STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume L, Number 2, 2005

COMPUTING DEFAULT EXTENSIONS. A HEURISTIC APPROACH

MIHAIELA LUPEA

Abstract. Default logics represent a simple but a powerful class of nonmonotonic formalisms. The main computational problem specific to these logical systems is a search problem: finding all the extensions (sets of nonmonotonic theorems - beliefs) of a default theory. GADEL is an automated system based on a heuristic approach of the classical default extension computing problem and applies the principles of genetic algorithms to solve the problem. The purpose of this paper is to extend this heuristic approach for computing all type of default extensions: classical, justified, constrained and rational.

1. Introduction

The *nonmonotonic reasoning* is an important part of human reasoning and represents the process of inferring conclusions (only plausible, not necessary true) from incomplete information. Adding new facts may later invalidate these conclusions, called *beliefs*.

The family of default logics is based on first-order logic and introduces a new type of inference rules, called *defaults*. These special inference rules model laws that are true with a few exceptions, formalizing a particular type of nonmonotonic reasoning, called *default reasoning*. The differences among different variants of default logic are caused by the semantics of the defaults. These logical systems are sintactically very simple, but very powerful in their inferential process.

A *default theory* $\Delta = (D, W)$ consists of a set W of consistent formulas of first order logic (the *facts*) and a set D of *default rules*. A *default* has the form $d = \frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}$, where: α is called *prerequisite*, $\beta_1, \ldots, \beta_m$ are called *justifications* and γ is called *consequent*.

A default $d = \frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}$ can be applied and thus derive γ if α is believed and it is consistent to assumed $\beta_1, \ldots, \beta_m$(meaning that $\neg\beta_1, \ldots, \neg\beta_m$ are not believed).

2000 *Mathematics Subject Classification.* 03B79, 68T15, 68T27.

1998 *CR Categories and Descriptors.* I2 [**Artificial Intelligence**]: Logic in artificial intelligence – *default logics, nonmonotonic reasoning, theorem proving* .

Using the classical inference rules and the defaults, the set of facts, W, can be extended with new formulas, called *nonmonotonic theorems* (*beliefs*) obtaining *extensions*.

The set of defaults used in the construction of an extension is called the *generating default set* for the considered extension.

In this paper we will use the following notations ($d = \frac{\alpha:\beta}{\gamma}$):
$Prereq(d) = \alpha$, $Justif(d) = \beta$, $Conseq(d) = \gamma$, $Prereq(D) = \bigcup_{d\in D} Prereq(d)$,
$Justif(D) = \bigcup_{d\in D} Justif(d)$, $Conseq(D) = \bigcup_{d\in D} Conseq(d)$,
$Th(X) = \{A|X \vdash A\}$ the classical deductive closure of the set X of formulas.

The versions (classical, justified, constrained, rational) of default logic try to provide an appropriate definition of consistency condition for the justifications of the defaults, and thus to obtain many interesting and useful properties for these logical systems:

- *Classical default logic* was proposed by Reiter [9]. Due to the individual consistency checking of justifications and thus the loss of implicit assumptions when are constructed the classical extensions, this logical system does not satisfy some desirable formal properties: semi-monotonicity, regularity, existence of extensions, commitment to assumptions.
- *Justified default logic* was introduced by Lukaszewicz [3]. The applicability condition of default rules is strengthen and thus individual inconsistencies between consequents and justifications are detected, but inconsistencies among justifications are neglected. In this logical system the existence of extensions and the semi-monotonicity property is guaranteed.
- *Constrained default logic* was developed by Schaub [10]. The consistency condition is a global one and it is based on the observation that in commonsense reasoning we assume things, we keep track of our assumptions and we verify that they do not contradict each other. This logic is strongly regular, semi-monotonic, strongly commits to assumptions and guarantees the existence of extensions.
- *Rational default logic* was introduced in [7] as a version of classical default logic, for solving the problem of handling disjunctive information. The defaults with mutually inconsistent justifications are never used together in constructing a rational default extension. This logic is strongly regular but does not guarantee the existence of extensions, is not semi-monotonic and does not commit to assumptions.

