Conflict Analysis and Information Systems: A Rough Set Approach

Andrzej Skowron[1], Sheela Ramanna[2], and James F. Peters[3]

[1] Institute of Mathematics,
Warsaw University
Banacha 2, 02-097 Warsaw, Poland
skowron@mimuw.edu.pl
[2] Department of Applied Computer Science,
University of Winnipeg,
Winnipeg, Manitoba R3B 2E9 Canada
s.ramanna@uwinnipeg.ca
[3] Department of Electrical and Computer Engineering,
University of Manitoba
Winnipeg, Manitoba R3T 5V6 Canada
jfpeters@ee.umanitoba.ca

Abstract. Conflict analysis and conflict resolution play an important role in negotiation during contract-management situations in government and industry. The problem to be solved is how to model conflict situations where there is uncertainty about agreement, neutrality and disagreement among agents in a conflict situation. The solution to this problem includes modeling a conflict situation relative to basic binary relations on a universe of agents, introducing a measure of the degree of conflict, and encapsulating a conflict situation in an information system. The basic approach to modeling conflict situations is illustrated in the context of contract negotiation during the initial phases of requirement negotiation for a systems engineering project. An example of a high-level requirements negotiation for an automated lighting system is presented. The contribution of this paper is a rough set based requirements determination model using a conflict relation for representing requirements agreements (or disagreements).

Keywords: Conflict, conflict graph, conflict resolution, negotiation, requirements engineering, rough sets.

1 Introduction

Conflict analysis and resolution play an important role in government and industry where disputes and negotiations about various issues are the norm. To this end, many mathematical formal models of conflict situations have been proposed and studied, e.g., [2,4,6,10,11,14,20,19,18]. More recently, conflict analysis as a basic issue in e-service intelligence has been proposed by [15]. Knowledge discovery in databases consists of searching for functional dependencies in the data set. The approach used in this paper, is based on a different kind of relationship

G. Wang et al. (Eds.): RSKT 2006, LNAI 4062, pp. 233–240, 2006.

in the data. This relationship is not a dependency, but a conflict [15]. Formally, a conflict relation can be viewed as a special kind of discernibility, i.e., negation (not necessarily, classical) of indiscernibility relation which is the basis of rough set theory [13]. Thus indiscernibility and conflict are closely related from logical point of view. It is also interesting to note that almost all mathematical models of conflict situations are strongly domain dependent. Previous work on the application of rough sets to conflict resolution and negotiations between agents made it possible to introduce approximate reasoning about vague concepts [15]. Recent work in the application of rough sets to handling uncertainty in software requirements can be found in [9]. Rough sets have also been applied to acceptance of software designs [16], analysis of software quality data [17]. However, the basic assumption in all of these papers, is that requirements have already been *decided* and the analysis of gathered requirements data is then performed.

By way of illustration of the rough set approach to conflict analysis and resolution, sample negotiation typically found during a system requirements engineering (SRE) project is considered. SRE is that portion of software engineering that focuses on the functional and non-functional requirements to be included in a system. The study of conflicts in software engineering has been studied extensively (see, e.g., [3,5,7]). A typical requirements negotiation process for a large system requires intense collaboration between project stakeholders that begins with requirements identification and leads to negotiated commitments by all concerned. In this paper, our approach is to represent and analyze $conflicts$ during a requirements-gathering process even before the requirements are *decided*. This entails representing and analyzing conflicts during a collaborative process of requirements identification by all stakeholders of a project. Our approach is based on the Win-Win approach [1,21]. The Win-Win approach has two principal features. First, one defines a decision rationale model using a minimal set of conceptual elements, such as win conditions, issues, options and agreements, that serves as an agreed upon ontology for collaboration and negotiation. Second, one defines a support framework to reason about decision rationale.

The contribution of this paper is a rough set based requirements determination model using a conflict relation for representing requirements agreements (or disagreements). Conflict graphs are used to analyze conflict situations, reason about the degree of conflict and explore coalitions. We illustrate our approach in determining high-level requirements of a complex engineering system through negotiation.

This paper is organized as follows. An introduction to basic concepts is given Sect. 2. Conflicts and information systems are discussed in Sect. 3. Sect. 4 begins with a model for a conflict situation during requirements identification, followed by an illustration high-level requirements negotiation for an automated lighting system in Sect. 4.1. Analysis of requirements conflicts are discussed in Sect. 4.2.

2 Basic Concepts of Conflict Theory

The basic concepts of conflict theory that we use in this paper are due to [15]. Let us assume that we are given a finite, non-empty set Ag called the *universe*.

