Yugoslav Journal of Operations Research
13 (2003), Number 1, 107-130

A TECHNIQUE FOR IMPROVING READABILITY OF FORRESTER DIAGRAM IN SYSTEM DYNAMICS

Wei-Tzen YANG

Graduate Institute of Management Sciences
Tamkang University, Tamsui, Taipei Hsien, Taiwan 25137
g5450022@tkgis.tku.edu.tw

Nai-Wen KUO

Department of Information Management
Chinese Culture University, Yang Ming Shan, Taipei, Taiwan 111
chingneven@giga.net.tw

Abstract: We describe a three-pass algorithm for improving the readability of Forrester Diagram in system dynamics. The first pass converts Forrester Diagram to recurrent hierarchy. The second pass sorts the vertices on each level, with the goal of minimizing crossings. The third pass is a finite tuning of the layout that determines the horizontal positions of vertices. An illustrative example is given to verify the result.

Keywords: System Dynamics (SD), Forrester Diagram (FD), recurrent hierarchies, readability, algorithm.

1. INTRODUCTION

System Dynamics (SD), developed by Forrester [1], is useful in the study of continuous dynamical system, which has been mainly applied to the simulation of social and managerial systems [1, 3, 5, 9-14, 16, 20]. Simulation packages such as DYSMAP, STELLA and I-THINK are used in system dynamics helping to formulate the data flow of system and to simulate the results of models for user. But, whenever system is fluctuating or the size of problems becomes larger, it is difficult to manipulate Forrester Diagram (FD) by the modeler.

Generally speaking, it is difficult to analyze the structure of a digraph readily unless vertices are laid out in some regular form edges are drawn in such a form that paths can be readily traced by human eyes. To consider common aspects of readability we can base it on the following aesthetic principles:

1. Hierarchical layout of vertices.
2. Avoid edge crossing and sharp bends.

3. Keep edges short.
4. Favor symmetry and balance.

This paper is intended to present a method for converting FD to a visually understandable drawing of hierarchies. This is a visual aid to show the relations of all vertices, and to understand the structures of complex systems. It enables us to add or delete elements and couplings over a Forrester Diagram more clearly, and will be useful in developing the result of simulation more accurately and conveniently.

A two dimensional diagram for system dynamics contains cycles, but a cycle contains at least one level variable (variables denotes by rectangular vertices due to the semantic restrictions stated in an FD). Therefore, the cycles can be eliminated by drawing the level variables twice as described in the following section.

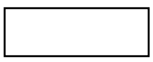
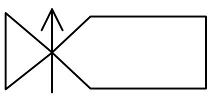
The definition of recurrent hierarchies in section 2 enables the application of this study to the hierarchy. The readability-improving algorithm has three passes, as shown in Figure 1. The first pass assigns vertices to level. Each vertex can be determined only after all the higher level vertices that are connected to the vertex have been determined. The second pass sets the order of vertices with levels to avoid edge crossings. We sort vertices by medians and transpose adjacent vertices. The final pass sets the actual layout coordinates of vertices. According to priority numbers, vertices can be adjusted to the best position. We will discuss the three passes in sections 3-5. An illustrative example is given to demonstrate its application in section 6. The final section concludes our discussion and future work.

```
procedure improve_read()
begin
   level_assign();
   ordering();
   position();
end
```

Figure 1. Main algorithm

2. BASIC DEFINITIONS

The enumeration of the semantic restrictions stated in an FD, is derived from some results of Dolado and Torrealdea [4]. Each components of an FD can be drawn symbolically in Figure 2.

Graphic Symbol	Set Symbol	Name
	X	level
	R	rate

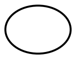
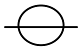
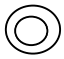
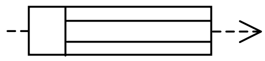
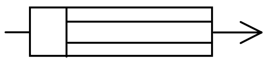
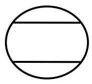
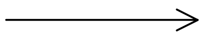
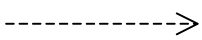
	V	auxiliary
	P	parameter
	U	input
	D	information delay
	E	material delay
	M	multiplier
	F	flow coupling
	I	information coupling

Figure 2. Symbolic representations of quantity and coupling in FD

Definition 1. *Properties of each element in FD are defined as:*

(a) *levels: for all* $q_j \in X \Rightarrow A_c(q_j) \subseteq F \cap E_c(q_j) \subseteq I$,

(b) *rates: for all* $q_j \in R \Rightarrow A_c(q_j) \subseteq I \cap E_c(q_j) \subseteq F$,

(c) *auxiliaries: for all* $q_j \in V \Rightarrow A_c(q_j) \subseteq I \cap E_c(q_j) \subseteq I$,

(d) *parameters: for all* $q_j \in P \Rightarrow A_c(q_j) = \phi \cap E_c(q_j) \subseteq I$,

(e) *inputs: for all* $q_j \in U \Rightarrow A_c(q_j) = \phi \cap E_c(q_j) \subseteq I$,

(f) *information delays: for all* $q_j \in D \Rightarrow A_c(q_j) \subseteq I \cap E_c(q_j) \subseteq I$,

(g) *material delays: for all* $q_j \in E \Rightarrow A_c(q_j) \subseteq F \cap E_c(q_j) \subseteq FI$,

(h) *multipliers: for all* $q_j \in M \Rightarrow A_c(q_j) \subseteq F \cap E_c(q_j) \subseteq I$.

Figure 3 summarizes the properties of each element in FD. The types of sets an element can belong to are shown on the left side. The intersection of row X with column A_c means that for all $q_j \in X$, $A_c(q_j) \subseteq F$. The similar expressing may apply to the other corresponding elements.

	A_c	E_c
X	F	I
R	I	F
V	I	I
P	ϕ	I
U	ϕ	I
D	I	I
E	F	FI
M	I	I

Figure 3. Properties of each element in FD

A directed graph $G=(L,E,n,\sigma)$ is called an n-level recurrent hierarchy if:

(1) L is partitioned into n subsets, that is

$L=L_1\cup L_2\cup\cdots\cup L_n$, where $L_i\cap L_j=\varnothing$, if $i\neq j$ besides $L_1=L_n$.

(2) E is partitioned into n subsets, that is

$E=E_1\cup E_2\cup\cdots\cup E_{n-1}$, where $E_i\subset L_i\times L_{i+1}$, if $i=1,\ldots,n-1$.

$E_i\cap E_j=\varnothing$, if $i\neq j$.

(3) n: the level numbers of hierarchies.

(4) $\sigma=\sigma_1\cup\sigma_2\cup\cdots\cup\sigma_n$, where σ_i is the sequence of all vertices of L_i for $i=1,\ldots,n$.

We define the span of an edge as the magnitude of the difference found by subtracting the number of the level from which the edge originates from the number of the level at which the edge terminates. If each edge in a recurrent hierarchy has span of 1, the recurrent hierarchy is called proper. In this study, edges are directed with ascending orders of levels. This definition is slightly different from Warfield [17]-[19]. And, the definition of the n-level recurrent hierarchy is also different from Sugiyama et al. [15] in (1) (2) of the previous definition. We use hierarchy as recurrent hierarchy in this paper.

