVE /PARAM/
      DO 15 X=1,N
C   SAME POPULATION FOR ALL VILLAGES
         P(X)=PO
         DO 5 U=1,NU
C   EMPLOYMENT IN TIME -1 IS 0 FOR ALL SERVICES
            EU(U,X)=0.0
C   FOR SERVICES WITH TUO = 0 , EMPLOYMENT MAY BE SET TO EUO
C   OTHERWISE, EMPLOYMENT IS 0
            IF(TUO(U).EQ.0) THEN
               EU1(U,X)=EUO(U)
            ELSE
               EU1(U,X)=0.0
            END IF
5           CONTINUE
C   EMPLOYMENT IN OTHER SECTORS IS 0
         DO 10 V=1,NV
            EV(V,X)=0.0
            EV1(V,X)=0.0
10          CONTINUE
15       CONTINUE
C   PERFORM RANDOM OPENING OF SERVICES WITH TUO = 0
      CALL NEWSER(0,EU,P,EU1)
      DO 20 X=1,N
         DO 20 U=1,NU
            EU(U,X)=EU1(U,X)
20          CONTINUE
C   OUTPUT THE STATE AT T=0
      CALL STOUT(0,P,EU,EV)
      RETURN
      END
```

SUBROUTINE STEP   73/73   OPT=0   FTN 5.0+518

```
      SUBROUTINE STEP(T,P,EU,EV,EU1,EV1)
C   TRANSITION FROM T TO T+1
      INTEGER T
      REAL P(36),EU(5,36),EV(2,36),DEU(5,36),DEV(2,36),
     *      EU1(5,36),EV1(2,36),P1(36)
C   CHANGE EMPLOYMENT IN SERVICES
      CALL GEMPSV(T,P,EU,EU1)
C   RANDOMIZATION OF OPENING OF NEW SERVICES
      CALL NEWSER(T,EU,P,EU1)
C   GENERATE EMPLOYMENT IN INDUSTRY+INTENSIVE AGRICULTURE
      CALL GEMPII(T,P,EU,EV,EV1)
C   GENERATE POPULATION
      CALL GPOP(T,P,EU,EU1,EV,EV1,P1)
C   UPDATE STATUS VECTORS
      CALL UPDT(T,EU,EU1,EV,EV1,P,P1)
      RETURN
      END
```

SUBROUTINE STOUT   73/73   OPT=0   FTN 5.0+518

```
      SUBROUTINE STOUT(T,P,EU,EV)
C   WRITE THE STATUS VECTORS TO OUTPUT FILE
      INTEGER T
      REAL P(36),EU(5,36),EV(2,36)
      WRITE(2) T,P,EU,EV
      RETURN
      END
```

```
      SUBROUTINE GEMPSV(T,P,EU,E1)
C     THIS SUBROUTINE COMPUTES THE EMPLOYMENT IN SERVICES
C     AT TIME T+1
C     INPUT: STATE VARIABLES P AND EU AND TIME T
C     OUTPUT: EMPLOYMENT AT T+1 STOPED IN E1
C             NOTE THAT E1 WILL CONTAIN THE POSSIBLE LEVEL
C             OF EMPLOYMENT FOR OPENING OF NEW SERVICES
C             RANDOMIZATION OF OPENING IS DONE BY NEWSER
C             WHICH WILL BE CALLED AFTERWARDS

      INTEGER T,U,UB,X,XB,Y
      REAL P(36),EU(5,36),E1(5,36)
      REAL Z1,ATTR(36,36),AS(36,36),S1,S2,H,EE,LIMIT,DIST,D
      REAL ALPHA,A,B,K,TAU,PO,EUO(5),PSI1(5,36),PSI2,LAMBDA(5,5),OU(5),
     *       DEL(2,36),DO,GAMMA
      INTEGER N,NU,NV,TUO(5)
      DOUBLE PRECISION SEED
      COMMON /PARAM/ SEED,N,NU,NV,ALPHA,A,B,K,TAU,EUO,TUO,
     *                  PSI1,PSI2,LAMBDA,OU,DEL,DO,PO,GAMMA
      SAVE /PARAM/
C     THERE ARE 3 CASES FOR T <,=,> TUO