Elements of Ag will be referred to as *agents*. Let a *voting function* $v : Ag \to \{-1, 0, 1\}$, or in short $\{-, 0, +\}$, be a number representing his/her voting result about some issue under negotiation, to be interpreted as *against,neutral* and *favorable*, respectively. The pair $CS = (Ag, V)$, where V is a set of voting functions, will be called a *conflict situation*.

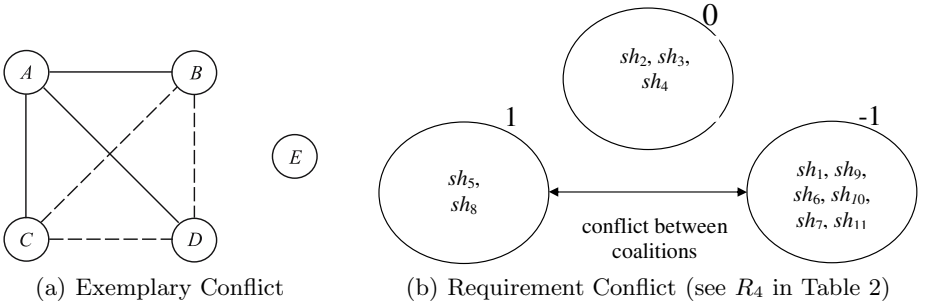
In order to express relations between agents, we define three basic binary relations on the universe: *agreement, neutrality*, and *disagreement*. To this end, for a given voting function v, we first define the following auxiliary function:

$$\phi_v(ag, ag') = \begin{cases} 1, & \text{if } v(ag)v(ag') = 1 \text{ or } ag = ag' \\ 0, & \text{if } v(ag)v(ag') = 0 \text{ and } ag \neq ag' \\ -1, & \text{if } v(ag)v(ag') = -1. \end{cases} \quad (1)$$

This means that, if $\phi_v(ag, ag') = 1$, agents ag and ag' have the same opinion about an issue v (*agree* on issue v); if $\phi_v(ag, ag') = 0$ means that at least one agent ag or ag' has no opinion about an issue v (is *neutral* on v), and if $\phi_v(ag, ag') = -1$, means that both agents have different opinions about an issue v (are in *conflict* on issue v). In what follows, we will define three basic relations R_v^+,R_v^0 and R_v^- on Ag^2 called *agreement, neutrality* and *disagreement* relations respectively, and defined by (i) $R_v^+(ag, ag')$ iff $\phi_v(ag, ag') = 1$; (ii) $R_v^0(ag, ag')$ iff $\phi_v(ag, ag') = 0$; (iii) $R_v^-(ag, ag')$ iff $\phi_v(ag, ag') = -1$. It is easily seen that the *agreement* relation is an *equivalence* relation. Each equivalence class of the agreement relation will be called a *coalition* with respect to v. For the conflict or disagreement relation we have: (i) not $R_v^-(ag, ag)$; (ii) if $R_v^-(ag, ag')$ then $R_v^-(ag', ag)$; (iii) if $R_v^-(ag, ag')$ and $R_v^+(ag', ag'')$ then $R_v^-(ag, ag'')$. For the neutrality relation we have: (i) not $R_v^0(ag, ag)$; (ii) $R_v^0(ag, ag') = R_v^0(ag', ag)$. In the conflict and neutrality relations there are no coalitions. In addition, $R_v^+ \cup R_v^0 \cup R_v^- = Ag^2$. All the three relations R_v^+, R_v^0 , R_v^- are pairwise disjoint.

With every conflict situation $CS = (Ag, v)$ we will associate a *conflict graph*. Examples of conflict graphs are shown in Figure 1.

In Figure 1(a), solid lines denote conflicts, dotted line denote agreements, and for simplicity, neutrality is not shown explicitly in the graph. As one can see B,



(a) Exemplary Conflict (b) Requirement Conflict (see R_4 in Table 2)

Fig. 1. Sample Conflict Graphs

C, and D form a coalition. A conflict degree $Con(CS)$ of the conflict situation $CS = (Ag, v)$ is defined by

$$Con(CS) = \frac{\sum_{\{(ag,ag'):\ \phi_v(ag,ag')=-1\}} |\phi_v(ag, ag')|}{2\lceil \frac{n}{2} \rceil \times (n - \lceil \frac{n}{2} \rceil)}. \tag{2}$$

where $n = Card(ag)$. Observe that $Con(CS)$ is a measure of discernibility between agents from Ag relative to the voting function v. For a more general conflict situation $CS = (Ag, V)$ where $V = \{v_1, \ldots, v_k\}$ is a finite set of voting functions each for a different issue/requirements the *conflict degree* in CS (*tension generated by* V) can be defined by $Con(CS) = \sum_{i=1}^{k} Con(CS_i)/k$ where $CS_i = (Ag, v_i)$ for $i = 1, \ldots, k$.