An example of 5-level recurrent hierarchy shown in Figure 4, can be described as follows:

$G=(L,E,n,\sigma)$, where

(1) $L=L_1\cup L_2\cup L_3\cup L_4\cup L_5$, $L_1=\{a,b\}$, $L_2=\{c,d,e,f\}$, $L_3=\{k,l,m\}$, $L_4=\{n,o,p\}$, $L_5=\{a,b\}$

(2) $E=E_1\cup E_2\cup E_3\cup E_4$, $E_1=\{\overline{ac},\overline{ad},\overline{af},\overline{bd},\overline{be}\}$, $E_2=\{\overline{cl},\overline{dk},\overline{ek},\overline{em},\overline{fm}\}$

$E_3=\{\overline{kn},\overline{ln},\overline{lo},\overline{lp},\overline{mp}\}$, $E_4=\{\overline{na},\overline{oa},\overline{pb}\}$,

(3) $n=5$,

(4) $\sigma=\sigma_1\cup\sigma_2\cup\sigma_3\cup\sigma_4\cup\sigma_5$, $\sigma_1=a.b$, $\sigma_2=c.d.e.f$, $\sigma_3=k.l.m$, $\sigma_4=n.o.p$, $\sigma_5=a.b$.

Formulas to calculate the number of crossing of n-level hierarchies have been given by Warfield [19]. The number of crossing of n-level hierarchies is defined as $K(M)$:

$$K(M) = K(M^{(1)}) + K(M^{(2)}) + \cdots + K(M^{(n-1)})$$

where $K(M^{(i)}) = \sum_{j=1}^{|L_i|-1} \sum_{k=j+1}^{|L_i|} \left(\sum_{a=1}^{|L_i+1|-1} \sum_{b=a+1}^{|L_i+1|} m_{jb}^{(i)} \cdot m_{ka}^{(i)} \right), \quad i = 1,\ldots,n-1$.

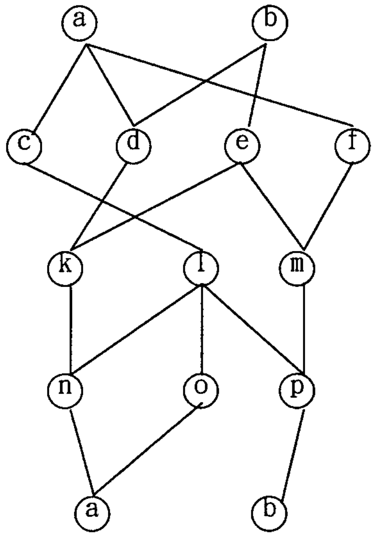


Figure 4. Example of a five-level recurrent hierarchy

The number of crossing in Figure 4 is described as

$$K(M) = K(M^{(1)}) + K(M^{(2)}) + K(M^{(3)}) + K(M^{(4)}) = 2 + 2 + 0 + 0 = 4 .$$

In a n-level recurrent hierarchy, the upper priority P_{ik}^U and the lower priority P_{ik}^L are defined by the formulas:

$$P_{ik}^U = \sum_{j=1}^{|L_i-1|} m_{jk}^{(i-1)}, \quad i = 2,\ldots,n, \; k = 1,\ldots,|L_i|,$$

$$P_{ik}^L = \sum_{l=1}^{|L_i+1|} m_{kl}^{(i-1)}, \quad i = 1,\ldots,(n-1), \; k = 1,\ldots,|L_i| .$$

Finally, upper and lower barycenters B_{ik}^U, B_{ik}^L of upper and lower vertices connected to the k th vertex L_k^i in the i th level are defined by

$$B_{ik}^{U}=\frac{\sum_{j=1}^{|L_i-1|} x_j^{i-1} m_{jk}^{(i-1)}}{P_{ik}^{U}}, \quad i=2,\dots,n, \ k=1,\dots,|L_i|,$$

$$B_{ik}^{L}=\frac{\sum_{l=1}^{|L_i+1|} x_l^{i+1} m_{kl}^{(i)}}{P_{ik}^{L}}, \quad i=1,\dots,(n-1), \ k=1,\dots,|L_i|.$$

where x_j^{i-1} is the horizontal position of j th vertex of $i-1$ th level. Figure 5 shows the priority numbers and barycenters of the horizontal position in Figure 4.

	upper priority	lower priority	upper barycenter	lower barycenter
a	-	3	-	2.3
b	-	2	-	2.5
c	1	1	1	2
d	2	1	1.5	1
e	1	2	2	2
f	1	1	1	3
k	2	1	2.5	1
l	1	3	1	2
m	2	1	3.5	3
n	2	1	1.5	1
o	1	1	2	1
p	2	1	2.5	2
a	2	-	1.5	-
b	1	-	3	-

Figure 5. The priority numbers and the barycenters of the horizontal positions in Figure 4.

3. CONVERTING FD TO HIERARCHY

In order to convert FD to hierarchy, we must solve the cycles in system dynamics. According to the semantic restrictions stated in an FD, a cycle contains at least one level variable. Therefore, the cycles can be eliminated by drawing the level variables twice at the top and bottom levels. A flowchart showing the relationship between the various steps and the iterative nature of the algorithm is given in Figure 6. Dummy vertices are added to break edges that span over more than two levels, so that the graph at the end of the first pass is a proper hierarchy.

The flowchart shows that the iteration process is required to facilitate the implementation process. A straight forward explanation that underlies the logic of the flowchart will be described as follows:

Step 1: Classify X, E and R.

Classifying those quantities whose affecter couplings is flow couplings, add to L_n. And, classifying those quantities whose affecter couplings is information couplings, add to L_{n-1}.

Step 2: Determine the levels of all vertices from L_{n-1} to L_2.

Each vertex can be determined only after all higher level vertices have been determined that is connected to the vertex. Let $A_q(L_{n-1})$ represent the set of quantities which have couplings directed to the quantities of L_{n-1}. L'_{n-1} represents the i th vertex of $n-1$ th level.

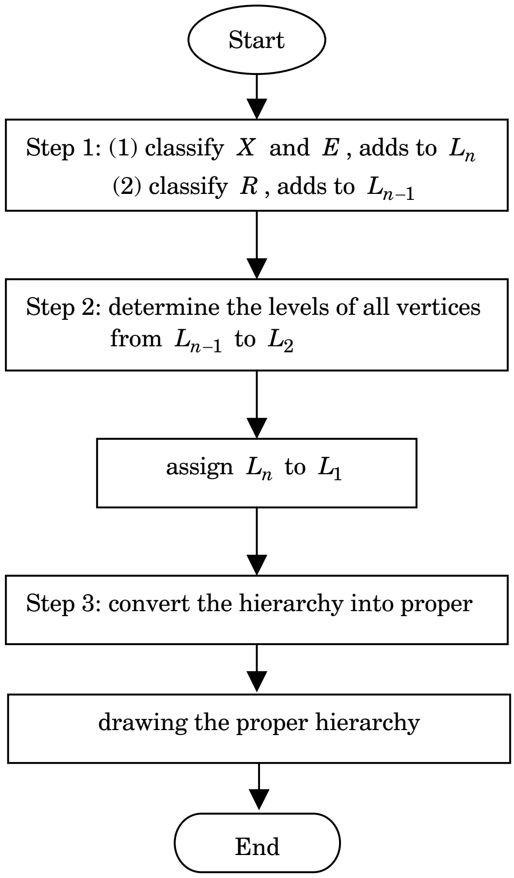


Figure 6: Flowchart for converting FD to hierarchy

For each $q_i \in L_{n-1}$, consider $A_q(L'_{n-1}) \notin X \cup E$, if $A_q(L'_{n-1})$ have not been determined, add $A_q(L'_{n-1})$ to L_{n-1}, otherwise, if $A_q(L'_{n-1})$ and L'_{n-1} in the same level, add $A_q(L'_{n-1})$ to L_{n-2} and subtract the same from L_{n-1}. When all the quantities of L_{n-1} have been determined, proceed the next loop, substitute n by $n-1$. If $n \neq 2$, the whole determined process has not been accomplished, then proceed to step 2, otherwise, proceed to the next step. Assigning the vertices of L_n to L_1 can be accomplished.