Automated theorem proving for default logics has began with solving the extension computing problem for particular default theories: normal, ordered semi-normal, and then was extended to general theories. The classical theorem proving methods: resolution, semantic tableaux method, connection method, were incorporated and adapted in automated systems to solve specific tasks:

- **DeRes** [2] computes classical extensions for stratified default theories, using a semantic tableaux propositional prover;
- **Exten** [1] is based on an operational approach for computing classical, justified and constrained extensions;
- **Xray** [12] represents an approach of the query-answering problem in constrained and cumulative default logics;
- **DARR** [5] is a theorem prover for constrained and rational default logics based on a modified version of propositional semantic tableaux method.

Due to its very high level of theoretical complexity ($\sum_2^p -complete$), caused by the great power of the inferrential process, the problem of finding the extensions of a default theory, can be solved in an efficient manner only for particular classes of default theories.

In the paper [8] a heuristic approach of the classical extension computation problem is presented. An efficient automated system, called **GADEL** [8], which computes the classical extensions for propositional default logic using the principles of genetic algorithms was also developed.

The purpose of this paper is to extend this heuristic approach for computing all type of default extensions: classical, justified, constrained and rational.

2. Default logics

The results from [6] show that default theories can be represented by unitary theories (all the defaults have only one justification, $d = \frac{\alpha:\beta}{\gamma}$) in such a way that extensions (classical, justified, constrained, rational) are preserved. In this paper we will use only unitary default theories.

Definition 1. [13] A set X of defaults is *grounded* in the set of facts W if there is an enumeration $\langle d_i \rangle_{i\in I}$ of the defaults from X such that:

$\forall i \in I$ we have $W \cup Prereq(\{d_0, d_1, ..., d_{i-1}\}) \vdash Prereq(d_i)$.

The following theorems provide global characterizations for classical, justified, constrained and rational extensions of a default theory using the generating default sets.

Theorem 1. [11] *Let (D, W) be a default theory, and let E be a set of formulas. E is a* **classical extension** *of (D, W) if and only if $E = Th(W \cup Conseq(D'))$ for a maximal set $D' \subseteq D$ such that D' is grounded in W and the following conditions are satisfied:*

- *For any $\frac{\alpha:\beta}{\gamma} \in D'$: $W \cup Conseq(D') \cup \{\beta\}$ is a consistent set;*
- *For any $\frac{\alpha:\beta}{\gamma} \notin D'$: $W \cup Conseq(D') \cup \{\beta\}$ is inconsistent* ***or*** *$W \cup Conseq(D') \cup \{\neg\alpha\}$ is consistent.*

This theorm shows that the defaults are nonmonotonic inference rules, meaning that conclusions derived using defaults can be later invalidated by adding new facts. The consistency condition for justifications is an individual one.

Theorem 2. [4] *Let (D, W) be a default theory, and let E, J be sets of formulas. (E,J) is a* **justified extension** *of (D, W) if and only if* $E = Th(W \cup Conseq(D'))$ *and* $J = Justif(D')$ *for a maximal set* $D' \subseteq D$ *such that D' is grounded in W and the conditions:*

- $\forall d \in D'$*: the set* $W \cup Conseq(D') \cup Justif(d)$ *is consistent.*

are satisfied.

The justifications of the generating default set satisfy an individual consistency condition (stronger than in the classical default logic) and are memorized in a support set J. Unfortunatly this set may be inconsistent, and thus two formulas of the actual extension E may be derived using contradictory assumptions.