3 Conflicts and Information Systems

An information system is a table rows of which are labeled by *objects* (*agents*), columns by *attributes* (*issues*) and entries of the table are *values of attributes* (*votes*), which are uniquely assigned to each team member and attribute, i.e. each entry corresponding to row x and column a represents opinion of an agent x about issue a. Formally an *information system* can be defined as a pair $S = (U, A)$, where U is a nonempty, finite set called the *universe*; elements of U will be called *objects* and A is a nonempty, finite set of *attributes* [13]. Every attribute $a \in A$ is a total function $a : U \to V_a$, where V_a is the set of *values* of a, called the *domain* of a; elements of V_a will be referred to as *opinions*, and $a(x)$ is opinion of agent x about issue a. The above given definition is general, but for conflict analysis we will need its simplified version, where the domain of each attribute is restricted to three values only, i.e. $V_a = \{-1, 0, 1\}$, for every a, meaning *disagreement, neutral* and *agreement* respectively. For the sake of simplicity we will assume $V_a = \{-, 0, +\}$. Every information system with the above mentioned restriction will be referred to as a *situation*.

We now observe that any conflict situation $CS = (Ag, V)$ can be treated as an information system where $Ag = \{ag_1, \ldots, ag_n\}$ and $V = \{v_1, \ldots, v_k\}$ with the set of objects Ag (*agents*) and the set V of attributes (*issues*).

4 Requirements Identification and Conflicts

A typical system requirements engineering process leads to conflicts between project stakeholders. A stakeholder is one who has a share or an interest in the requirements for a systems engineering project. Let Ag be represented by the set SH (stakeholders). Let V denote the set of requirements. Let $CS = (SH, V)$ where $SH = \{sh_1, \ldots, sh_n\}$ and $V = \{v_1, \ldots, v_k\}$.

4.1 Example: System Requirements Identification

Cost effective engineering of complex software systems involves a collaborative process of requirements identification through negotiation. This is one of

the key ideas of the Win-Win approach [1] used in requirements engineering. This approach also includes a decision model where a minimal set of conceptual elements, such as win conditions, issues, options and agreements, serves as an agreed upon ontology for collaboration and negotiation defined by the Win-Win process. System requirements (goals) are viewed as conditions. If all members agree on a requirement (i.e., no conflicts), then that requirement becomes an agreement. Otherwise, the requirement becomes an issue for further negotiation. Each issue could have an option (i.e., an alternate requirement) suggested by the team. We illustrate our ideas with a problem of achieving agreement on high-level system requirements for a home lighting automation system (HLAS) [8]. The initial HLAS requirements user group consists of members drawn from a stakeholders list which is comprised of builders, distributors, electrical contractors, homeowners, system development team, marketing team and management. The user group (the set SH of agents) prepares the preliminary list of requirements. Then a questionnaire based survey (on a wide audience) is conducted and the result of the initial votes is presented in Table 1. Let $R = \{R_i \mid 1 \leq i \leq 16\}$ denote a set of project requirements shown in Table 1. Support for each requirement from members of SH is indicated by

Table 1. Initial Requirements

Voting Results		
ID	Requirements	Votes or Support
R_1	Custom Lighting Scenes	120
R_2	Automatic Time Setting for lights	110
R_3	Built-in security features	104
R_4	100% System Reliability	100
R_5	Vacation Setting	95
R_6	Easy-to-program non-PC control unit	93
R_7	Any light can be dimmed	90
R_8	Interface to Home Security System	80
R_9	Voice Activation	70
R_{10}	Close garage doors	67
R_{11}	Easy to Install	55
R_{12}	Easily expanded when remodeling	39
R_{13}	Automatically turn on lights when someone approaches the door	60
R_{14}	Restore after power fail	30
R_{15}	International User Interface	10
R_{16}	Control Lighting via phone	43

the number of votes for each option. Votes (Support) for each requirement is defined as: $Votes\,(CS_v) = \sum_{\{ag \in SH : v(ag)=1\}} 1$ where $CS_v = (SH, v)$, $v \in R$. Hence, we are counting the number of votes for the issue v by members of SH. After the initial round of voting, the HLAS requirements user group decides to prioritize the requirements where *requirements* with $card(Votes(CS_v))$ less than 40 will be discarded. We now have a new conflict group (situation) with