Step 3: Convert the hierarchy into proper.

If the hierarchy is improper, it can be replaced with a proper hierarchy by introducing additional dummy vertices are required to make all spans equal to 1. Let $E_q(L_j)$ represents the set of vertices which have edges directed away from L_j. And s_u represents the dummy vertices. Now all vertices have been finally determined, and the proper hierarchy can be delineated. The detailed algorithm is presented in Appendix.

4. REDUCTION OF THE NUMBER OF CROSSINGS

Two common methods for reducing the number of crossings are the barycenter function and the median function. Let v be a vertex and P the list of positions of its incident vertices on the appropriate adjacent level. Note that the position of an adjacent node is only its ordinal number in the current ordering. The barycenter method defines the weight of v as the average of elements in P. The median method defines the weight of v as the median of elements in P. When the number of elements in P is even, there are two medians. This gives rise to two median methods: always using the left median, and always using the right median. The median method consistently performs better than the barycenter method and has a slight theoretical advantage since Eades and Wormald [6] have shown that the median layout of a two level graphs has no more than three times the minimum number of crossings. No such bound is known for the barycenter method [2, 15, 19].

The study of reduction of the number of crossings is a refinement of the median method with two major innovations. First, when there are two median values, we use an interpolated valued biased toward the side where vertices are more closely packed. The second improvement uses an additional heuristic to reduce obvious crossing after the vertices have been sorted, transforming a given ordering the one that is locally optimal with respect to interchange of adjacent vertices.

It is an iterative method that computes at each iteration a tentative placement. The first iteration scans the graph from level 2 to max_level, where at level i $(i = 1,\ldots,\text{max\_level})$ the vertices are sorted according to the up_median. The second iteration works from level max_level – 1 to level 1, and uses the down_median. Successive iterations alternate between top-down and bottom-up scanning of the levels. A flowchart of the algorithm is given in Figure 7. The detailed algorithm is presented in Appendix.

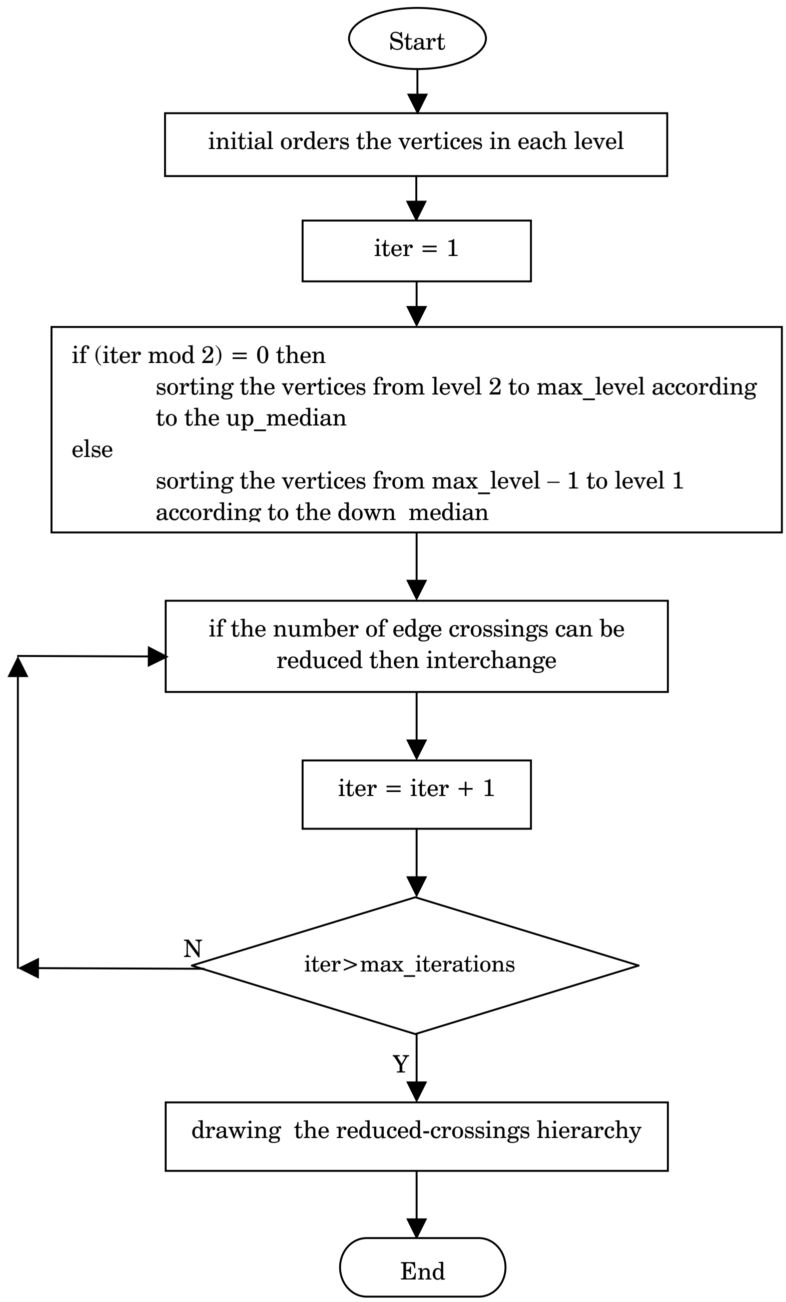


Figure 7: Flowchart for reducing the number of edge crossings

5. ADJUSTMENT OF THE HORIZONTAL POSITIONS OF VERTICES

Adjusting the horizontal positions of vertices according to priority numbers given to vertices is introduced to improved readability of hierarchies. Priority numbers given to the other vertices are the connectivity of the vertices calculated by the previous definition in section 2. This method satisfies the following conditions:

(a) The order of vertices of each level should be preserved.

(b) The principle to improve the position of a vertex is to minimize the difference between the present position of the vertex and the upper (or lower) barycenter of the vertex.

(c) The horizontal position of the vertex should be integer, otherwise replace it with approximate integer value. And, it can not be equal to the positions of the other vertices in the same level.

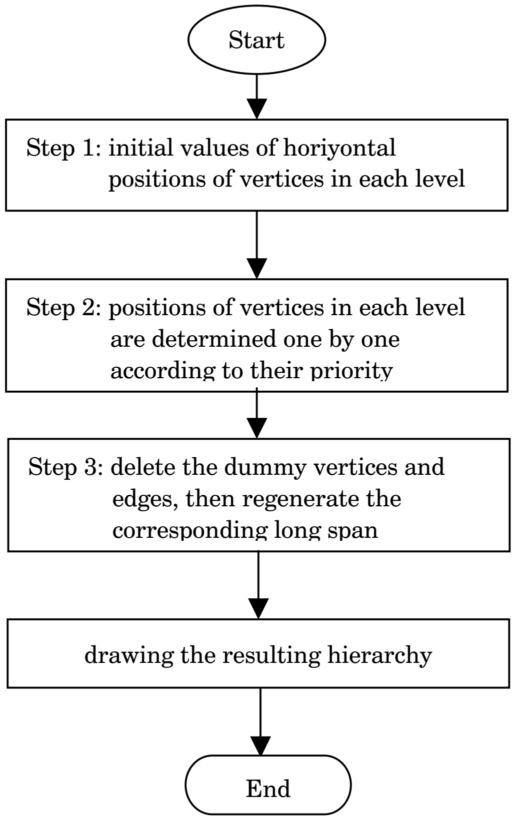


Figure 8. Flowchart for adjusting the horizontal positions of vertices

A flowchart of the algorithm is given in Figure 8. A straight forward explanation underlies the logic of flowchart is described as follows:

Step 1: Initials values of horizontal positions of vertices in each level.

