Rough Sets and Flow Graphs

Zdzisław Pawlak

Institute for Theoretical and Applied Informatics,
Polish Academy of Sciences,
ul. Bałtycka 5, 44-100 Gliwice, Poland
Warsaw School of Information Technology,
ul. Newelska 6, 01-447 Warsaw, Poland
zpw@ii.pw.edu.pl

Abstract. This paper concerns the relationship between rough sets and flow graphs. It is shown that flow graph can be used both as formal language for computing approximations of sets in the sense of rough set theory, and as description tool for data structure. This description is employed next for finding patterns in data. To this end decision algorithm induced by the flow graph is defined and studied.

Keywords: rough sets; flow graphs; decision algorithms.

1 Introduction

We study in this paper the relationship between rough sets and flow graphs. It is revealed that flow graph can be used as a formal language for rough set theory and can be also used for decision algorithm simplification. Flow graphs introduced in this paper are different from those proposed by Ford and Fulkerson for optimal flow analysis [1].

Flow graphs can be used for approximate reasoning modeling based on the flow principle. In particular, it is shown in this paper that if we interpret nodes of flow graphs as subsets of a finite universe, such that for any branch (x,y) of the flow graph (x is an input of y) we have $x \cap y \neq \emptyset$, then the union of all inputs x of y is the upper approximation of y. Similarly, the union of all inputs x of y, such that $x \subseteq y$, is the lower approximation of y, provided that all inputs of y are mutually disjoint.

Besides, independency and dependency (statistical) of conditions and decisions of decision rules are defined and discussed.

This paper is a continuation of the author's ideas presented in [7,8] (see also [2,3]).

2 Rough Sets

In this section we recall briefly after [6] basic concept of rough set theory.

A starting point of rough set based data analysis is a data set, called an information system.

D. Ślęzak et al. (Eds.): RSFDGrC 2005, LNAI 3641, pp. 1–11, 2005.

Formally, by an *information system* we will understand a pair $S = (U, A)$, where U and A, are finite, nonempty sets called the *universe*, and the set of *attributes,* respectively. With every attribute $a \in A$ we associate a set V_a of its *values*, called the *domain* of a. Any subset B of A determines a binary relation I(B) on U, called an *indiscernibility relation*, and defined as follows: $(x, y) \in I(B)$ if and only if $a(x) = a(y)$ for every $a \in A$, where a(x) denotes the value of attribute a for element x.

Obviously I(B) is an equivalence relation. The family of all equivalence classes of I(B), i.e., a partition determined by B, will be denoted by U/I(B), or simply by U/B. An equivalence class of I(B), i.e., block of the partition U/B, containing x will be denoted by B(x).

If (x, y) belongs to I(B), we will say that x and y are B-*indiscernible* (*indiscernible with respect to* B). Equivalence classes of the relation I(B) (or blocks of the partition U/B) are referred to as B-*elementary sets* or B-*granules*.

If we distinguish in an information system two disjoint classes of attributes, called *condition* and *decision attributes*, respectively, then the system will be called a *decision system*, denoted by $S = (U, C, D)$, where C and D are disjoint sets of condition and decision attributes, respectively.

Suppose we are given an information system $S = (U, A)$, $X \subseteq U$, and $B \subseteq A$. Our task is to describe the set X in terms of attribute values from B. To this end we define two operations assigning to every $X \subseteq U$ two sets $B_*(X)$ and $B^*(X)$ called the B-*lower* and the B-*upper approximation* of X, respectively, and defined as follows:

$$B_*(X) = \bigcup_{x \in U} \{B(x) : B(x) \subseteq X\} \tag{1}$$

$$B^*(X) = \bigcup_{x \in U} \{B(x) : B(x) \cap X \neq \emptyset\} \tag{2}$$

Hence, the B-lower approximation of a set X is the union of all B-granules that are included in X, whereas its B-upper approximation is the union of all B-granules that have a nonempty intersection with X. The set

$$BN_B(X) = B^*(X) - B_*(X) \tag{3}$$

will be referred to as to the B-*boundary region* of X.