a smaller set of team members SH' and a subset of requirements defined as follows: $CS' = (SH', V')$, where $V' = \{R_1, \ldots, R_{11}, R_{13}\}$. The new user group SH' will now consist of sh_1, sh_2 representing electrical contractors responsible for installation and support, sh_3, sh_4 representing builders who are general contractors responsible to the homeowners, sh_5 is a marketer of the product, sh_6, sh_7, sh_8, sh_9 representing the systems development team and sh_{10}, sh_{11} representing the management that is responsible for approving funding for the project. The new user group (team) will vote on the new set of requirements (win conditions) to establish agreements. The voting result is given in Table 2. From the voting results the indiscernibility relations $Ind_{R_i}(V')$ for $i = 1, \ldots,$

Table 2. Win Conditions

	Voting Results											
SH'	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9	R_{10}	R_{11}	R_{13}
sh_1	0	1	0	-1	1	1	0	1	-1	1	1	0
sh_2	0	1	0	0	1	1	0	1	-1	1	1	0
sh_3	1	1	1	0	1	1	1	1	0	1	0	1
sh_4	1	1	1	0	1	1	1	1	0	1	0	1
sh_5	1	1	1	1	1	1	1	1	1	1	0	1
sh_6	1	1	1	-1	1	0	1	1	-1	1	1	1
sh_7	1	1	1	-1	1	0	1	1	1	1	1	1
sh_8	1	1	1	1	1	0	1	1	-1	1	1	1
sh_9	1	1	1	-1	1	0	1	1	-1	1	1	1
sh_{10}	0	1	1	-1	1	1	1	0	-1	0	-1	1
sh_{11}	0	1	1	-1	1	1	1	0	-1	1	-1	1

$11, 13$ (see [13]) identified by partitions of V' are defined. For example: $Ind_{R_1}(V')$ $= \{\{sh_3, sh_4, sh_5, sh_6, sh_7, sh_8, sh_9\}, \{sh_1, sh_2, sh_{10}, sh_{11}\}\}$.

Algorithm 1. Algorithm for determining win agreements

Input : Equivalence Classes (EC) defined by Ind_{R_1}, Ind_{R_2}, ..., $Ind_{R_{11}}$, $Ind_{R_{13}}$

Output: Non-conflicting requirements from $\{R_1, \ldots, R_{11}, R_{13}\}$

(all $e \in EC$*) select* e *where* $e = R_i^{-1}\{0\} = \{sh \in SH' : R_i(sh) = 0\}$ *or* $e = R_i^{-1}\{1\} = \{sh \in SH' : R_i(sh) = 1\}$ *and* $i \in \{1, \ldots, 11, 13\}$;

// select all equivalence classes corresponding to the values 0 or 1 of voting functions

The output of Alg. 1 will now consist of a set of requirements that are deemed as agreement between all stakeholders. This means that the team disagrees on the following three requirements: R_4, R_9 and R_{11}. Also, note that an abstention (vote of 0) for any requirement is considered a tacit approval for the purposes of requirements negotiation. The conflict graph $CS'_{R_4} = (SH', R_4)$ can be presented

in a simplified form as a graph with nodes represented by coalitions and edges representing conflicts between coalitions as shown in Fig. 1(b).

4.2 Requirements Conflict Analysis and Negotiation

Since win conflict negotiation necessitates agreement on each requirement, we do not use the definition of a more general conflict situation. From this graph, one can compute the conflict degree using using Eqn. 2 where $Con(CS'_{R_4}) = 0.4$. The degree of conflict for the remaining two requirements are $Con(CS'_{R_9}) = 0.5$ and $Con(CS'_{R_{11}}) = 0.4$. Clearly, there is disagreement over the following requirements: 100% System Reliability, Voice Activation and Ease of installation. This indicates that the team is not comfortable with such stringent (100% reliability) or unclear (easy to install) requirements. Since this situation calls for a new round of negotiations with a new set of options (modified requirements), it would interesting to look at coalitions.

The conflict degree $Con(CS')$ in $CS' = (SH', V')$ (tension generated byV') for this round of negotiations can be calculated using formula for $Con(CS)$ and is equal to 13/120. This means that we have a new conflict situation defined as follows: $CS'' = (SH', V'')$ where V'' represents new options for the three requirements. The options could include a more granular definition of reliability (e.g., 80 to 90% or 80 to 85%), a more quantifiable definition of ease of installation requirement (e.g., installation time between 3-5 hours). Notice we retain the same number of team members (SH'). So the voting and negotiation continues until there is complete agreement on the high-level requirements for the system.