We let $L_i^k = x_0 + k,\ k = 1,\ldots,|L_i|,\ i = 1,\ldots,n,$ where x_0 is a given integer.

Step 2: Positions of vertices in each level are determined one by one according to their priority numbers.

Positions of vertices in level 1 are improved according to the lower priority and the lower barycenter. The improvements of the positions of vertices in level $2,\ldots,n$ are made according to the upper priority and the upper barycenter. The highest priority number is given to dummy vertices to improve the readability.

Step 3: Delete the dummy vertices and edges, and then regenerate the corresponding long span. Now all vertices have been finally adjusted, and the resulting hierarchy can be available. The detailed algorithm is presented in Appendix.

6. EXAMPLE

The casual diagram and Forrester Diagram chosen for demonstration of the algorithm discussed in this study are shown in Figure 9 and Figure 10 which are based upon the residential community model excerpted from Goodman [8]. The names of quantities and associated designators are exhibited in Table 1.

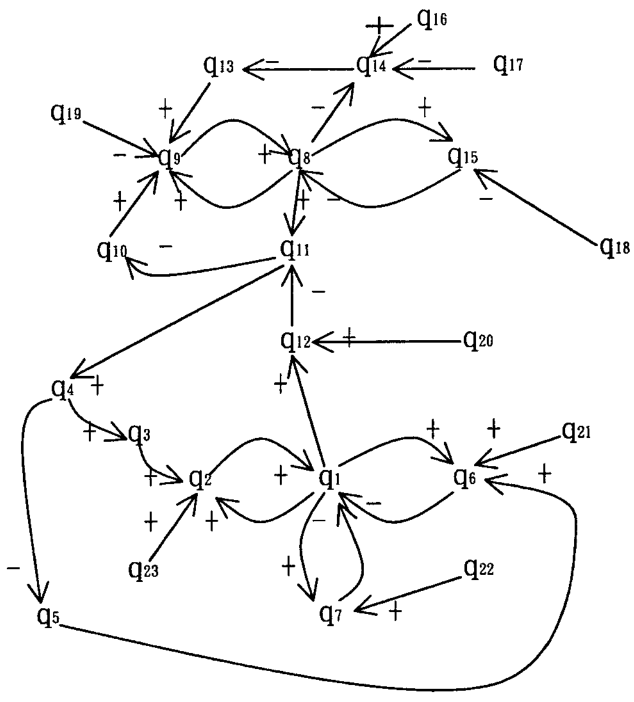


Figure 9: Causal diagram of a residential community model [8]

Table 1: Quantities show in residential community model

Quantities	Name
q_1	Population
q_2	In-migration rate
q_3	Attractiveness for migration multiplier perceived
q_4	Attractiveness for migration multiplier
q_5	Departure migration multiplier
q_6	Out-migration rate
q_7	Net death rate
q_8	Housing
q_9	Housing construction rate
q_{10}	Housing construction multiplier
q_{11}	Housing ratio
q_{12}	Housing desired
q_{13}	Land availability multiplier
q_{14}	Land fraction occupied
q_{15}	Housing demolition rate
q_{16}	Land
q_{17}	Land per unit
q_{18}	Average lifetime of housing
q_{19}	Normal housing construction
q_{20}	Units per person
q_{21}	Normal out-migration
q_{22}	Death rate factor
q_{23}	Normal in-migration

The process of the first pass is as follows:

Step 1: Classify X_1 and X_8 then add $\{X_1, X_8\}$ to L_8; classify R_2, R_6, R_7, R_9 and R_{15} add $\{R_2, R_6, R_7, R_9, R_{15}\}$ to L_7.

Step 2: Consider all quantities which have couplings directed toward R_2 we add $\{D_3, P_{23}\}$ to L_6. The similar operations may apply to $\{R_6, R_7, R_9, R_{15}\}$, add $\{M_5, P_{21}, P_{22}, M_{10}, M_{13}, P_{19}, P_{18}\}$ to L_6.

Because L_7 is empty, we classify the quantities of L_6, then add $\{M_4, V_{11}, V_{14}\}$ to L_5. L_6 is empty, proceed the next loop, consider V_{11}, the affecter quantity of M_4.

V_{11} and M_4 are in the same level, so add V_{11} to M_4, and subtract the same from L_5. Furthermore, add $\{P_{16}, P_{17}\}$ to L_4.

Again proceed to the loop of L_4 and L_3, we add $\{V_{12}\}$ to L_3, $\{P_{20}\}$ to L_2. And, assign L_8 to L_1.

Step 3: Add $\{S_1, S_2, S_3, S_4, S_5, S_6, S_7\}$ to L_2, $\{S_8, S_9, S_{10}, S_{11}, S_{12}, S_{13}\}$ to L_3, $\{S_{14}, S_{15}, S_{16}, S_{17}, S_{18}, S_{19}\}$ to L_4, $\{S_{20}, S_{21}, S_{22}, S_{23}, S_{24}, S_{25}\}$ to L_5, and $\{S_{26}, S_{27}, S_{28}, S_{29}, S_{30}\}$ to L_6. The first pass is accomplished. The number of edge crossings is shown as $K(M) = 24$. The proper hierarchy is presented in Figure 11.