If the boundary region of X is the empty set, i.e., $BN_B(X) = \emptyset$, then X is *crisp* (*exact*) with respect to B. In the opposite case, i.e., if $BN_B(X) \neq \emptyset$, then X is referred to as to *rough* (*inexact*) with respect to B.

3 Flow Graphs

In this section we recall after [7] basic definitions and properties of flow graphs.

Flow graphs can be considered as a special kind of databases, where instead of data about individual objects some statistical features of objects are presented in terms of information flow distribution. It turns out that such data representation

gives a new insight into data structures and leads to new methods of intelligent data analysis.

A flow graph is a *directed acyclic finite* graph $G = (N, \mathcal{B}, \varphi)$, where N is a set of *nodes*, $\mathcal{B} \subseteq N \times N$ is a set of *directed branches*, $\varphi : \mathcal{B} \to R^+$ is a *flow function*, and R^+ is the set of non-negative reals. We list basic concepts of flow graphs:

- If $(x, y) \in \mathcal{B}$ then x is an *input* of y and y is an *output* of x.
- If $x \in N$ then $I(x)$ and $O(x)$ denote the sets of all x's inputs and outputs.
- *Input* and *output* of a graph G are defined, respectively, as

$$I(G) = \{x \in N : I(x) = \emptyset\} \text{ and } O(G) = \{x \in N : O(x) = \emptyset\}$$

- Inputs and outputs of G are its *external nodes*; other nodes are *internal*.
- If $(x, y) \in \mathcal{B}$ then $\varphi(x, y)$ is a *throughflow* from x to y;
  We will assume in what follows that $\varphi(x, y) \neq 0$ for every $(x, y) \in \mathcal{B}$.

With every node x of a flow graph G we associate its *inflow*

$$\varphi_+(x) = \sum_{y \in I(x)} \varphi(y, x) \tag{4}$$

and *outflow*

$$\varphi_-(x) = \sum_{y \in O(x)} \varphi(x, y) \tag{5}$$

Similarly, we define an inflow and an outflow for the whole flow graph G:

$$\varphi_+(G) = \sum_{x \in I(G)} \varphi_-(x) \tag{6}$$

$$\varphi_-(G) = \sum_{x \in O(G)} \varphi_+(x) \tag{7}$$

We assume that for any internal node x, $\varphi_+(x) = \varphi_-(x) = \varphi(x)$, where $\varphi(x)$ is the throughflow of node x.

Obviously, $\varphi_+(G) = \varphi_-(G) = \varphi(G)$, where $\varphi(G)$ is the throughflow of G.

The above formulas can be considered as *flow conservation equations* [2].

Example: Assume that there are 100 play blocks in the collection; 60 are triangular, 40 are square, 70 are blue, 10 are red, 20 are green, 10 are small and 90 are large. Flow graph for the set of play blocks is presented in Fig. 1. We see that there are 45 triangular and blue play blocks, etc. Thus the flow gives clear picture of the relationship between different features of play blocks. □

If we replace flow by relative flow with respect to total flow, we obtain a *normalized flow graph* – a *directed acyclic finite* graph $G = (N, \mathcal{B}, \sigma)$, where, as before,

