GADEL : a Genetic Algorithm to Compute Default Logic Extensions

Pascal Nicolas, Frédéric Saubion and **Igor Stéphan** [1]

Abstract. In the area of Default Logic, after many theoretical works, some operational systems are now able to deal with real world applications. However, due to the theoretical complexity of the problem, finding a default logic extension in a practical way is not yet possible in whole generality. Our work presents a new methodology to implement an automated default reasoning system based on Genetic Algorithms techniques. The aim of this paper is not to exhibit a program able to compute extensions of every kind of default theories in a minimal time, but to present a new promising approach of the problem. We provide here a formal description of the components required for a default logic extension search, based on Genetic Algorithms principles. We give also a formal result to ensure the correctness of our approach and some very interesting experimental results w.r.t. other existing systems.

1 INTRODUCTION

Default Logic has been introduced by Reiter [12] in order to formalize common sense reasoning from incomplete information, and is now recognized as one of the most appropriate framework for *non monotonic reasoning*. In this formalism, knowledge is represented by a default theory whose sets of plausible conclusions are called *extensions*. But, due to the level of theoretical complexity of default logic ($\Sigma_2^p - complete$ [6]), the computation of these extensions is a great challenge. Previous works [4, 10, 14] have already investigated this computational aspect of default logic. Even if the system DeRes [4] has very good performances on certain classes of default theories, there is no efficient system for general extension calculus. The purpose of the present work is not to exhibit a system able to compute extensions of every default theory in a minimal time. But, we show that techniques issued from *Genetic Algorithms* can be very useful in order to build an efficient default reasoning system with the ability to deal with any propositional finite default theory without restriction on formulas. Furthermore, our approach can easily be adapted to any variant of default logic. But, dealing with a semimonotonic default logic as [8] needs a lesser computational effort because we can use a greedy algorithm, and do not need to do a final checking of the build extension that can invalidate all the previous work. In this case it is not obvious that a genetic algorithm would have good performances in face of more classical approaches.

Based on the principle of natural selection, genetic algorithms [9, 7] have been quite successfully applied to combinatorial problems such as scheduling or transportation problems. The fundamental principle of this approach states that, species evolve through adaptations to a changing environment and that the gained knowledge is embedded in the structure of the population and its members, encoded in their chromosomes. If individuals are considered as potential solutions to a given problem, applying a genetic algorithm consists in generating better and better individuals w.r.t. the problem by *selecting*, *crossing* and *mutating* them. This approach seems very useful for problems with huge search spaces and for which no tractable algorithm is available, such as our problem of default theory's extension search.

The paper is organized as follows : section 2 is a short preliminary section where basic definitions and concepts related to default logic are recalled. Section 3 provides the formal description of our system *GADEL* (Genetic Algorithms for DEfault Logic) and explains how the key principles of genetic algorithms are used to build an extension. The section 4 provides the validation of our work by giving a formal correctness result and some experiments that show that our new approach is very promising.

2 DEFAULT LOGIC

First, we recall only the materials about Default Logic that is necessary to understand the rest of our paper and we invite the non familiar reader to consult [3, 2, 14].

In Default Logic knowledge is represented by means of a *default theory* (W, D) where W contains the "sure" knowledge (in this work it is a set of propositional formulas) and D is a set of *default rules* (or defaults). A *default* $\delta = \frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma}$ is an inference rule (α, γ and all β_i are propositional formulas) whose meaning is "if the *prerequisite* α is proved, and if for all $i = 1, \ldots, n$ each *justification* β_i is individually consistent (in other words if nothing proves its negation) then one concludes the *consequent* γ[2]". Given a default theory it is possible to infer a set of plausible conclusions called an *extension* and defined by Reiter as the fixpoint of a special operator. But, we prefer to recall here the equivalent following pseudoiterative characterization because it is closer to our approach of the extension computation problem.

Theorem 1 *[12] Let (W, D) be a default theory and E a formula set. We define $E_0 = W$ and for all $k \geq 0$,*

$$E_{k+1} = Th(E_k) \cup \left\{ \gamma \;\middle|\; \frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma} \in D, E_k \vdash \alpha, \text{and } E \nvdash \neg\beta_i, \forall i = 1, \ldots, n \right\}$$

Then, E is an extension of (W, D) iff $E = \bigcup_{k=0}^{\infty} E_k$.

For a set of formulas E, $Th(E)$ denotes as usual the set of logical consequences of E, and $E \vdash \phi$ has its common sense of deduction in

[1] LERIA, Université d'Angers,2 Bd Lavoisier,F-49045 Angers Cedex 01 email: Pascal.Nicolas,Frederic.Saubion,Igor.Stephan@univ-angers.fr

[2] If δ is a default rule, $pre(\delta)$, $jus(\delta)$ and $cons(\delta)$ respectively denotes the prerequisite, the set of justifications and the consequent of δ. These definitions will be also extended for sets of defaults.

classical logic. It is important to note that a default theory may have one or multiple extensions and sometimes no extension at all as we can see below.