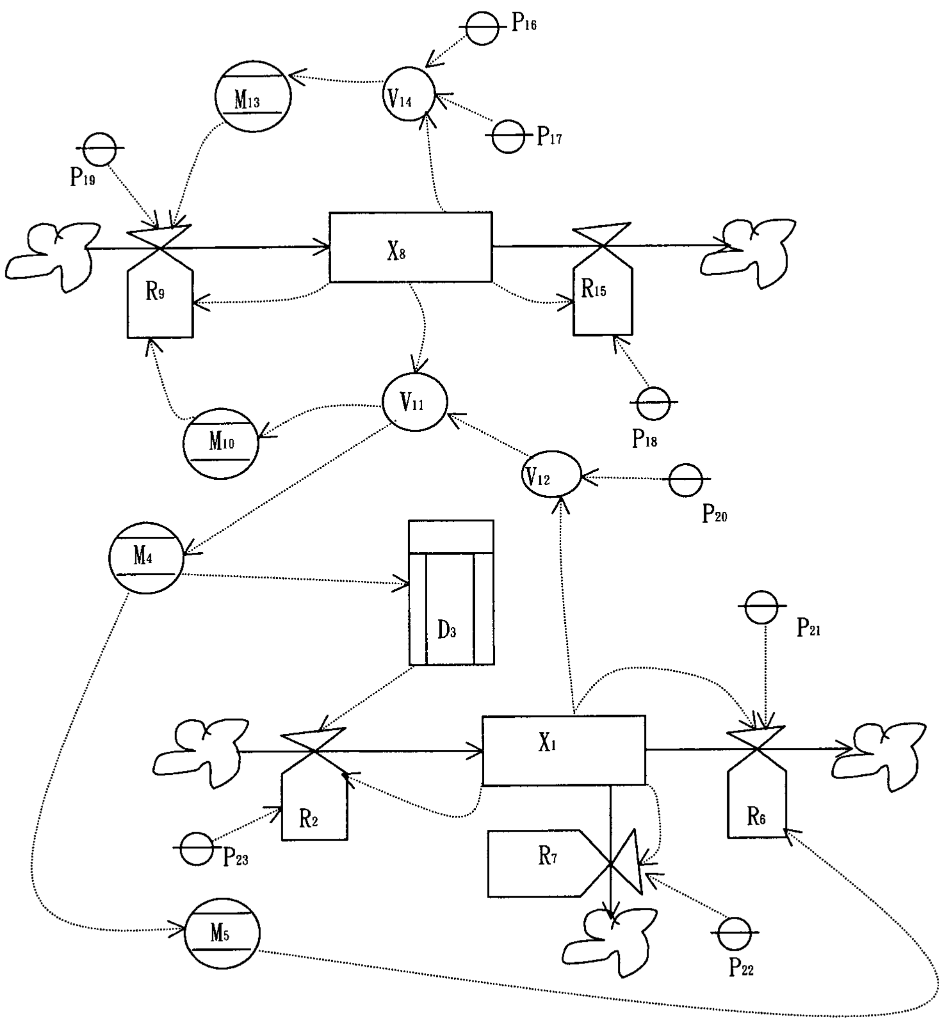


Figure 10: FD of a residential community model [8]

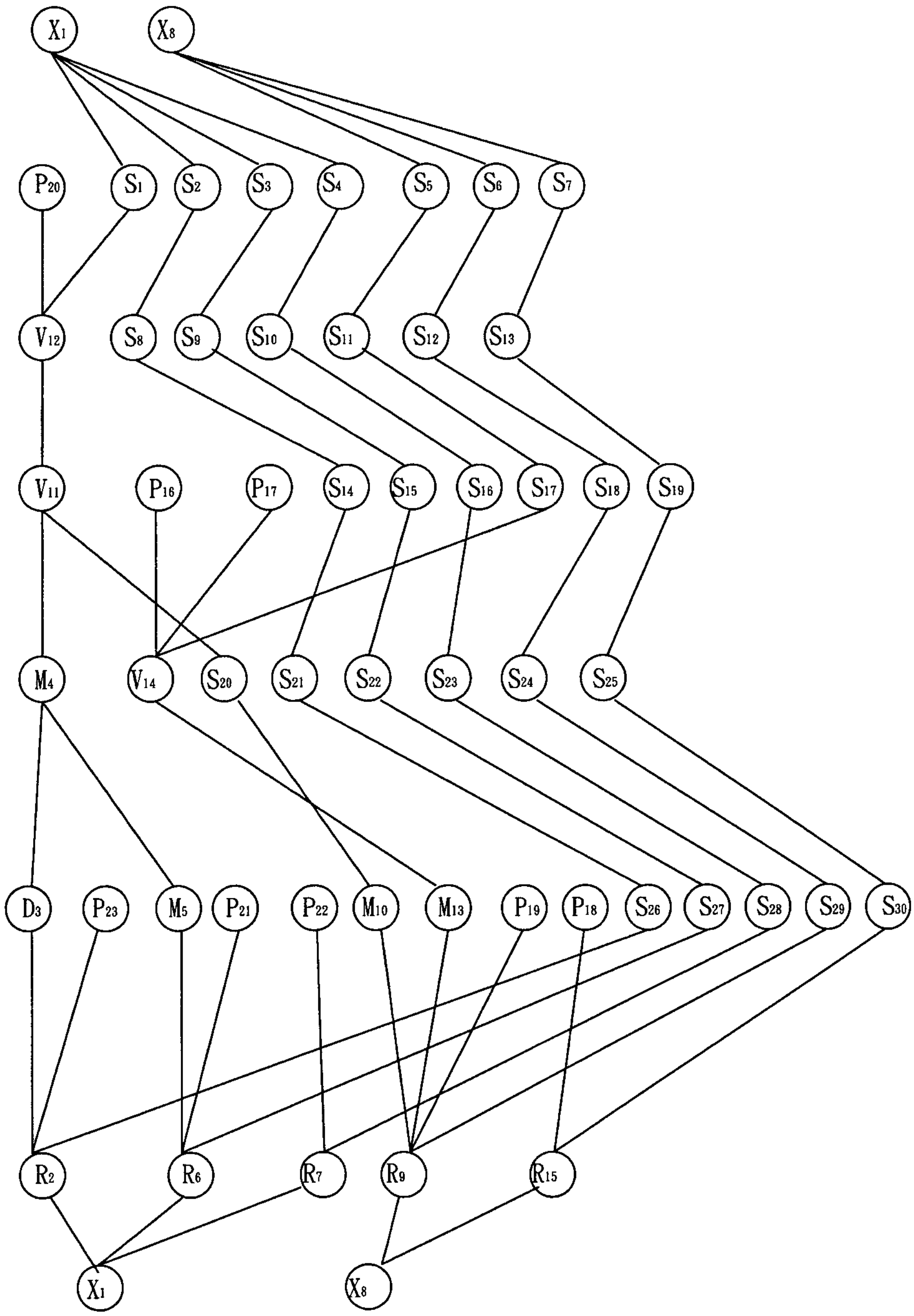


Figure 11: The proper hierarchy of the residential community model

The process and result of applying pass 2 is given as follows:

The first iteration sorts the vertices from level 2 to level 8 according to the up_median. The orders of all vertices are no change in level 2 – 4. In level 5, we interchange the order of S_{20}, V_{14} . There are no changes in level 6 – 8. The number of edge crossings is reduced from 24 to 20.

Then, the second iteration sorts the vertices from level 7 to level 1 according to down_median. The order of all the vertices in level 7 is no change. By exchanging the sequence of the vertices in level 6, we obtain $\sigma_6 = \{P_{23}, D_3, S_{26}, M_5, P_{21}, S_{27}, P_{22}, S_{28}, M_{10}, M_{13}, P_{19}, S_{29}, P_{18}, S_{30}\}$. For level 2 – 5 we have $\sigma_5 = \{M_4, S_{21}, S_{22}, S_{23}, S_{20}, V_{14}, S_{24}, S_{25}\}$, $\sigma_4 = \{S_{14}, V_{11}, S_{15}, S_{16}, P_{16}, P_{17}, S_{17}, S_{18}, S_{19}\}$, $\sigma_3 = \{S_8, V_{12}, S_9, S_{10}, S_{11}, S_{12}, S_{13}\}$, $\sigma_2 = \{S_2, P_{10}, S_1, S_3, S_4, S_5, S_6, S_7\}$. The number of edge crossings is shown as $K(M) = 4$.

Precede the next successive iterations; sort the vertices from level 2 to level 8 according to the up_median. The orders of all vertices are no further change in level 2 – 4. For level 5 – 6 we have exchange $\sigma_5 = \{S_{21}, M_4, S_{20}, S_{22}, S_{23}, V_{14}, S_{24}, S_{25}\}$, $\sigma_6 = \{P_{23}, S_{26}, D_3, M_5, P_{21}, M_{10}, S_{27}, P_{22}, S_{28}, M_{13}, P_{19}, S_{29}, P_{18}, S_{30}\}$. There are no changes in level 7 – 8. The number of edge crossings is reduced to 3.