      DO 40 U=1,NU

C     THE MOST IMPORTANT CASE IS T > TUO

         IF(T.GT.TUO(U)) THEN

C     COMPUTJE ATTRACTIVENESS ATTR AND POSSIBLE ATTRACTIVENESS AS
C     FOR EACH PAIR OF VILLAGES <X,Y>

            DO 15 X=1,N
               Z1=0.0
               DO 5 UB=1,NU
                  IF(UB.EQ.U) THEN
                     Z1=Z1+EU(U,X)
                  ELSE
                     Z1=Z1+LAMBDA(UB,U)*EU(UB,X)
                  END IF
5                 CONTINUE
               DO 10 Y=1,N
                  D=DIST(X,Y)
                  IF(D.LE.DO) THEN
                     ATTR(X,Y)=Z1/(D**ALPHA)
                     IF(EU(U,X).EQ.0.0) ATTR(X,Y)=0.0
                     AS(X,Y)=(Z1+EUO(U))/(D**ALPHA)
                  ELSE
                     ATTR(X,Y)=0.0
                     AS(X,Y)=0.0
                  END IF
10                CONTINUE
15             CONTINUE

C     COMPUTE RELATIVE ATTRACTIVENESS ETA AND FROM IT THE
C     REQUIRED LEVEL OF EMPLOYMENT EE
```

```
               DO 30 X=1,N
                  S2=0.0
                  DO 25 Y=1,N
                     S1=0.0
                     DO 20 XB=1,N
                        IF(XB.NE.X) S1=S1+ATTR(XB,Y)
 20                     CONTINUE
                     IF(EU(U,X).NE.0.0) THEN
                        H=ATTR(X,Y)
                     ELSE
                        H=AS(X,Y)
                     END IF
                     IF(H.GT.0.0) S2=S2+P(Y)*QU(U)*(H/(S1+H))
 25                  CONTINUE
                  EE=EU(U,X)+A*(S2-EU(U,X))

C   CHECK IF THE SERVICE WILL BECOME TO SMALL AND HAS TO BE SHUT
C   OR IF THE LEVEL OF DEMAND IS TO SMALL TO OPEN A NEW SERVICE

                  LIMIT=EUO(U)
                  IF(EU(U,X).GT.0.0) LIMIT=B*LIMIT
                  IF(EE.LT.LIMIT) EE=0.0
                  E1(U,X)=EE
 30               CONTINUE

C   CASE 2: T = TUO
C   IN THIS CASE, E1 IS SET TO EUO FOR ALL VILLAGES
C   RANDOMIZATION OF OPENING WILL BE DONE BY ROUTINE NEWSER

            ELSE IF(T.EQ.TUO(U)) THEN
               DO 33 X=1,N
                  E1(U,X)=EUO(U)
 33               CONTINUE

C   CASE 3: T < TUO
C   EMPLOYMENT WILL BE KEPT AT LEVEL ZERO

            ELSE
               DO 35 X=1,N
                  E1(U,X)=0.0
 35               CONTINUE
            END IF
 40         CONTINUE
         RETURN
         END
```

SUBROUTINE GEMPII   73/73   OPT=0   FTN 5.0+518

```
      SUBROUTINE GEMPII(T,P,EU,EV,E1)

C     THIS SUBROUTINE COMPUTES THE EMPLOYMENT IN INTENSIVE AGRICULTURE
C     AND INDUSTRY

C     INPUT: TIME T AND STATE VARIABLES P EU EV
C     OUTPUT: EV AT TIME T+1 STORED IN E1

      INTEGER X,U,V,T
      REAL P(36),EU(5,36),EV(2,36),DEV(2,36),E1(2,36),S

C     DECLARATRION OF MODEL PARAMETERS AND THEIR COMMON BLOCK

      REAL ALPHA,A,B,K,TAU,PO,EUO(5),PSI1(5,36),PSI2,LAMBDA(5,5),QU(5),
     *        DEL(2,36),DO,GAMMA
      INTEGER N,NU,NV,TUO(5)
      DOUBLE PRECISION SEED
      COMMON /PARAM/ SEED,N,NU,NV,ALPHA,A,B,K,TAU,EUO,TUO,
     *                  PSI1,PSI2,LAMBDA,QU,DEL,DO,PO,GAMMA
      SAVE /PARAM/