Example 2.1 • $(W_1, D_1) = (\{a, b \vee c\}, \{\frac{a:\neg b}{d}, \frac{c:e}{e}, \frac{d:f}{g}\})$ *has a unique extension* $Th(W_1 \cup \{d, g\})$.
• $(W_2, D_2) = (\{a, b \vee c\}, \{\frac{a:\neg b}{\neg b}, \frac{a:\neg c}{\neg c}\})$ *has two extensions* $E = Th(W_2 \cup \{\neg b\})$ *and* $E' = Th(W_2 \cup \{\neg c\})$
• $(W_3, D_3) = (\{a\}, \{\frac{a:b}{\neg b}\})$ *has no extension.*

In fact, given a default theory (W, D), to compute its extension E is equivalent to find its *Generating Default Set* Δ since $E = Th(W \cup cons(\Delta))$ [13].

Definition 2.1 *Let E be an extension of a default theory (W, D)*
$GD(W, D, E) = \left\{ \frac{\alpha:\beta_1,\ldots,\beta_n}{\gamma} \in D \;\middle|\; \begin{array}{l} E \vdash \alpha \text{ and} \\ E \nvdash \neg\beta_i, \forall i = 1, \ldots, n \end{array} \right\}$
is called the Generating Default Set of E and all defaults in $GD(W, D, E)$ are said to be applied.

To end this technical part, we recall that every generating default set is *grounded.*

Definition 2.2 *[15] Given a default theory (W, D), a set of defaults $\Delta \subseteq D$ is grounded if Δ can be ordered as a sequence $(\delta_1, \ldots, \delta_n)$ satisfying the property:*
$\forall i = 1, \ldots, n, W \cup cons(\{\delta_1, \ldots, \delta_{i-1}\}) \vdash pre(\delta_i)$

3 DESCRIPTION OF THE METHOD

Genetic Algorithms [9, 7] are based on the principle of natural selection. We first consider a *population* of individuals which are represented by their *chromosomes.* Each chromosome represents a potential solution to the given problem. An evaluation process and genetic operators determine the evolution of the population in order to get better and better individuals. Considering our extension search problem, potential solutions will be called *candidate extensions* represented by chromosomes and the purpose of our algorithm is to generate a candidate which is indeed an extension (i.e. satisfying theorem 1).

We now introduce the different parts of our search mechanism which consists of the following components, as a genetic algorithm:

1. a representation of the potential solutions : in most cases, chromosomes will be strings of bits representing its *genes*,
2. a way to generate an initial population,
3. an *evaluation function*: it rates each potential solution,
4. genetic operators that define the evolution of the population : two different operators will be considered : *Crossover* allows to generate two new chromosomes (the offsprings) by crossing two chromosomes of the current population (the parents), *Mutation* arbitrarily alters one or more genes of a selected chromosome,
5. parameters : population size p_{size} and probabilities of crossover p_c and mutation p_m. We choose $p_{size} \mid \exists N, p_{size} = \frac{N(N+1)}{2}$.

3.1 Representation

A representation scheme consists of the two following elements : a chromosome language $\mathcal{G}$ defined by a chosen size and an interpretation mapping to translate chromosomes in term of possibly applied defaults, which provides the semantics of the chromosomes. In our context, for each default $\frac{\alpha:\beta_1,\ldots,\beta_n}{\gamma}$ we encode in the chromosome the prerequisite α and all justifications $\beta_1, ..., \beta_n$ conjointly. Therefore, given a set of defaults $D = \{\delta_1, \cdots, \delta_n\}$ the size of the chromosome will be $2n$ and the chromosome language $\mathcal{G}$ is the regular language $(0+1)^{2n}$ (i.e. strings of $2n$ bits). Given a chromosome $G \in \mathcal{G}$, $G|_i$ denotes the value of G at occurrence i. Occurrences of G are elements of $\{1..2n\}$. The interpretation mapping, defining the semantics of the previous chromosomes, can be formally described as :

Definition 3.1 *Given a default set D and chromosome language $\mathcal{G}$, an interpretation mapping is defined as $\phi: \mathcal{G} \times D \rightarrow \{true, false\}$ such that:*
$\forall \delta_i \in D, \phi(G, \delta_i) = \left\{ \begin{array}{l} true \text{ if } G|_{2i-1} = 1 \text{ and } G|_{2i} = 0 \\ false \text{ in other cases} \end{array} \right.$

Our chromosomes are introduced to encode candidate extensions (i.e. potential solutions to our problem). In fact, building an extension consists in finding its *Generating Default Set* (see definition 2.1). Thus, the candidate extension $CE(W, D, G)$associated to each chromosome can also be characterized by its candidate generating default set $CGD(W, D, G)$. These two sets are easily defined w.r.t. the interpretation mapping.