Theorem 3. [10] *Let (D, W) be a default theory, and let E, C be sets of formulas. (E,C) is a* **constrained extension** *of (D, W) if and only if* $E = Th(W \cup Conseq(D'))$ *and* $C = Th(W \cup Conseq(D') \cup Justif(D'))$ *for a maximal set* $D' \subseteq D$ *such that D' is grounded in W and the following condition is satisfied:*

- *the set* $W \cup Conseq(D') \cup Justif(D')$ *is consistent.*

Each constrained extension is generated by a set of defaults whose justifications and consequents are together consistent, and at the same time consistent with the set of facts. The actual extension E is embedded in a consistent context C where all the assumptions (justifications) used in the reasoning process are retained.

Theorem 4. [4] *Let (D, W) be a default theory, and let E, C be sets of formulas. (E,C) is a* **rational extension** *of (D, W) if and only if* $E = Th(W \cup Conseq(D'))$ *and* $C = Th(W \cup Conseq(D') \cup Justif(D'))$ *for a maximal set* $D' \subseteq D$ *such that D' is grounded in W and the conditions:*

- *the set* $W \cup Conseq(D') \cup Justif(D')$ *is consistent;*
- $\forall d \in D \backslash D'$ *we have:*
  - $W \cup Conseq(D') \cup \neg Prereq(d)$ *is consistent* ***or***
  - $W \cup Conseq(D') \cup Justif(D' \cup d)$ *is inconsistent,*

are satisfied.

The theorem above states that the reasoning context C must be consistent and the set of generating defaults must be maximal-active [7] with respect to W and the actual extension E, for a rational extension.

From theorems 1,2,3 and 4 we can conclude that all four types of extensions are deductive closures of the set W (explicit content) and the consequents of the **generating default set *D'***(implicit content).

According to the initial fixed-point definitions of all variants of default logic we have the following definitions for the generating default sets:

Definition 2. Let E_1 be a classical extension, (E_2, J) be a justified extension, (E_3, C_3) be a constrained extension and (E_4, C_4) be a rational extension of the default theory *(D, W)*, then we have:

- $GD^{E_1}_{(D,W)} = \left\{ \frac{\alpha:\beta}{\gamma} \in D | \alpha \in E_1 \textit{ and } E_1 \cup \{\beta\} \textit{ is consistent} \right\}$
  is the *generating default set* for the classical extension E_1;
- $GD^{(E_2,J)}_{(D,W)} = \left\{ \frac{\alpha:\beta}{\gamma} \in D | \alpha \in E_2 \textit{ and } \forall \eta \in J \cup E2, E2 \cup \{\gamma, \eta\} \textit{ consistent} \right\}$
  is the *generating default set* for the justified extension (E_2, J);
- $GD^{(E_3,C_3)}_{(D,W)} = \left\{ \frac{\alpha:\beta}{\gamma} \in D | \alpha \in E_3 \textit{ and } C_3 \cup \{\beta, \gamma\} \textit{ is consistent} \right\}$
  is the *generating default set* for constrained extension (E_3, C_3);
- $GD^{(E_4,C_4)}_{(D,W)} = \left\{ \frac{\alpha:\beta}{\gamma} \in D | \alpha \in E_4 \textit{ and } C_4 \cup \{\beta\} \textit{ is consistent} \right\}$
  is the *generating default set* for the rational extension (E_4, C_4).

From [11] and [13] we have the following result regarding the generating default sets of different types of extensions.

Theorem 5. *The generating default sets for every type of extension are grounded in the set of facts of the default theory.*

Example: In this example we illustrate many types of contradictory information in consequents and justifications of the defaults and show how the versions of default logic solve them. The default theory *(D, W)* with $W = \{F \vee C\}$ and $D = \left\{ d1 = \frac{:A}{B}, d2 = \frac{:\neg A}{C}, d3 = \frac{:\neg B \wedge \neg F}{G}, d4 = \frac{:\neg B \wedge \neg C}{E} \right\}$ has:

- One classical extension: $E1 = Th(\{F \vee C, B, C\})$ with $D1 = \{d1, d2\}$ as a generating default set.
- Three justified extensions:
  - $(E1, J1)$=$(Th(\{F \vee C, B, C\}), \{A, \neg A\})$ with $D1$ as a generating default set;
  - $(E2, J2)$=$(Th(\{F \vee C, G, E\})$,$\{\neg B \wedge \neg C, \neg B \wedge \neg F\})$ with $D2$=$\{d3, d4\}$ as a generating default set;
  - $(E3, J3)$=$(Th(\{F \vee C, C, G\}), \{\neg A, \neg B \wedge \neg F\})$ with $D3 = \{d2, d3\}$ as a generating default set.
- Three constrained extensions:
  - $(E4, C4)$ =$(Th(\{F \vee C, B\}), Th(\{F \vee C, B, A\}))$ with $D4 = \{d1\}$ as a generating default set;
  - $(E5, C5)$ =$(Th(\{F \vee C, C, G\})$,$Th(\{F \vee C, G, \neg A, \neg B \wedge \neg F\}))$ with $D5 = \{d2, d3\}$ as a generating default set;
  - $(E6, C6)$ =$(Th(\{F \vee C, E\}), Th(\{F \vee C, E, \neg B \wedge \neg C\}))$with $D6 = \{d4\}$ as a generating default set.
- Two rational extensions: $(E4, C4)$ and $(E5, C5)$.

3. A heuristic approach of the extension computation problem

In this section we extend the heuristic approach of the classical extension problem from [8] to all types of default extensions: justified, constrained, rational. The theorems from the previous section show that the problem of finding extensions can be reduced to the problem of finding the generating default sets for those extensions.

In this heuristic approach we need to define a search space for generating default sets and an evaluation function to compute the fitness of each element of this space according to the definitions of different types of default extensions.

For a default theory *(D, W)* we define the search space as the set $CGD = 2^D$, representing all possible configurations, called *candidate generating default sets*.

Definition 3. For a default theory *(D, W)* and $X \in CGD$ we define:
- *candidate extension* associated to X: $CE(X) = Th(W \cup Conseq(X))$;
- *candidate context* associated to X: $CC(X) = Th(W \cup Conseq(X) \cup Justif(X))$;
- *candidate support set* associated to X: $CJ(X) = Justif(X)$.

For defining the evaluation function we need four intermediate functions: $f_0^{type}, f_1^{type}, f_2^{type}, f_3^{type}$, where *type=clas* for *clas*sical extensions, *type=just* for *just*ified extensions, *type=cons* for *cons*trained extensions and *type=rat* for *rat*ional extensions.

f_0^{type} rates if the candidate extension / candidate context is consistent or not as follows:

$$f_0^{clas}(X), f_0^{just}(X) = \begin{cases} 0 & \text{if } CE(X) \text{ is consistent} \\ 1 & \text{otherwise} \end{cases}$$

$$f_0^{cons}(X), f_0^{rat}(X) = \begin{cases} 0 & \text{if } CC(X) \text{ is consistent} \\ 1 & \text{otherwise} \end{cases}$$

f_1^{type} rates the correctness of the candidate generating default set according to the definitions of different types of default extensions:

$f_1^{type}(X) = \sum_{i=1}^{n} \pi(d_i)$, where $D = \{d_1, d_2, ..., d_n\}$

The next table defines $\pi(d_i)$ - a penalty for each default of D, where k>0.

case	$d_i \in X$	$CE(X) \vdash \alpha_i$	*Cond-justif*^{type}	$\pi(d_i)$	$d_i = \frac{\alpha_i : \beta_i}{\gamma_i}$
1	true	true	true	0	d_i correctly applied
2	true	true	false	k	d_i wrongly applied
3	true	false	true	k	d_i wrongly applied
4	true	false	false	k	d_i wrongly applied
5	false	true	true	k	d_i wrongly not applied
6	false	true	false	0	d_i correctly not applied
7	false	false	true	0	d_i correctly not applied
8	false	false	false	0	d_i correctly not applied

The condition $Cond - justif^{type}$ represents the consistency condition for the justifications of defaults according to the Definition 2 of the generating default sets for different types of extensions.