C     COMPUTE UNEMPLOYED LABOUR FORCE
C     AS POPULATION/FAMILYSIZE -EMPLOYMENT(SERVICE) - EMPLOYMENT(SECT)
      S=0.0
      DO 15 X=1,N
         S=S+P(X)/K
         DO 5 U=1,NU
            S=S-EU(U,X)
5           CONTINUE
         DO 10 V=1,NV
            S=S-EV(V,X)
10          CONTINUE
15       CONTINUE

C     SPREAD AMONG SECTORS

      DO 20 V=1,NV
         DO 20 X=1,N
            DEV(V,X)=DEL(V,X)*S
            E1(V,X)=EV(V,X)+DEV(V,X)
20          CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GPOP(T,P,EU,EU1,EV,EV1,P1)

C     COMPUTE POPULATION AT TIME T+1

      INTEGER T,U,V,X
      REAL DEU(5,36),DEV(2,36),P(36),P1(36),MS(36),GAM1,PS(36),
     *      PTS,SM,MX,S,EU(5,36),EU1(5,36),EV(2,36),EV1(2,36)
      REAL ALPHA,A,B,K,TAU,P0,EU0(5),PSI1(5,36),PSI2,LAMBDA(5,5),QU(5)
     *      DEL(2,36),D0,GAMMA
      INTEGER N,NU,NV,TU0(5)
      DOUBLE PRECISION SEED
      COMMON /PARAM/ SEED,N,NU,NV,ALPHA,A,B,K,TAU,EU0,TU0,
     *              PSI1,PSI2,LAMBDA,QU,DEL,D0,P0,GAMMA
      SAVE /PARAM/

C     COMPUTE CHANGES IN EMPLOYMENT

      DO 4 X=1,N
         DO 2 U=1,NU
            DEU(U,X)=EU1(U,X)-EU(U,X)
2           CONTINUE
         DO 3 V=1,NV
            DEV(V,X)=EV1(V,X)-EV(V,X)
3           CONTINUE
4        CONTINUE

C     COMPUTE POSSIBLE MIGRATION AND SUMS

      PTS=0.0
      SM=0.0
      GAM1=0.0
      DO 15 X=1,N
         S=0.0
         DO 5 U=1,NU
            S=S+DEU(U,X)
5           CONTINUE
         DO 10 V=1,NV
            S=S+DEV(V,X)
10          CONTINUE
         MS(X)=K*S
         PS(X)=P(X)-MS(X)
         PTS=PTS+PS(X)
         GAM1=GAM1+MS(X)
15       CONTINUE
      GAM1=GAM1/PTS

C     COMPUTE ACTUAL MIGRATION, NEW POPULATION
C     AND SUM OF MIGRATION FLOWS

      SM=0.0
      DO 20 X=1,N
         MX=MS(X)-GAM1*PS(X)
         SM=SM+MX
         P1(X)=(1.0+TAU)*P(X)+MX
20       CONTINUE

C     CHECK IF MIGRATION FLOWS SUM UP TO ZERO

      IF(ABS(SM).GT.1.E-7) WRITE(*,*) ' MIGRATION ERROR,SUM=',SM
      RETURN
      END
```

SUBROUTINE UPDT   73/73   OPT=0   FTN 5.0+518

```
      SUBROUTINE UPDT(T,EU,EU1,EV,EV1,P,P1)
C   MOVE THE STATUS VECTORS OF T+1 (EU1,EV1,P1)
C   TO THE CURRENT STATUS VECTORS EU, EV AND P
      INTEGER T,X,U,V
      REAL EU(5,36),EU1(5,36),EV(2,36),EV1(2,36),P(36),P1(36)
      REAL ALPHA,A,B,K,TAU,P0,EU0(5),PSI1(5,36),PSI2,LAMBDA(5,5),QU(5),
     *     DEL(2,36),D0,GAMMA
      INTEGER N,NU,NV,TU0(5)
      DOUBLE PRECISION SEED
      COMMON /PARAM/ SEED,N,NU,NV,ALPHA,A,B,K,TAU,EU0,TU0,
     *               PSI1,PSI2,LAMBDA,QU,DEL,D0,P0,GAMMA
      SAVE /PARAM/
      DO 15 X=1,N
         P(X)=P1(X)
         DO 5 U=1,NU
            EU(U,X)=EU1(U,X)
5           CONTINUE
         DO 10 V=1,NV
            EV(V,X)=EV1(V,X)
10          CONTINUE
15       CONTINUE
      RETURN
      END
```