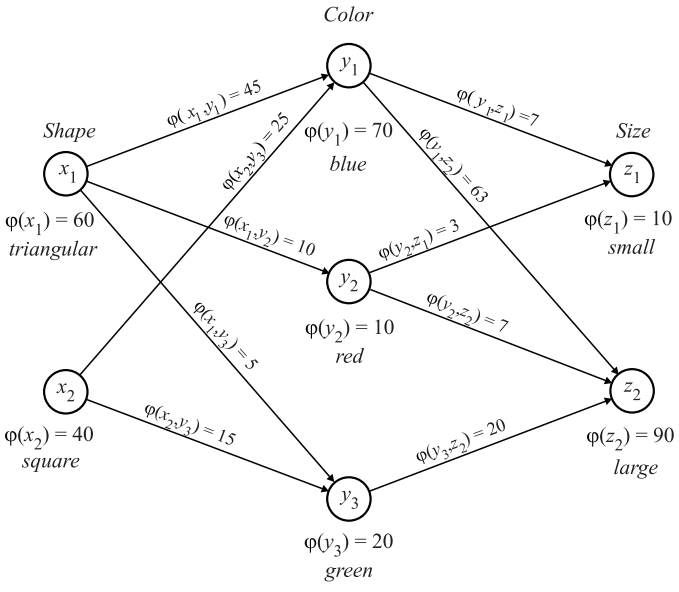

Fig. 1. Flow graph

N is a set of *nodes*, $\mathcal{B} \subseteq \mathrm{N} \times \mathrm{N}$ is a set of *directed branches*, but instead of $\varphi : \mathcal{B} \to \mathrm{R}^+$ we have a *normalized flow* defined by

$$\sigma(x, y) = \frac{\varphi(x, y)}{\varphi(G)} \tag{8}$$

for any $(\mathrm{x}, \mathrm{y}) \in \mathcal{B}$.

The value of $\sigma(\mathrm{x}, \mathrm{y})$ is called the *strength* of (x, y). Obviously, $0 \leq \sigma(x, y) \leq 1$. The strength of the branch expresses simply the ratio of throughflow of the branch to the total flow.

Normalized graphs have interesting properties which are discussed next. In what follows we will use normalized flow graphs only, therefore by flow graphs we will understand normalized flow graphs, unless stated otherwise.

For the sake of further study, if we invert all arrows in a flow graph the new resulting flow graph will be called *inverse*.

With every node x of a flow graph G we associate its *normalized inflow*

$$\sigma_+(\mathrm{x}) = \frac{\varphi_+(\mathrm{x})}{\varphi(\mathrm{G})} = \sum_{\mathrm{y} \in \mathrm{I}(\mathrm{x})} \sigma(\mathrm{y}, \mathrm{x}) \tag{9}$$

and *normalized outflow*

$$\sigma_-(\mathrm{x}) = \frac{\varphi_-(\mathrm{x})}{\varphi(\mathrm{G})} = \sum_{\mathrm{y} \in \mathrm{O}(\mathrm{x})} \sigma(\mathrm{x}, \mathrm{y}) \tag{10}$$

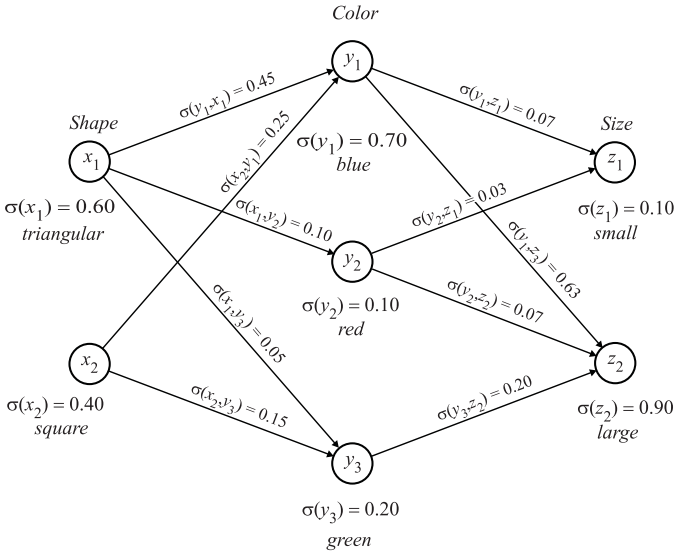

Fig. 2. Normalized flow graph

Obviously, for any internal node x, we have $\sigma_+(\mathrm{x}) = \sigma_-(\mathrm{x}) = \sigma(\mathrm{x})$, where $\sigma(\mathrm{x})$ is a *normalized throughflow* of x. Moreover, let