5 Conclusion

This paper introduces rough set based requirements determination model using the notion of conflict relations for representing requirements agreements, disagreements and neutrality. Conflict graphs are used to analyze conflict situations, reason about the degree of conflict and explore coalitions. An application of this approach is given using a complete example of a home lighting automation system high level requirements. The model takes into account only the first level of negotiation. However, this can be extended to the second level, where each agreement (requirement) will now consist of several low-level requirements. In other words, there will be an implicit hierarchical relationship between requirements. So the stakeholders will now have to negotiate by voting on the lower-level requirements. At the lower level, coalitions (like-minded team members) and conflict degrees amongst coalitions become important. The proposed attempt to conflict analysis offers deeper insight into structure of conflicts, enables analysis of relationship between stakeholders and requirements being debated. Finally, the simplicity of the mathematical model of conflicts considered, suggests the possibility of automated tool support for requirements negotiation.

Acknowledgments. The authors gratefully acknowledge suggestions by Zdzisław Pawlak about conflict graphs. The research of Andrzej Skowron has been supported

by grant 3 T11C 002 26 from Ministry of Scientific Research and Information Technology of the Republic of Poland. The research of Sheela Ramanna and James F. Peters is supported by NSERC grants 194376 and 185986, respectively.

References

1. Boehm, B., Grnbacher, P., Kepler, J.: Developing Groupware for Requirements Negotiation: Lessons Learned, *IEEE Software,* May/June (2001) 46-55.
2. Casti, J.L: Alternative Realities – Mathematical Models of Nature and Man, John Wiley and Sons (1989).
3. Cohene, T., Easterbrook, S.: Contextual risk analysis for interview design, Proc. 13th IEEE Int. Requirements Eng. Conference (RE'05), Paris (2005) 1-10.
4. Coombs, C.H., Avrunin, G.S.: The Structure of Conflict, Lawrence Erlbaum Associates (1988).
5. Curtis,B., Krasner, H., Iscoe, N.: A field study of the software design process for large systems, Communications of the ACM **31**(11) (1988) 1268-1287.
6. Deja, R., Skowron, A.: On Some Conflict Models and Conflict Resolutions, *Romanian Journal of Information Science and Technology* **5**(1-2) (2002) 69-82.
7. Easterbrook, S.: Handling Conflict between Domain Descriptions with Computer-Supported Negotiation, *Knowledge Acquisition* **3** (1991) 255-289.
8. Leffingwell, D. Widrig, D.: Managing Software Requirements, Addision-Wesley (2003).
9. Li, Z., Ruhe, G.: *Uncertainty handling in Tabular-Based Requirements Using Rough Sets,* LNAI **3642**. Springer, Berlin (2005) 678-687.
10. Maeda, Y., Senoo, K., Tanaka, H.: *Interval density function in conflict analysis,* LNAI **1711**, Springer-Verlag, Berlin (1999) 382-389.
11. Nakamura, A.: Conflict logic with degrees. In: S. K. Pal, A. Skowron (Eds.), *Rough Fuzzy Hybridization: A New Trend in Decision-Making,* Springer-Verlag (1999) 136-150.
12. Pawlak, Z.: On Conflicts, Int. J. of Man-Machine Studies **21** (1984) 127-134.
13. Pawlak, Z.: *Rough Sets – Theoretical Aspects of Reasoning about Data,* Kluwer Academic Publishers (1991).
14. Pawlak, Z.: An inquiry into anatomy of conflicts, *Journal of Information Sciences*, **109** (1998) 65-78.
15. Pawlak, Z., Skowron, A.: Rough Sets and Conflict Analysis. *E-Service Intelligence, Series on Computational Intelligence,* Springer, Berlin (2006) [to appear].
16. Peters, J.F., Ramanna, S.: Approximation Space for Software Models. In: J.F. Peters, A. Skowron (Eds.), *Transactions on Rough Sets I: Journal Subline*, LNCS **3100**, Springer, Berlin (2004)338-354.
17. Peters, J.F., Ramanna, S.: Towards a software change classification system: A rough set approach. *Software Quality Journal* **11** (2003) 121-147.
18. Lai, G., Li, C., Sycara, K., Giampapa, J.: Literature review on multi-attribute negotiations, Technical Report CMU-RI-TR-04-66 (2004) 1-35.
19. Kowalski, R.: A logic-based approach to conflict resolution (2003) 1-28 [manuscript].
20. Kraus, S.: *Strategic Negotiations in Multiagent Environments,*The MIT Press (2001).
21. WINWIN Homepage at *http://sunset.usc.edu/research/WINWIN*