```
      SUBROUTINE NEWSER(T,EU,P,EU1)

C     RANDOMIZED OPENING OF SERVICES
C     INPUT: TIME T, OLD STATE EU AND P
C            POSSIBLE LEVEL OF SERVICES IN E1
C     OUTPUT: NEW LEVEL OF SERVICES IN E1, WHERE OPENINGS OF NEW
C             SERVICES ARE DROPPED RANDOMLY FROM PREVIOUS E1

      INTEGER NR,U,X,T,UU
      REAL P(36),EU(5,36),EU1(5,36),RR,RX(36),R(1),PROB
      REAL ALPHA,A,B,K,TAU,P0,EU0(5),PSI1(5,36),PSI2,LAMBDA(5,5),OU(5),
     *     DEL(2,36),D0,GAMMA
      INTEGER N,NU,NV,TU0(5)
      DOUBLE PRECISION SEED
      COMMON /PARAM/ SEED,N,NU,NV,ALPHA,A,B,K,TAU,EU0,TU0,
     *               PSI1,PSI2,LAMBDA,QU,DEL,D0,P0,GAMMA
      SAVE /PARAM/
      NR=1
      DO 15 U=1,NU

C     OPENING MAY OCCUR ONLY AT T > TU0

         IF(T.GE.TU0(U)) THEN

C     COMPUTE PROBABILITY MODIFIER TERM FOR VILLAGES

            RR=0.0
            DO 5 X=1,N
               RX(X)=0.0
               DO 3 UU=1,NU
                  RX(X)=RX(X)+EU(U,X)
3                 CONTINUE
               RR=RR+P(X)+GAMMA*RX(X)
5              CONTINUE

            DO 10 X=1,N

C     USE DIFFERENT PSI FOR T=TU0 AND LATER

               IF(T.EQ.TU0(U)) THEN
                  PROB=PSI1(U,X)*N*(P(X)+GAMMA*RX(X))/RR
               ELSE
                  PROB=PSI2*N*(P(X)+GAMMA*RX(X))/RR
               END IF

C     SUBJECT TO RANDOMIZATION IF EU = 0
C     (I.E. IF IT IS AN OPENING AND NOT A MODIFICATION OF SERVICE)

               CALL GGUBS(SEED,NR,R)
               IF(R(1).GE.PROB.AND.EU(U,X).EQ.0.0) EU1(U,X)=0.0
10             CONTINUE
         END IF
15       CONTINUE
      RETURN
      END
```

FUNCTION DIST 73/73 OPT=0 FTN 5.0+518

```
      REAL FUNCTION DIST(I,J)
C   THIS FUNCTION IS USED TO COMPUTE THE DISTANCE BETWEEN TWO
C   VILLAGES IN THE NETWORK
      INTEGER I,J,I1,J1,X1,X2,Y1,Y2,DX,DY
C   THE DISTANCE WITHIN A VILLAGE IS SET TO 0.4
      IF(I.EQ.J) THEN
         DIST=0.4
         RETURN
C   COMPUTE THE DISTANCE BETWEEN DIFFERENT VILLAGES
      ELSE
         I1=I-1
         J1=J-1
C   X1 IS THE ROW NUMBER OF VILLAGE 1, Y1 ITS COLUMN NUMBER
C   NOTE THAT VILLAGES ARE AT ODD COLOUMNS IN EVEN ROWS AND AT
C   EVEN COLOUMNS IN ODD ROWS
         X1=I1/6
         Y1=2*MOD(I1,6)+MOD(X1+1,2)
         X2=J1/6
         Y2=2*MOD(J1,6)+MOD(X2+1,2)
C   COMPUTE DISTANCES IN ROWS AND COLOUMNS
C   ONLY COLOUMN DISTANCES > ROW DISTANCE ARE OF INTEREST BECAUSE
C   SMALLER ROW DISTANCES CAN BE CROSSED BY MOVING DIAGONALLY
         DX=ABS(X1-X2)
         DY=ABS(Y1-Y2)-DX
         IF(DY.LT.0) DY=0
C   CORRECT FOR THE FACT THAT COLOUMN DISTANCES ARE NOW 2*ROW DISTANCES
C   BECAUSE OF THE ODD/EVEN COLOUMN LOCATION OF VILLAGES
         DY=DY/2
C   DISTANCE BETWEEN TWO ADJACENT VILLAGES IS 4 KM
         DIST=4.0*(DX+DY)
         RETURN
      END IF
      END
```