Then, sort the vertices from level 7 to level 1 according to down_median. The order of the all vertices in level 7 is no change. By exchanging the sequence of the vertices from level 6 to level 2, we obtain $\sigma_6 = \{P_{23}, S_{26}, D_3, M_5, P_{21}, S_{27}, P_{22}, S_{28}, M_{10}, M_{13}, P_{19}, S_{29}, P_{18}, S_{30}\}$, $\sigma_5 = \{S_{21}, M_4, S_{22}, S_{23}, S_{20}, V_{14}, S_{24}, S_{25}\}$, $\sigma_4 = \{S_{14}, S_{15}, V_{11}, S_{16}, P_{16}, P_{17}, S_{17}, S_{18}, S_{19}\}$, $\sigma_3 = \{S_8, S_9, V_{12}, S_{10}, S_{11}, S_{12}, S_{13}\}$, $\sigma_2 = \{S_2, S_3, P_{20}, S_1, S_4, S_5, S_6, S_7\}$. The number of edge crossings is shown as $K(M) = 2$.

The next iteration is executed, sorting the vertices from level 2 to level 8 according to the up_median. By exchanging the sequence of the vertices from level 2 to level 8, we obtain $\sigma_2 = \{S_4, S_2, S_3, P_{20}, S_1, S_5, S_6, S_7\}$, $\sigma_3 = \{S_{10}, S_8, S_9, V_{12}, S_{11}, S_{12}, S_{13}\}$, $\sigma_4 = \{S_{16}, S_{14}, S_{15}, V_{11}, P_{16}, P_{17}, S_{17}, S_{18}, S_{19}\}$, $\sigma_5 = \{S_{23}, S_{21}, S_{22}, M_4, S_{20}, V_{14}, S_{24}, S_{25}\}$, $\sigma_6 = \{P_{22}, S_{28}, P_{23}, S_{26}, S_{27}, D_3, M_5, P_{21}, M_{10}, M_{13}, P_{19}, S_{29}, P_{18}, S_{30}\}$. The number of edge crossings is reduced to 1.

Then, sort the vertices from level 7 to level 1 according to down_median. The order of all the vertices in level 6 and level 5, we obtain $\sigma_6 = \{P_{22}, S_{28}, S_{26}, P_{23}, D_3, S_{27}, M_5, P_{21}, M_{10}, M_{13}, P_{19}, S_{29}, P_{18}, S_{30}\}$, $\sigma_5 = \{S_{23}, S_{21}, M_4, S_{22}, S_{20}, V_{14}, S_{24}, S_{25}\}$. The orders of all vertices are no change in level 1 – 4. The number of edge crossings is shown as $K(M) = 2$. It does not decrease, so the second pass is completed. The result is displayed in Figure 12, the number of crossings is reduced from 24 to 1.

The process and result of applying pass 3 is given as follows:

First, we improve the position of vertices in level 1. The position of X_1 is improved, since it has the highest priority number (=4). The horizontal position of X_1 is 2.75 $\left(\frac{1+2+3+5}{4} = 2.75\right)$, so we get 3 as the best horizontal position of X_1 . And, the position of X_8 can be adjusted to its horizontal position 7 $\left(\frac{6+7+8}{3} = 7\right)$.

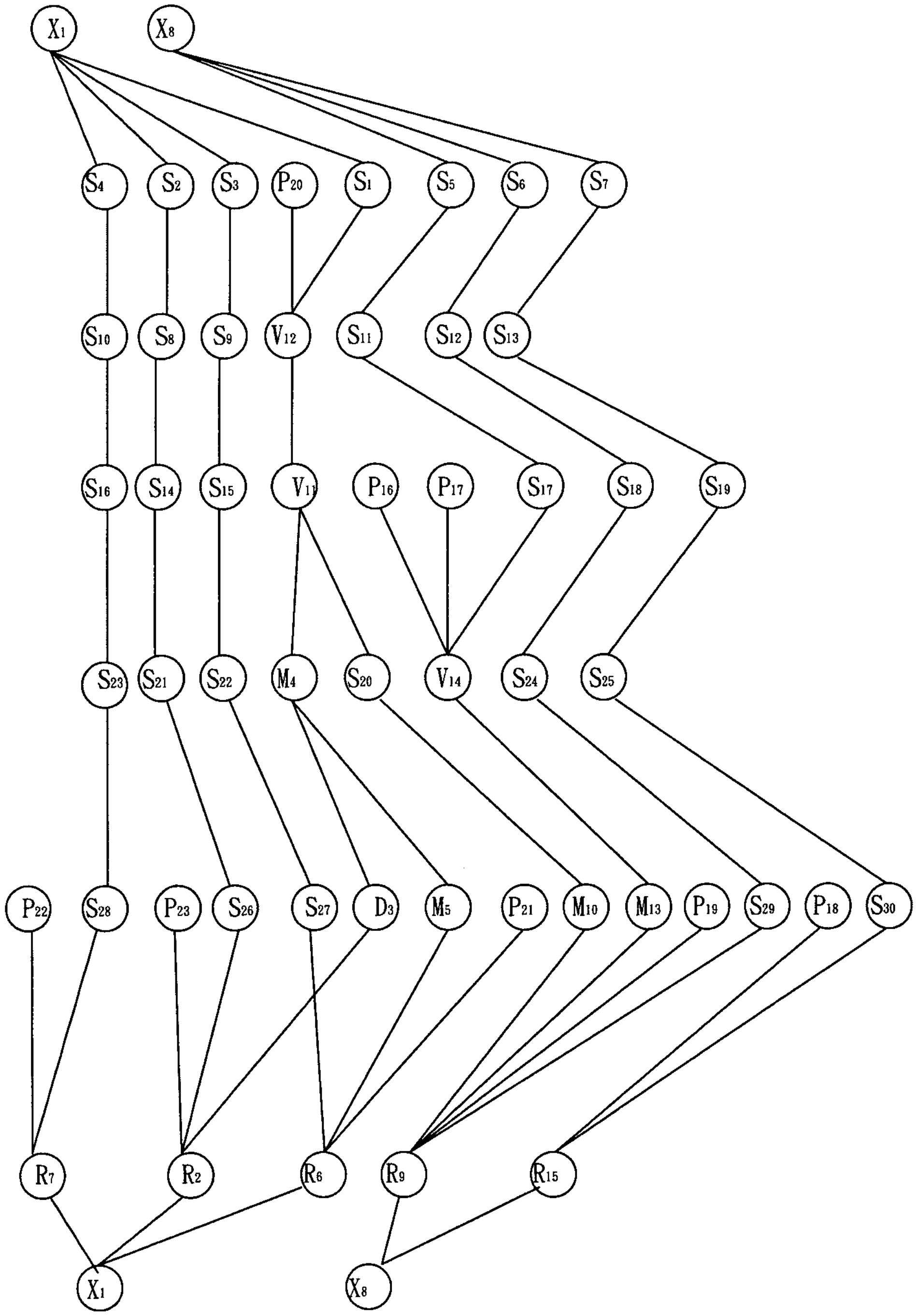


Figure 12: Results of reducing the number of crossings

By improvement in level 3, $\{S_{10}, S_8, S_9, V_{12}, S_{11}, S_{12}, S_{13}\}$ are displaced to {1,2,3, 5,6,7,8} of the horizontal positions. For level 4, we have the horizontal positions 1,2,3,8, 9,10 for the dummy vertices $S_{16}, S_{14}, S_{15}, S_{17}, S_{18}, S_{19}$, and add V_{11}, P_{16}, P_{17} to position 5, 6, and 7.