$$\sigma_+(\mathrm{G}) = \frac{\varphi_+(\mathrm{G})}{\varphi(\mathrm{G})} = \sum_{\mathrm{x}\in \mathrm{I(G)}} \sigma_-(\mathrm{x}) \tag{11}$$

$$\sigma_-(\mathrm{G}) = \frac{\varphi_-(\mathrm{G})}{\varphi(\mathrm{G})} = \sum_{\mathrm{x}\in \mathrm{O(G)}} \sigma_+(\mathrm{x}) \tag{12}$$

Obviously, $\sigma_+(G) = \sigma_-(G) = \sigma(G) = 1$.

A (*directed*) *path* from x to y, $\mathrm{x} \neq \mathrm{y}$, in G is a sequence of nodes $\mathrm{x}_1, \ldots, \mathrm{x}_\mathrm{n}$ such that $\mathrm{x}_1 = \mathrm{x}$, $\mathrm{x}_\mathrm{n} = \mathrm{y}$ and $(\mathrm{x}_\mathrm{i}, \mathrm{x}_{\mathrm{i}+1}) \in \mathcal{B}$ for every $\mathrm{i}, 1 \leq \mathrm{i} \leq \mathrm{n}-1$. A path from x to y is denoted by $[\mathrm{x} \ldots \mathrm{y}]$ and $\mathrm{n}-1$ is called *length* of the path.

A flow graph is *linear* if all paths from node x to node y have the same length, for every pair of nodes x, y.

A set of nodes of a linear flow graph is called a k-*layer* if it consists of all nodes of this graph linked by a path of the length k with some input node.

The set of all inputs will be called the *input layer* of the flow graph, whereas the set of all outputs is the *output layer* of the flow graph. For any input node x and output node y of a linear graph, the length of $[\mathrm{x} \ldots \mathrm{y}]$ is the same. The layers different than input and output layers will be referred to as to *hidden layers*.

Example (cont.): Fig. 2 shows normalized flow graph for the play blocks. We have three layers $\{x_1, x_2\}$, $\{y_1, y_2, y_3\}$ and $\{z_1, z_2\}$, where $\{x_1, x_2\}$ is the input layer, $\{z_1, z_2\}$ is the output layer and $\{y_1, y_2, y_3\}$ is the hidden layer. □

4 Certainty and Coverage Factors

With every branch (x, y) of a flow graph G we associate the *certainty factor*

$$\mathrm{cer(x,y)} = \frac{\sigma(\mathrm{x,y})}{\sigma(\mathrm{x})} \tag{13}$$

and the *coverage factor*

$$\mathrm{cov(x,y)} = \frac{\sigma(\mathrm{x,y})}{\sigma(\mathrm{y})} \tag{14}$$

where $\sigma(x) \neq 0$ and $\sigma(y) \neq 0$.

These coefficients are widely used in data mining (see, e.g., [5,11,12]) but they can be traced back to Łukasiewicz [4], who used them first in connection with his research on logic and probability.

If we interpret nodes of a flow graph as subsets of a fixed set U (the universe), then cer(x, y) can be understood as the degree of inclusion of x in y [9], while cov(x, y) – as the degree of inclusion of y in x, for any sets x, y, where:

$$\mathrm{cer\,(X,Y)} = \begin{cases} \frac{|X \cap Y|}{|X|} & \text{if } X \neq \emptyset \\ 1 & \text{if } X = \emptyset \end{cases}$$

$cov(X, Y) = cer(Y, X)$, and $|x|$ denotes the cardinality of set x.

Observe that by x, y we denote both nodes of the flow graph and subsets of the universe U. However, it does not lead to confusion, because it is always clear from the context when we speak about nodes or sets.