```
PROGRAM ALLVL       73/73    OPT=0                                  FTN 5.0+518

          PROGRAM ALLVL
          INTEGER T,N,I
          REAL P(36),EU(5,36),EV(2,36),S
          CALL PRTPAR
          DO 15 N=1,36
          WRITE(2,'(''1'',A,I5)') 'TIME PATH FOR VILLAGE',N
          WRITE(2,101)
          WRITE(2,102) 1,2,3,4,5,1,2
    5     READ(1,END=90) T,P,EU,EV
          S=0.0
          DO 7  I=1,5
             S=S+EU(I,N)
    7        CONTINUE
          WRITE(2,103) T,P(N),(EU(I,N),I=1,5),S,(EV(I,N),I=1,2)
          GOTO 5
    90    REWIND 1
          DO 10 I=1,15
             READ (1)
    10       CONTINUE
    15    CONTINUE
          STOP
    101   FORMAT(//52X,'EMPLOYMENT IN SERVICE'
         * ,32X,'EMPLOYMENT IN SECTOR')
    102   FORMAT('      TIME  POPULATION ',I9,4I13,'     SUM      ',2I13)
    103   FORMAT(I10,9G13.6)
          END
```

```
PROGRAM LONG        73/73   OPT=0                                FTN 5.0+518

          PROGRAM LONG

C      COMPLETE SET OF MAPS (POPULATION, SERVICES, OTHER SECTORS)

          INTEGER T,X,U,V,I
          REAL P(36),EU(5,36),EV(2,36),VAL(36)

C      PERFORM INITIALIZATION

          CALL PRTPAR
          WRITE(6,*) 'ENTER PRINT INTERVAL: '
          READ(5,*) I

C      LOOP FOR ALL TIME PERIODS

5         READ(1,END=90) T,P,EU,EV
          IF(MOD(T,I).NE.0) GOTO 5

C      MAP OF POPULATION

          WRITE(2,'(''1'',A,I5)') '   POPULATION AT TIME ',T
          CALL MAP(36,P)

C      MAPS OF SERVICES

          DO 15 U=1,5
             IF(MOD(U,2).EQ.1) WRITE(2,'(''1'')')
             DO 10 X=1,36
                VAL(X)=EU(U,X)
10           CONTINUE
             WRITE(2,*) '   EMPLOYMENT IN SERVICE ',U,' AT TIME ',T
             CALL MAP(36,VAL)
15           CONTINUE

C      MAPS OF OTHER SECTORS

          WRITE(2,'(''1'')')
          DO 25 V=1,2
             DO 20 X=1,36
                VAL(X)=EV(V,X)
20              CONTINUE
             WRITE(2,*) '   EMPLOYMENT IN SECTOR ',V,' AT TIME ',T
             CALL MAP(36,VAL)
25           CONTINUE
          GOTO 5
90        STOP
          END
```

```
      PROGRAM SHORT

C     SHORT MAPS (POPULATION AND OTHER SECTORS)
C     USED FOR THE BASE SOLUTION

      INTEGER T,X,U,V,I
      REAL P(36),EU(5,36),EV(2,36),VAL(36)

C     PERFORM INITIALIZATION

      CALL PRTPAR
      WRITE(6,*) 'ENTER PRINT INTERVAL: '
      READ(5,*) I

C     LOOP FOR ALL PERIODS

5     READ(1,END=90) T,P,EU,EV
      IF(MOD(T,I).NE.0) GOTO 5
      WRITE(2,'(''1'',A,I5)') '    POPULATION AT TIME ',T

C     MAP OF POPULATION

      CALL MAP(36,P)
      WRITE(2,'(''1'')')