The dummy vertices $\{S_{23}, S_{21}, S_{22}, S_{20}, S_{24}, S_{25}\}$ are displaced to {1,2,3,6,9,10} of the horizontal positions in level 5. V_{14} has the largest priority number (=3) and its horizontal position is 7 $\left(\frac{6+7+8}{3}=7\right)$, so we get 7 is the best horizontal position of V_{14}. Then, add M_4 to position 5. For level 6, we have the horizontal positions 2,4,5, 12,14 for the dummy vertices $S_{28}, S_{26}, S_{27}, S_{29}, S_{30}$, and add $P_{22}, P_{23}, D_3, M_5, P_{21}, M_{10}, M_{13}$, P_{19}, P_{18} to position 1,3,6,7,8,9,10,11,13.

Proceed to level 7, R_9 has the largest priority number (=4) and its horizontal position is 10.5 $\left(\frac{9+10+11+12}{4}=10.5\right)$, so we get 11 is the best horizontal position R_9. Next, R_2, R_6 has the second largest priority number, adjust R_2 to horizontal position 4 $\left(\frac{3+4+5}{3}=4\right)$ and R_6 to horizontal position 7 $\left(\frac{5+7+8}{3}=6.7\right)$. The processes of R_7, R_{15} are so as R_2, add R_7 to position 2 and R_{15} to position 14.

Finally, for level 8, the position of X_1 is improved, since it has the largest priority number (=3). The horizontal position of X_1 is 4.3 $\left(\frac{2+4+7}{3}=4.3\right)$, so we get 4 is the best horizontal position of X_1. And, the horizontal position of X_8 is 12.5 $\left(\frac{11+14}{2}=12.5\right)$, so we get 13 is the best horizontal position of X_8. Then, dummy vertices and edges are deleted, and the corresponding long edges are regenerated. The third pass is accomplished. Further, the final map of hierarchy is available in Figure 13.

7. CONCLUSION

We have presented a method for improving readability of FD, which consists of three passes. The first pass converts FD to recurrent hierarchy. The second pass sorts the vertices on each level to avoid edge crossings. The third pass adjusts the horizontal positions of vertices. The study proposed here can make up a deficiency of the past work in manipulating FD in system dynamics. Converting FD to the hierarchies is useful to realize the relations of all vertices in complex systems, but also operate elements and couplings over an FD more clearly and conveniently. Moreover, permits easy implementation by the modeler.

By developing the theoretical algorithms, we can recognize the nature of the problem to generate the readable representation of FD in system dynamics. On the other hand, by developing the heuristic algorithms, we enlarge the size of the problems we can deal with.

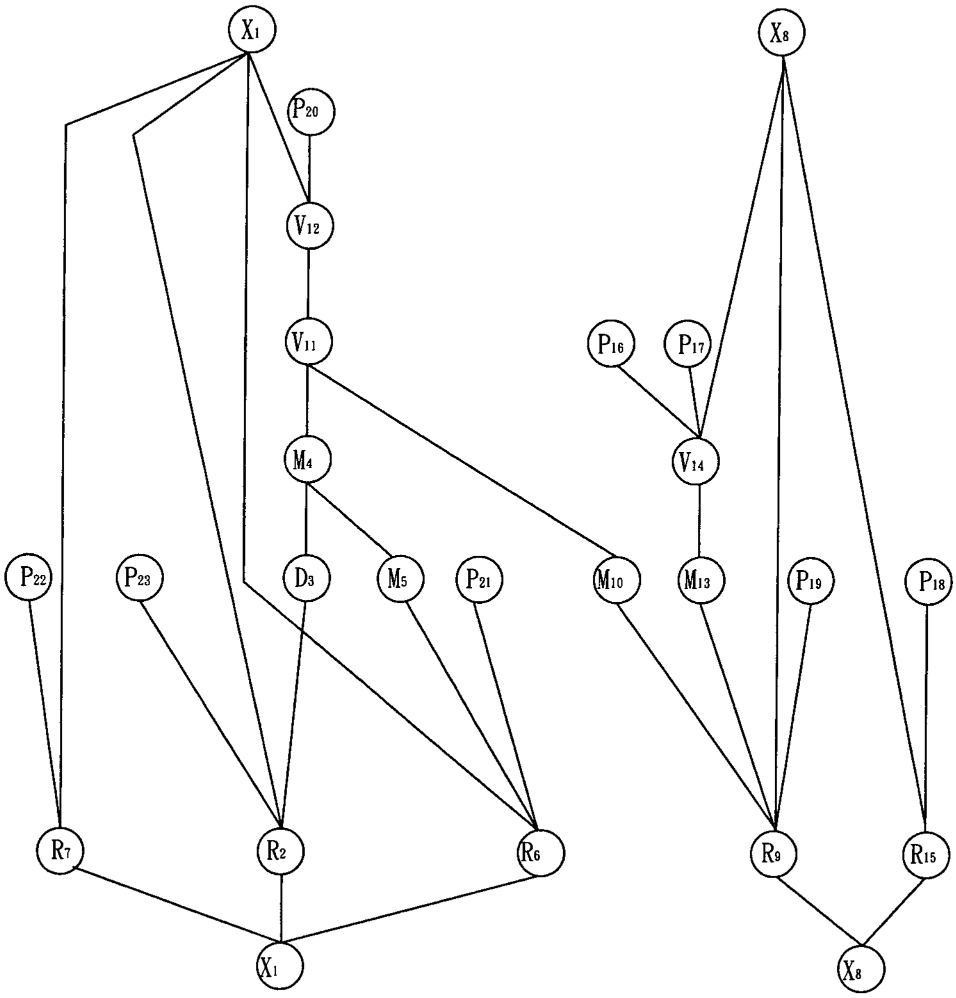


Figure 13. Results of hierarchical graph visualization

Finally, the following studies are envisaged for future research:

(1) understand how to modify the graph or its layout to enhance readability,
(2) developments of methods for readable drawing of undirected graphs, and
(3) associate with graph-processing tool to develop the tool of computer simulation in system dynamics.

REFERENCES

[1] Andersen, D.F., "Analyzing who gains and who loses: the case of school finance reform in New York State", *System Dynamics Review*, 6 (1) (1990) 21-43.

[2] Carpano, M., "Automatic display of hierarchized graphs for computer aided decision analysis", *IEEE Trans. S.M.C.*, 10 (11) (1980) 705-715.

[3] Coyle, R.G., and Alexander, M.W.D., "Quantitative modeling of a nation's drugs trade", *System Dynamics Review*, 12 (2) (1997) 205-222.

[4] Dolado, J.J., and Torrealdea, F.J., "Formal manipulation of Forrester diagram by graph grammars", *IEEE Trans. S.M.C.,* 18 (6) (1988) 981-996.

[5] Draper, F., "A proposed sequence for developing system's thinking in a grades 4-12 curriculum", *System Dynamics Review,* 9 (2) (1993) 207-214.

[6] Eades, P., and Wormald, N., "The median heuristic for drawing 2-layers networks", Tech. Rep. 69, Dept. of Computer Science, University of Queensland, 1986.