Assume that if $\{\mathrm{x_1}, \ldots, \mathrm{x_n}\}$ is a layer then $\mathrm{x_i} \cap \mathrm{x_j} = \emptyset$ for any $\mathrm{x_i} \neq \mathrm{x_j}$, and $\sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{x_i} = \mathrm{U}$, i.e., every layer is a partition of the universe.

Consequently, the union of all inputs x of y can be understood as the upper approximation of y and the union of all inputs x of y such that $\mathrm{cer(x,y)} = 1$ as the lower approximation of y.

In what follows, we will interpret layers as attributes in information systems, input and hidden layers are interpreted as condition attributes, whereas output layer is interpreted as the decision attribute.

Example (cont.): Fig. 3 illustrates certainty and coverage factors for previously considered example. Here, the input layer represents condition attribute *shape*, the hidden layer – the condition attribute *color*, whereas the output layer – the decision attribute *size*. We can see from the graph that the lower approximation of $\mathrm{z_1}$ is the empty set, whereas the upper approximation of $\mathrm{z_1}$ is $\mathrm{y_1} \cup \mathrm{y_2}$. The lower approximation of $\mathrm{z_2}$ is $\mathrm{y_3}$, whereas the upper approximation is $\mathrm{y_1} \cup \mathrm{y_2} \cup \mathrm{y_3}$. The lower approximation of $\mathrm{y_1}$ is the empty set, and the upper approximation is $\mathrm{x_1} \cup \mathrm{x_2}$. For the set $\mathrm{y_2}$ both approximations are equal $\mathrm{x_1}$, etc.

From the inverse flow graph we get, e.g., that the lower approximation of $\mathrm{x_2}$ is the empty set, the upper approximation is the set $\mathrm{y_1} \cup \mathrm{y_3}$, and the lower and upper approximations of $\mathrm{y_3}$ are both equal to $\mathrm{z_2}$. □

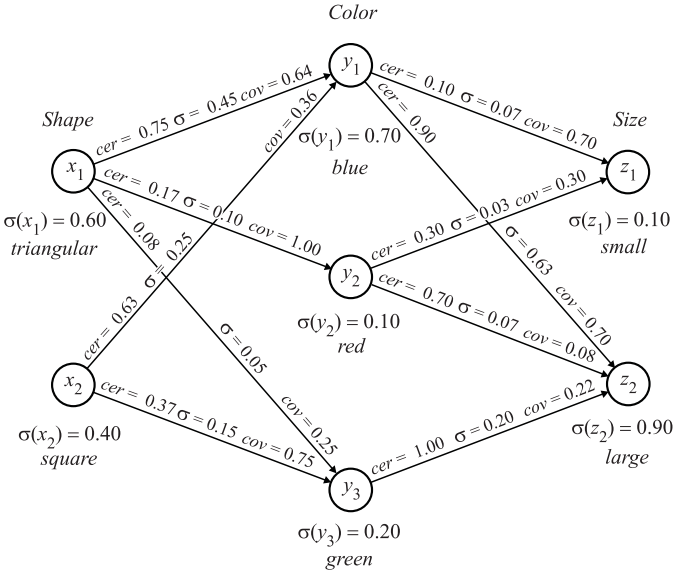

Fig. 3. Certainty and coverage factors

The following properties are immediate consequences of definitions given above:

$$\sum_{\mathrm{y}\in \mathrm{O(x)}} \mathrm{cer(x,y)} = 1 \tag{15}$$

$$\sum_{\mathrm{x}\in \mathrm{I(y)}} \mathrm{cov(x,y)} = 1 \tag{16}$$

$$\sigma(\mathrm{x}) = \sum_{\mathrm{y}\in \mathrm{O(x)}} \mathrm{cer(x,y)}\sigma(\mathrm{x}) = \sum_{\mathrm{y}\in \mathrm{O(x)}} \sigma(\mathrm{x,y}) \tag{17}$$