- $Cond - justif^{clas} : CE(X) \cup \{\beta_i\}$ is consistent;
- $Cond - justif^{just} : \forall \eta \in CJ(X) \cup \beta_i : CE(X) \cup \{\eta, \gamma_i\}$ is consistent;
- $Cond - justif^{cons} : CC(X) \cup \{\beta_i, \gamma_i\}$ is consistent;
- $Cond - justif^{rat} : CC(X) \cup \{\beta_i\}$ is consistent.

f_2^{type} rates the level of groundness of the candidate generating default set as follows: $f_2^{type}(X) = card(Y)$, where Y is the biggest grounded set $Y \subseteq CGD$.

f_3^{type} checks the groundness property of X:

$$f_3^{type}(X) = \begin{cases} 0 & \text{if } X \text{ is grounded} \\ 1 & \text{otherwise} \end{cases}$$

Definition 4. For a default theory *(D,W)* the evaluation function for a candidate generating default set $X \in CGD$ of an extension of $type \in \{clas, just, cons, rat\}$ is defined by:

$eval^{type} : CGD \longmapsto Z \cup \{\perp, \top\}$
if $f_0^{type}(X) = 1$
  then $eval^{type}(X) = \top$
  else if $f_1^{type}(X) = 0$ and $f_3^{type}(X) = 0$
    then $eval^{type}(X) = \perp$
    else $eval^{type}(X) = f_1^{type}(X) - f_2^{type}(X)$
  endif
endif

The following theorem provides a necessary and sufficient condition for a set of defaults to be a generating set for an extension of $type \in \{clas, just, cons, rat\}$ using the evaluation function $eval^{type}$.

Theorem 6. *Let (D,W) be a default theory. A candidate generating default set $X \in CGD$ generates an extension of $type \in \{clas, just, cons, rat\}$ if and only if $eval^{type}(X) = \perp$.*

Proof of "$\Rightarrow$":

Let (W,D) be a default theory, D' a generating default set for an extension of $type \in \{clas, just, cons, rat\}$ and suppose that $eval^{type}(D') \neq \perp$. We have two cases a) $eval^{type}(D') = \top$ **or** b) $eval^{type}(D') \in Z$.

a) If $eval^{type}(D') = \top$ then according to Definition 4, $f_0^{type}(D') = 1$, which means that:

- $CE(D') = Th(W \cup Conseq(D'))$ is inconsistent for classical and justified default logics.

  But from Theorem 1 and Theorem 2, with D' as a generating default

set, $W \cup Conseq(D')$ is consistent as a subset of consistent sets.
Thus we have that the deductive closure of a consistent set is inconsistent, which is a contradiction.
Therefore for a generating default set of a classical or justified extension we can not have $eval^{type}(D') = \top$.

- $CC(D') = Th(W \cup Conseq(D') \cup Justif(D'))$ is inconsistent for constrained and rational default logics.
  But from Theorem 3 and Theorem 4, with D' as a generating default set, $W \cup Conseq(D') \cup Justif(D')$ is consistent.
  Thus we have that the deductive closure of a consistent set is inconsistent, which is a contradiction.
  Therefore for a generating default set of a constrained or rational extension we can not have $eval^{type}(D') = \top$.

b) If $eval^{type}(D') \in Z$, then according to Definition 4, we have $f_1^{type}(D') \neq 0$ or $f_3^{type}(D') \neq 0$

- $f_3^{type}(D') \neq 0$ means that D' is not grounded in W, which contradicts the fact that a generating default set for every type of extension is grounded in W.
- $f_1^{type}(D') \neq 0$ implies that $\exists d \in D$ such that $\pi(d) \neq 0$.
  - According to the definition of $\pi(d)$ there is only the case 5: $\exists d \in D - D'$ such that $\pi(d) \neq 0$, meaning that the default d is wrongly not applied. This contradicts the maximality of the generating default set D' for all types of extensions, from the theorems 1,2,3,4.
  - There are 3 cases for which $\exists d \in D'$ such that $\pi(d) \neq 0$, with the meaning that the default d is wrongly applied.
    i) case 3 and case 4 from definition of $\pi(d)$ imply that the condition for the prerequisite: $CE(D') \vdash \alpha$ is false, which contradicts the property of groundness for a generating default set.
    ii) case 2 from the same definition of penalty implies that the consistency condition for the justification of the default d is false, which contradicts Definition 1 of the generating default sets for all types of extensions.