C     MAPS OF OTHER SECTORS

      DO 25 V=1,2
         DO 20 X=1,36
            VAL(X)=EV(V,X)
20       CONTINUE
         WRITE(2,*) '    EMPLOYMENT IN SECTOR ',V,' AT TIME ',T
         CALL MAP(36,VAL)
25       CONTINUE
      GOTO 5
90    STOP
      END
```

```
      PROGRAM SUMRY

C     SUMMARY OUTPUT FOR THE WHOLE SYSTEM

      INTEGER T,I,J
      REAL P(36),PS,EU(5,36),EUS(5),EV(2,36),EVS(2),S,QU(5)
      COMMON /PARAM/ QU

C     PERFORM INITIALIZATION

      CALL PRTPAR
      WRITE(2,'(''1'',A)') 'TIME PATH FOR THE WHOLE SYSTEM'
      WRITE(2,101)
      WRITE(2,102) 1,2,3,4,5,1,2

C     TIME PATH FOR THE SYSTEM

5     READ(1,END=25) T,P,EU,EV
      DO 10 I=1,5
         EUS(I)=0.0
10       CONTINUE
      EVS(1)=0.0
      EVS(2)=0.0
      PS=0.0
      S=0.0
      DO 20 J=1,36
         DO 15 I=1,5
            EUS(I)=EUS(I)+EU(I,J)
            S=S+EU(I,J)
15          CONTINUE
         EVS(1)=EVS(1)+EV(1,J)
         EVS(2)=EVS(2)+EV(2,J)
         PS=PS+P(J)
20       CONTINUE
      WRITE(2,'(I10,9G13.6)') T,PS,(EUS(J),J=1,5),S,(EVS(J),J=1,2)
      GOTO 5
25    CONTINUE

C     DEMAND SATISFACTION COEFFINCIENTS

      REWIND 1
      DO 30 I=1,15
         READ(1)
30       CONTINUE
      WRITE(2,'(''1'',A)') 'DEMAND SATISFACTION COEFFICIENTS'
      WRITE(2,'(/A/)') '        TIME       COEFFICIENTS'
35    READ(1,END=90) T,P,EU,EV
      DO 40 I=1,5
         EUS(I)=0.0
40       CONTINUE
      PS=0.0
      DO 50 J=1,36
         PS=PS+P(J)
         DO 45 I=1,5
            EUS(I)=EUS(I)+EU(I,J)
45          CONTINUE
50       CONTINUE
```

```
      DO 55 I=1,5
         EUS(I)=EUS(I)/(PS*OU(I))
55       CONTINUE
      WRITE(2,'(I10,5G13.6)') T,(EUS(I),I=1,5)
      GOTO 35
90    STOP
101   FORMAT(//52X,'EMPLOYMENT IN SERVICE'
     * ,32X,'EMPLOYMENT IN SECTOR')
102   FORMAT('       TIME  POPULATION ',I9,4I13,'      SUM       ',2I13)
      END
```

PROGRAM VLIST 73/73 OPT=0 FTN 5.0+518

```
      PROGRAM VLIST
C     TIME PATH FOR ONE VILLAGE
      INTEGER T,N,I
      REAL P(36),EU(5,36),EV(2,36)
C     PERFORM INITIALIZATION
      WRITE(6,*) 'ENTER VILLAGE NUMBER: '
      READ(5,*) N
      CALL PRTPAR
C     PRINT PAGE HAEDING
      WRITE(2,'(''1'',A,I5)') 'TIME PATH FOR VILLAGE',N
      WRITE(2,101)
      WRITE(2,102) 1,2,3,4,5,1,2
C     LOOP FOR ALL TIME PERIODS
5     READ(1,END=90) T,P,EU,EV
C     SUM OF SERVICES
      S=0.0
      DO 10 I=1,5
         S=S+EU(I,N)
10       CONTINUE
      WRITE(2,103) T,P(N),(EU(I,N),I=1,5),S,(EV(I,N),I=1,2)
      GOTO 5
90    STOP
101   FORMAT(//52X,'EMPLOYMENT IN SERVICE'
     * ,32X,'EMPLOYMENT IN SECTOR')
102   FORMAT('        TIME    POPULATION   ',I9,2I13,'      SUM      ',4I13)
103   FORMAT(/I10,9G13.6)
      END
```