$$\sigma(\mathrm{y}) = \sum_{\mathrm{x}\in \mathrm{I(y)}} \mathrm{cov(x,y)}\sigma(\mathrm{y}) = \sum_{\mathrm{x}\in \mathrm{I(y)}} \sigma(\mathrm{x,y}) \tag{18}$$

$$\mathrm{cer(x,y)} = \frac{\mathrm{cov(x,y)}\sigma(\mathrm{y})}{\sigma(\mathrm{x})} \tag{19}$$

$$\mathrm{cov(x,y)} = \frac{\mathrm{cer(x,y)}\sigma(\mathrm{x})}{\sigma(\mathrm{y})} \tag{20}$$

The above properties have a probabilistic flavor, e.g., equations (17) and (18) have a form of total probability theorem, whereas formulas (19) and (20) are

Bayes' rules [10]. However, in our approach, these properties are interpreted in a deterministic way and they describe flow distribution among branches in the network.

The *certainty*, *coverage*, and *strength* of the path $[x_1 \dots x_n]$ are defined as

$$\mathrm{cer}[\mathrm{x}_1 \dots \mathrm{x}_\mathrm{n}] = \prod_{\mathrm{i}=1}^{\mathrm{n}-1} \mathrm{cer}(\mathrm{x}_\mathrm{i}, \mathrm{x}_{\mathrm{i}+1}) \tag{21}$$

$$\mathrm{cov}[\mathrm{x}_1 \dots \mathrm{x}_\mathrm{n}] = \prod_{\mathrm{i}=1}^{\mathrm{n}-1} \mathrm{cov}(\mathrm{x}_\mathrm{i}, \mathrm{x}_{\mathrm{i}+1}) \tag{22}$$

$$\sigma[\mathrm{x} \dots \mathrm{y}] = \sigma(\mathrm{x})\mathrm{cer}[\mathrm{x} \dots \mathrm{y}] = \sigma(\mathrm{y})\mathrm{cov}[\mathrm{x} \dots \mathrm{y}] \tag{23}$$

respectively.

5 Flow Graph and Decision Algorithms

Flow graphs can be interpreted as decision algorithms [7].

Let us assume that the set of nodes of a flow graph is interpreted as a set of logical formulas. The formulas are understood as propositional functions and if x is a formula, then $\sigma(\mathrm{x})$ is to be interpreted as a truth value of the formula. Let us observe that the truth values are numbers from the closed interval $< 0, 1 >$, i.e., $0 \leq \sigma(\mathrm{x}) \leq 1$.

According to [4] these truth values can be also interpreted as probabilities. Thus $\sigma(\mathrm{x})$ can be understood as flow distribution ratio (percentage), truth value or probability. We will stick to the first interpretation.

With every branch (x, y) we associate a decision rule $\mathrm{x} \to \mathrm{y}$, read *if* x *then* y; x will be referred to as to *condition*, whereas y – *decision* of the rule. Such a rule is characterized by three numbers: $\sigma(\mathrm{x}, \mathrm{y})$, $\mathrm{cer}(\mathrm{x}, \mathrm{y})$, and $\mathrm{cov}(\mathrm{x}, \mathrm{y})$.

Table 1. Decision algorithm

	certainty	coverage	strength
$x_1, y_1 \to z_1$	0.10	0.45	0.05
$x_1, y_1 \to z_2$	0.90	0.45	0.41
$x_1, y_2 \to z_1$	0.30	0.30	0.03
$x_1, y_2 \to z_2$	0.70	0.08	0.07
$x_1, y_3 \to z_2$	1.00	0.06	0.05
$x_2, y_1 \to z_1$	0.10	0.25	0.02
$x_2, y_1 \to z_2$	0.90	0.25	0.23
$x_2, y_3 \to z_2$	1.00	0.17	0.15