Therefore for a generating default set of any $type \in \{clas, just, cons, rat\}$ of extension we can not have $eval^{type}(D') \in Z$.

From a) and b) we can conclude that $eval^{type}(D') = \perp$, where D' is a generating default set of any $type \in \{clas, just, cons, rat\}$ of extension.

Proof of "$\Leftarrow$": Suppose that $X \in CGD$ is a candidate generating default set of $type \in \{clas, just, cons, rat\}$ of extension and $eval^{type}(X) = \perp$. From Definition 4 and $eval^{type}(X) = \perp$ we have that $f_0^{type}(X) = 0$, $f_1^{type}(X) = 0$, $f_3^{type}(X) = 0$.

- $f_0^{type}(X) = 0$ means that $CE(X) = Th(W \cup Conseq(X))$ is consistent for classical and justified extensions and $CC(X) = Th(W \cup Conseq(X) \cup$

$Justif(X))$ is consistent for constrained and rational extension. From here we have that $W \cup Conseq(X)$ is consistent, respectivelly $W \cup Conseq(X) \cup Jusif(X)$ is consistent.

- $f_3^{type}(X) = 0$ means that X is grounded in the set of facts W.
- $f_1^{type}(X) = 0$ in cases 1,6,7,8 implies that the conditions for prerequisites and justifications for the defaults from X are satisfied according to different types of extensions, meaning that the defaults from X are generating defaults. $f_1^{type}(X) = k$ in cases 2,3,4,5 implies that all the defaults from D-X can not be generating defaults.

Now we can easy prove that the conditions from the Theorems 1,2,3 and 4 are satisfied, therefore X is a generating default set for different types of extensions.

Example - continued: we will calculate the evaluation function for different candidate generating default sets, according to different types of extensions.

- $eval^{clas}(D1) = \perp$, and thus D1 is a generating default set for E1 because:

- $CE(D1) = Th(\{F \vee C, B, C\})$ consistent implies $f_0^{clas}(D1) = 0$;
- the defaults from *D1* have no prerequisites, and thus $f_3^{clas}(D1) = 0$;
- $f_1^{clas}(D1) = \pi(d1) + \pi(d2) + \pi(d3) + \pi(d4) = 0 + 0 + 0 + 0 = 0$ according to the definition of the penalty. d1, d2 are correctly applied ($Cond - justif^{clas}$ for d1 and d2 is satisfied) and d3, d4 are correctly not applied ($Cond - justif^{clas}$ for d3 and d4 is not satisfied).

- $eval^{cons}(D1) = \top$, therefore D1 is not a generating default set for a constrained extension because $f_0^{cons}(D1) = 1$ ($CC(D1) = Th(\{F \vee C, B, C, A, \neg A\})$ is inconsistent).

- $eval^{cons}(D5) = eval^{rat}(D5) = \perp$, therefore $D5 = \{d2, d3\}$ is a generating default set for the constrained and rational extension *(E5,C5)* because:

- $CC(D5) = Th(\{F \vee C, G, \neg A, \neg B \wedge \neg F\})$ is consistent, and thus $f_0^{cons}(D5) = f_0^{rat}(D5) = 0$;
- no prerequisites for *d2* and *d3* implies $f_3^{cons}(D5) = f_3^{rat}(D5) = 0$;
- $f_1^{cons}(D5) = f_1^{rat}(D5) = 0$, there is no penalty for the defaults of *D5*: d2, d3 are correcty applied and d1, d4 are correctly not applied.