[7] Forrester, J.W., *Industrial Dynamics*, MIT Press, Cambridge, MA, 1961.

[8] Goodman, M.R., *Study Notes in System Dynamics*, Wright Allen Press, Cambridge, MA, 1974.

[9] Kim, D.-H., and Kim, D.-H., "A system dynamics model for a mixed strategy game between police and driver", *System Dynamics Review*, 13 (1) (1997) 33-52.

[10] Lane, D.C., "Social theory and system dynamics practice", *European Journal of Operations Research*, (1999) 501-527.

[11] Lin, C., et al., "A generic methodology that aids the application of system dynamics to manufacturing system modeling", *International Conference on Simulation*, 457 (1998) 344-349.

[12] Nuthmannf, C., "Using human judgment in system dynamics models of social system", *System Dynamics Review*, 10 (1) (1994) 1-27.

[13] Peterson, D.W., and Eberlein, R.L., "Reality check: A bridge between systems thinking and system dynamics", *System Dynamics Review*, 10 (2-3) (1994) 159-174.

[14] Richardson, G.P., "Problems for the future of system dynamics", *System Dynamics Review*, 12 (2) (1996) 141-157.

[15] Sugiyama, K., Tagawa, S., and Toda, M., "Method for visual understanding of hierarchical system structures", *IEEE Trans. S.M.C.,* 11 (2) (1981) 109-125.

[16] Vazquez, M., Liz, M., and Aracil, J., "Knowledge and reality: some conceptual issues in system dynamics modeling", *System Dynamics Review,* 12 (1) (1996) 21-37.

[17] Warfield, J.N., "Toward interpretation of complex structural models", *IEEE Trans. S.M.C.,* 4 (1974) 405-407.

[18] Warfield, J.N., *Societal System*, Wiley, New York, 1976.

[19] Warfield, J.N., "Crossing theory and hierarchy mapping", *IEEE Trans. S.M.C.*, 7 (7) (1977) 505-523.

[20] Wei, Z., and Xiongjian, L., "A system dynamics model for pricing telecommunication services", *International Conference on Communication Technology*, (1998) 22-24.

APPENDIX

Algorithm 1: Converting FD to hierarchy.

```
procedure level_assign()
begin
    r = 1 ;
    s = 1 ;
    for i = 1 to |Q|
        if q_i ∈ X ∪ E then
            q_i → L_j ;
            r = r + 1 ;
        elseif q_i ∈ R then
            q_i → L_{j-1} ;
            s = s + 1 ;
        endif
    next i
    t = |Q| − r − s ;
    p = 2 ;
    k = 1 ;
    do while k ≤ t
        u = 1 ;
        do while u ≤ |L_{j-1}|
            if A_q(L^u_{j-1}) = φ then
                u = u + 1 ;
                k = k + 1 ;
            elseif (visited A_q(L^u_{j-1}) = .true. ) then
                { if [L^u_{j-1} ∩ A_q(L^u_{j-1})] ∈ L_{j-1} then
                        L_{j-1} − A_q(L^u_{j-1}) → L_{j-1} ;
                        A_q(L^u_{j-1}) → L_{j-2} ;
                        k = k − 1;
                  else
                        u = u + 1 ;
                        k = k + 1 ;
                  endif}
            elseif A_q(L^u_{j-1}) ∈ X ∪ E then
                u = u + 1 ;
                k = k + 1 ;
            else
```

$A_q(L_{j-1}^u) \rightarrow L_{j-2}$;
endif
enddo
$j = j - 1$;
$p = p + 1$;
enddo
$n = 2$;
do while $n \leq p + 1$
$L_j \rightarrow L_n$;
$n = n + 1$;
$j = j + 1$;
enddo
$L_n \rightarrow L_1$;
proper()
end

Algorithm 2: Converting the hierarchy into proper.

procedure proper()
begin
for $i = 1$ to $n - 1$
for $j = 1$ to $|L_i|$
$Q_j = E_q(L_i^j) - L_i^j - [E_q(L_i^j) \cap L_{i+1}]$;
do while $|Q_j| \neq 0$
{
$|L_{i+1}| + 1 \rightarrow u$;
$L_{i+1} \cup s_u \rightarrow L_{i+1}$;
$E_q(L_i^j) \cup s_u \rightarrow E_q(L_i^j)$;
$Q_j \rightarrow E_q(s_u)$;
$L_{i+1} + 1 \rightarrow |L_{i+1}|$;
$|Q_j| - 1 \rightarrow |Q_j|$;
} enddo
next j
do while $[E_q(s_p) = E_q(s_q)]$
{
$A_q(s_q) \rightarrow A_q(s_p)$;
delete s_q ;
} enddo
next i

```
end
```

Algorithm 3: Reduction of the number of crossings.

```
procedure ordering()
begin
    order = init_order();
    best = order;
    for i = 1 to max_iterations do
        wmedian(order, i );
        interchange(order);
        if crossing(order) < crossing(best) then
            best = order;
        endif
    end
    return best;
end
```

Algorithm 4: The median procedure.

```
procedure wmedian(order, iter)
begin
    if (iter mod 2)=0 then
        for r = 2 to max_level do
            for v in order[r] do
                median[v] =median_value( v,r − 1 );
            end
        sort(order[r] , median);
        end
    else

        for r = (max_level – 1) to 1 do
            for v in order[r] do
                median[v] =median_value( v,r + 1 );
            end
        sort(order[r] , median);
        end
    endif
end

procedure median_value(v, adj_level)
begin
    p = adj_position(v, adj_level);
```

```
    m = | p | / 2 ;
    if | p | = 0 then
        return 0;
    elseif (| p | mod 2) = 1 then
        return p[m];
    elseif | p | = 2 then
        return (p[0] + p[1]) / 2;
    else
        left = p[m − 1] − p[0];
        right = p[| p | − 1] − p[m];
        return (p[m − 1] ∗ right + p[m] ∗ left)/(left + right);
    endif
end
```

Algorithm 5: The interchange procedure.

```
procedure interchange(level)
begin
    inproved=.true.;
    while improved do
        improved=.false.;
        for r = 1 to max_level do
            for i = 0 to | level[r] | −2 do
                v = level[r][i];
                w = level[r][i + 1];
                if crossing(v,w) > crossing(w,v) then
                    improved=.true.;
                    exchange(level[r][i], level[r][i + 1]);
                endif
            end
        end
    end
end
```

Algorithm 6: Adjustment of the horizontal positions of vertices.

```
procedure position()
begin
    for i = 1 to n
        for k = 1 to | L_i |
            L_i^k = x_0 + k;
```

```
            next  k
        next  i
        compute  P_{1k}^L ;
        do while  L_1 ≠ φ
            {
                P_{1k}^L  is highest in  L_i ;
                adjust  L_i^k ;
                L_i = L_i − L_i^k ;
            }
        enddo
        for  i = 3  to n
            for  k = 1  to  | L_i |
            if ( L_i^k  is dummy vetex) then
                adjust  L_i^k ;
            else
                P_{ik}^U = sum_{j=1}^{|L_i −1|} m_{jk}^{(i−1)} ;
            endif
        next  k
            do while  L_i ≠ φ
                {
                    P_{ik}^U  is highest in  L_i ;
                    adjust  L_i^k ;
                    L_i = L_i − L_i^k ;
                }
            enddo
        next  i
end
```