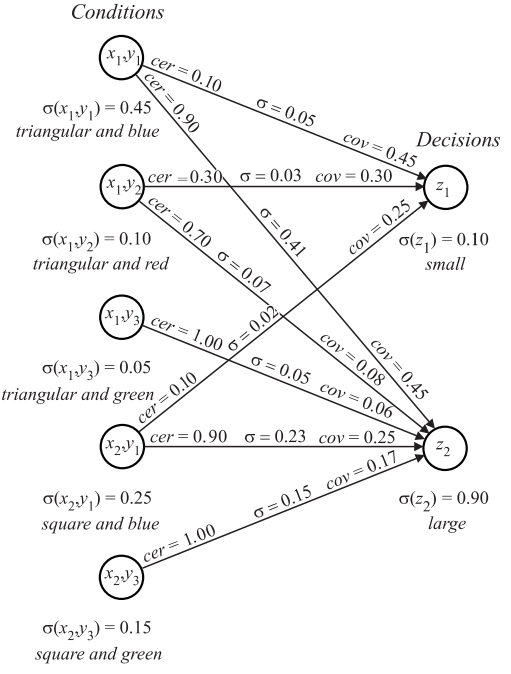

Fig. 4. Flow graph of the decision algorithm

Thus every path $[x_1 \dots x_n]$ determines a sequence of decision rules $x_1 \to x_2$, $x_2 \to x_3, \dots, x_{n-1} \to x_n$. From previous considerations it follows that such sequence can be interpreted as a single decision rule $x_1x_2 \dots x_{n-1} \to x_n$, in short $x^* \to x_n$, where $x^* = x_1x_2 \dots x_{n-1}$, characterized by

$$\mathrm{cer}(x^*, x_n) = \frac{\sigma(x^*, x_n)}{\sigma(x^*)} \tag{24}$$

$$\mathrm{cov}(x^*, x_n) = \frac{\sigma(x^*, x_n)}{\sigma(x_n)} \tag{25}$$

where

$$\sigma(x^*, x_n) = \sigma[x_1, \dots, x_{n-1}, x_n] \text{ and } \sigma(x^*) = \sigma[x_1, \dots, x_{n-1}] \tag{26}$$

The set of all decision rules $x_{i_1}x_{i_2} \dots x_{i_{n-1}} \to x_{i_n}$ associated with paths $[x_{i_1} \dots x_{i_n}]$, such that x_{i_1} and x_{i_n} are input and output of the flow graph, respectively, will be called a *decision algorithm* induced by the flow graph.

Example (cont.): Decision algorithm induced by the flow graph given in Fig. 3 is shown in Table 1. With the decision algorithm we can associate a flow graph as

shown in Fig. 4. Observe that the values of coefficients in Fig. 4 may not satisfy exactly formulas (15)-(18) due to the round off errors in the computations.

One can see that the lower approximation of set z_1 is the empty set and the upper approximation of z_1 is $(x_1 \cap y_1) \cup (x_1 \cap y_2) \cup (x_2 \cap y_1)$. The lower approximation of z_2 is $(x_1 \cap y_3) \cup (x_2 \cap y_3)$ and its upper approximation is equal to $(x_1 \cap y_1) \cup (x_1 \cap y_2) \cup (x_1 \cap y_3) \cup (x_2 \cap y_1) \cup (x_2 \cap y_3)$. □

6 Conclusion

We have shown in this paper that approximations, basic operations in rough set theory, which are in fact topological interior and closure operations defined in algebraic terms, can be also defined in terms of flow intensity in a flow graph. If we associate with every node of a flow graph a subset of a fixed universe then the flow graph induces a relational structure such that two nodes connected by a branch may be interpreted as partial inclusion of the corresponding subsets.

In particular, if X is included in Y then X belongs to the lower approximation of Y and if X is partially included in Y then X belongs to the upper approximation of Y. Thus, the flow graph can be interpreted as family of lower and upper approximations of subsets associated its nodes. This leads to a very simple method of computing approximations without involving set theoretical operations but employing only certainty and coverage coefficients.