```
      SUBROUTINE MAP(N,VAL)

C   PRINT A MAP OF THE VALUES PASSED IN VAL

      INTEGER N,I,J
      REAL VAL(36)
      WRITE(2,'(//)')
      DO 5 I=0,(N/6)-1
         IF(MOD(I,2).EQ.0) THEN
            WRITE(2,101) (VAL(6*I+J),J=1,6)
         ELSE
            WRITE(2,102) (VAL(6*I+J),J=1,6)
         END IF
5       CONTINUE
      WRITE(2,'(//)')
101    FORMAT(//T10,6G15.6)
102    FORMAT(//T20,6G15.6)
      RETURN
      END
```

```
      SUBROUTINE PRTPAR
C   READ AND PRINT THE PARAMETER PART OF THE RESULT FILE
      CHARACTER*8 F1,F2
      INTEGER I,J
      REAL ALPHA,A,B,K,TAU,P0,EU0(5),PSI1,PSI2,LAMBDA(5,5),QU(5),
     *       DEL(2,36),D0,GAMMA
      INTEGER N,NU,NV,INTYP,TU0(5)
      DOUBLE PRECISION SEED
      COMMON /PARAM/ QU
      SAVE /PARAM/
      WRITE(6,*) 'ENTER RESULT FILE NAME: '
      READ(5,*) F1
      OPEN(UNIT=1,FILE=F1,FORM='UNFORMATTED')
      REWIND 1
      WRITE(6,*) 'ENTER LIST FILE NAME: '
      READ(5,*) F2
      OPEN(UNIT=2,FILE=F2)
      WRITE(2,'(A)') '1      SETTLEMENT NETWORK MODEL'
      WRITE(2,'(A/)') '0      SETTING OF PARAMETERS:'
      READ(1) N,NU,NV
      WRITE(2,*)  '      NR OF VILLAGES:          ',N
      WRITE(2,*)  '      NR OF SERVICES:          ',NU
      WRITE(2,*)  '      NR OF OTHER SECTORS:     ',NV
      WRITE(2,*)
      READ(1) ALPHA,D0,A,B
      WRITE(2,*)  '      SPACE WEIGHT ALPHA:      ',ALPHA
      WRITE(2,*)  '      INFLUENCE RANGE D0:      ',D0
      WRITE(2,*)  '      REACTION LEVEL A:        ',A
      WRITE(2,*)  '      CLOSING LEVEL B:         ',B
      WRITE(2,*)
      READ(1) K,TAU,P0
      WRITE(2,*)  '      FAMILY SIZE K:           ',K
      WRITE(2,*)  '      POPULATION GROWTH RATE   ',TAU*100.0,' %'
      WRITE(2,*)  '      INITIAL POPULATION SIZE:',P0
      WRITE(2,*)
      READ(1) (EU0(I),I=1,NU)
      WRITE(2,101)  '         INITIAL SERVICE SIZES:  ',(EU0(I),I=1,NU)
      READ(1) (TU0(I),I=1,NU)
      WRITE(2,*)  '      STARTING TIMES:          ',(TU0(I),I=1,NU)
      WRITE(2,*)
      READ(1) PSI1,PSI2,GAMMA
      WRITE(2,*)  '      PROBABILITY LEVEL PSI1: ',PSI1
      WRITE(2,*)  '      PROBABILITY LEVEL PSI2: ',PSI2
      WRITE(2,*)  '      CLUSTERING GAMMA:        ',GAMMA
      DO 5 I=1,NU
         READ(1) (LAMBDA(I,J),J=1,NU)
5        CONTINUE
      READ(1) (QU(I),I=1,NU)
      WRITE(2,101) '      EMPLOYMENT/DEMAND UNIT:',(QU(I),I=1,NU)
      DO 15 I=1,NV
         READ(1) (DEL(I,J),J=1,N)
15       CONTINUE
      READ(1) SEED
101   FORMAT(A,6G15.6)
      RETURN

      END
```

REFERENCES

Domanski, R., and A. Wierzbicki (forthcoming) A simulation model for restructuring a rural settlement network.