- $eval^{cons}(D6) = \perp$ but $eval^{rat}(D6) \in Z$, therefore $D6 = \{d4\}$ is a generating default set for the constrained extension *(E5,C5)* but can not be a generating default set for a rational extension as follows:

- $CC(D6) = Th(\{F \vee C, E, \neg B \wedge \neg C\})$ is consistent, and thus $f_0^{cons}(D6) = f_0^{rat}(D6) = 0$;
- no prerequisite for *d4* implies $f_3^{cons}(D6) = f_3^{rat}(D6) = 0$;
- $f_1^{cons}(D6) = 0$, there is no penalty for the defaults of *D6*: d4 is correcty applied and d1,d2,d3 are correctly not applied.
- $f_1^{cons}(D6) \neq 0$, there is a penalty for the defaults d1 and d2 because they are wrongly not applied (case 5: the conditions for prerequisites and justifications are satisfied, but d1, d2 does not belong to D6).

If we try to add one or both of these defaults to D6 to obtain a new candidate default set, the new candidate context will not be consistent, which means that even if d1, d2 can be applied, their application will give inconsistency.

4. Conclusions

Based on the results from [8], in this paper we proposed a heuristic approach of the extension computing problem for all variants of default logic: classical, justified, constrained and rational. The evaluation function is used to compute the fitness of the elements of the search space according to the definitions of the generating default sets for different types of extensions. Future works will consist in developping an automated system, based on this approach and applying the principles of genetic algorithms.

References

[1] G. Antoniou, A.P. Courtney, J. Ernst, MA. Williams, *A System for Computing Constrained Default logic Extensions*, Logics in Artificial Intelligence, JELIA'96, Lecture Notes in Artificial Intelligence, 1126, 1996, pg. 237-250.

[2] P. Cholewinski, W. Marek si M. Truszczynski, *Default reasoning system DeReS*, Proceedings of KR-96, Morgan Kaufmann, 1996, pg. 518-528.

[3] W. Lukasiewicz, *Considerations on default logic - an alternative approach*, Computational Intelligence, 4, 1988, pg. 1-16.

[4] M. Lupea, *Nonmonotonic reasoning using default logics*, Ph.D. Thesis, Babes-Bolyai University, Cluj-Napoca, 2002.

[5] M. Lupea, *DARR - A theorem prover for constrained and rational default logics*, Studia Universitas Babes-Bolyai, Informatica, XLVII, No.1, 2002, pg. 45-52.

[6] W. Marek, M. Truszczynski, *Normal form results for default logics*, Non-monotonic and Inductive logic, Springer Verlag, LNAI 659, 1993, pg. 153-174.

[7] A. Mikitiuk, M. Truszczynski, *Rational default logic and disjunctive logic programming*, A. Nerode, L.Pereira, Logic programming and non-monotonic reasoning, MIT Press, 1993, pg. 283-299.

[8] P. Nicolas, F. Saubion, I. Stephan, *Genetic algorithm for extension search in default logic*, 8-th International Workshop on Non-Monotonic Reasoning, 2000.

[9] R. Reiter, *A Logic for Default reasoning*, Artificial Intelligence 13, 1980, pg. 81-132.

[10] T.H. Schaub, *Considerations on default logics*, Ph.D. Thesis, Technischen Hochschule Darmstadt, Germany, 1992.

[11] T.H. Schaub, *The automation of reasoning with incomplete information*, Springer-Verlag, Berlin, 1997.

[12] T.H. Schaub, *XRay system: An implementation platform for local query-answering in default logics*, Applications of Uncertainty Formalisms, Lecture Notes in Computer Science, vol 1455, Springer Verlag, 1998, pg. 254-378.

[13] C. Schwind, *A tableaux-based theorem prover for a decidable subset of default logic*, Proceedings CADE, Springer Verlag, 1990.

Babeş Bolyai University, Cluj Napoca,Romania
E-mail address: lupea@cs.ubbcluj.ro