This can be specially useful when data are given in a form of a decision table, and the associated flow graph can be easily used to compute approximations and consequently – decision rules (sure and possible).

This idea can be also formulated simpler by defining approximation of nodes in a flow graph – instead of approximation of sets associated with nodes of the flow graph. However, we have not consider this idea in this paper.

Finally, we would like to present some research topics related to flow graphs:

1. *Extracting relevant flow graphs from data.* Flow graphs derived from data tables can be treated as a form of knowledge representation encoded in these tables. Reasoning based on flow graphs can be much more efficient than reasoning performed directly from data tables. However, one should develop algorithms for extracting from data flow graphs that make it possible to perform such reasoning with the satisfactory quality.
2. *Developing case-based reasoning methods for cases with decisions represented by flow graphs.* In particular, this includes developing methods for incremental learning with flow graphs as compound decisions.
3. *Reasoning about changes of flow graphs.* Flow graphs can also be used in reasoning about changes. This will require developing algorithms for inducing rules predicting changes of flow graphs from properties of changing data.

Acknowledgments

Thanks are due to Prof. Andrzej Skowron and Dr. Dominik Ślęzak for critical remarks.

References

1. Ford, L.R., Fulkerson, D.R.: Flows in Networks. Princeton University Press, Princeton. New Jersey (1962)
2. Greco, S., Pawlak, Z., Słowiński, R.: Generalized decision algorithms, rough inference rules and flow graphs. In: J.J. Alpigini, J.F. Peters, A. Skowron, N. Zhong (eds.), Rough Sets and Current Trends in Computing. Lecture Notes in Artificial Intelligence 2475, Springer-Verlag, Berlin (2002) 93-104
3. Greco, S., Pawlak, Z., Słowiński, R.: Bayesian confirmation measures within rough set approach. In: S. Tsumoto, R. Słowiński, J. Komorowski, J. Grzymała-Busse (eds.), Rough Sets and Current Trends in Computing (RSCTC 2004). Lecture Notes in Artificial Intelligence 3066, Springer Verlag, Berlin (2004) 261-270
4. Łukasiewicz, J.: Die logishen Grundlagen der Wahrscheinilchkeitsrechnung. Kraków (1913). In: L. Borkowski (ed.), Jan Łukasiewicz – Selected Works. North Holland Publishing Company, Amsterdam, London, Polish Scientific Publishers, Warsaw (1970) 16-63
5. E. Kloesgen, J. Żytkow (eds.): Handbook of Knowledge Discovery and Data Mining. Oxford University Press, Oxford, UK (2002)
6. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. System Theory, Knowledge Engineering and Problem Solving 9, Kluwer Academic Publishers, Dordrecht (1991)
7. Pawlak, Z.: Rough sets, decision algorithms and Bayes' theorem. European Journal of Operational Research, 136 (2002) 181-189
8. Pawlak, Z.: Flow graphs and decision algorithms. In: G. Wang, Y. Yao, A. Skowron (eds.), Rough Sets, Fuzzy Sets, Data Mining and Granular Computing (RSFDGrC 2003). Lecture Notes in Artificial Intelligence, 2639, Springer Verlag, Berlin (2003) 1-10
9. Polkowski, L., Skowron, A.: Rough mereology: A new paradigm for approximation reasoning. International Journal of Approximate Reasoning, 15(4) (1996) 333-365
10. Swinburne, R. (ed.): Bayes's Theorem. In: Proceedings of the British Academy, 113, Oxford University Press (2002)
11. Tsumoto, S.: Modelling medical diagnostic rules based on rough sets. In: L. Polkowski, A. Skowron (eds.), Rough Sets and Current Trends in Computing (RSCTC'98). Lecture Notes in Artificial Intelligence 1424, Springer Verlag, Berlin (1998) 475-482
12. Wong, S.K.M., Ziarko, W.: Algorithm for inductive learning. Bull. Polish Academy of Sciences 34(5-6) (1986) 271-276