Definition 3.2 *Given a default theory (W, D), a chromosome $G \in \mathcal{G}$, the candidate generating default set associated to G is :*
$CGD(W, D, G) = \{\delta_i \mid \phi(G, \delta_i) = true\}$

Definition 3.3 *Given a default theory (W, D), a chromosome $G \in \mathcal{G}$, the candidate extension associated to G is :*
$CE(W, D, G) = Th(W \cup \{cons(\delta) \mid \delta \in CGD(W, D, G)\})$

Intuitively, for a default δ_i, if $G|_{2i-1} = 1$ then its prerequisite is considered to be in the candidate extension and if $G|_{2i} = 0$ no negation of its justifications is assumed to belong to the candidate extension induced by G. $CE(W, D, G)$ and $CGD(W, D, G)$ will be simply denoted $CE(G)$ and $CGD(G)$ when it is clear from the context. Remark that since we have to compute the set of logical consequences of W and of the consequents of the supposed applied defaults, a theorem prover will be needed in our system.

Example 3.1 *Let (W, D) be a default theory where : $W = \{a\}$, $D = \{\frac{a:b}{c}, \frac{a:\neg c}{\neg b}, \frac{d:e}{f}\}$. We get : $CGD(100011) = \{\frac{a:b}{c}\}$ $CE(100011) = Th(\{a, c\})$ which is really an extension but also $CGD(101011) = \{\frac{a:b}{c}, \frac{a:\neg c}{\neg b}\}$ $CE(101011) = Th(\{a, c, \neg b\})$ which is not an extension.*

3.2 Evaluation

Definition 3.4 *Given a chromosome language $\mathcal{G}$, an evaluation function is a mapping $eval: \mathcal{G} \rightarrow \mathcal{A}$, where $\mathcal{A}$ is any set such that there exists an ordering $<$ on it (to achieve the selection process).*

Here, the evaluation function is mainly based on two criteria : the notion of generating default set (definition 2.1) and the notion of grounded default set (definition 2.2). These two aspects are rated by two intermediate functions f_1 and f_2.

For a default $\delta_i = \frac{\alpha_i:\beta_i^1,\ldots,\beta_i^{k_i}}{\gamma_i}$, we defined a function π described in table 1. Given the two positions $G|_{2i-1}$ and $G|_{2i}$ associated to the default δ_i in the chromosome, the first point is to determine w.r.t. these values if this default is supposed to be involved in the construction of the candidate extension (i.e. its conclusion has to be added to the candidate extension or not). Then, we check if this application is relevant. This evalutation is strongly related to the semantics of

Case	$G\|_{2i-1}$	$G\|_{2i}$	$CE(G) \vdash \alpha_i$	$\exists j, CE(G) \vdash \neg\beta_i^j$	π
1	1	0	$true$	$false$	n
2	1	0	$true$	$true$	y
3	1	0	$false$	$true$	y
4	1	0	$false$	$false$	y
5	1	1	$true$	$false$	y
6	1	1	$true$	$true$	n
7	1	1	$false$	$true$	n
8	1	1	$false$	$false$	n
9	0	1	$true$	$false$	y
10	0	1	$true$	$true$	n
11	0	1	$false$	$true$	n
12	0	1	$false$	$false$	n
13	0	0	$true$	$false$	y
14	0	0	$true$	$true$	n
15	0	0	$false$	$true$	n
16	0	0	$false$	$false$	n

Table 1. Evaluation

the chromosome given by definition 3.1. A y in the penality column π means that a positive value is assigned to $\pi(G|_{2i-1}, G|_{2i})$. Note that only cases 1 to 4 correspond to default considered to be applied (i.e. such that $\phi(\delta_i, G) = true$). The conditions $CE(G) \vdash \alpha_i$ and $\exists j, CE(G) \vdash \neg\beta_i^j$ uses the classical notion of logical consequence $\vdash$ and will be checked by a theorem prover. The global evaluation function f_1 is then defined by

$$f_1(G) = \Sigma_{i=1}^{n} \pi(G|_{2i-1}, G|_{2i}) \; where \; n = card(D)$$

Justifications of the penalties:

- Cases 2,3,4 : the consequent γ_i is in the candidate extension (because $G|_{2i-1} = 1$ and $G|_{2i} = 0$) while the default should not have been applied (because either $CE(G) \nvdash \alpha_i$ or $\exists j, CE(G) \vdash \neg\beta_i^j$).
- Cases 5,9,13: the consequent of the default is not in $CE(G)$ while it should since the prerequisite of the default is in the extension and no negation of justifications is deducible from it.
- Other cases : even if the chromosome value does not agree with the generated candidate extension, these cases can be ignored since they do not affect the extension.

The second part of the evaluation is based on the fact that every generating default set is grounded. From the definition 2.2, we introduce a function f_2 to evaluate the "groundedness" rate of a candidate generating default set $CGD(G)$ as :

$$f_2(G) = 1 - \frac{card(\Gamma)}{card(CGD(G))}$$

where Γ is the biggest set such that $\Gamma \subseteq CGD(G)$ and Γ is grounded.

At last, we have to take into account the logical consistency of $CE(G)$ to keep the meaning of the two previous evaluations. Since if $CE(G)$ is inconsistent then the conditions $CE(G) \vdash \alpha_i$ and $\exists j, CE(G) \vdash \neg\beta_i^j$ will be always true and moreover the "groundedness" rate will be always maximal (check definition 2.2). Therefore, a third function f_0 appears in the evaluation process and is defined by

$$f_0(G) = \begin{cases} 0 & if \; CE(G) \nvdash \bot \text{(CE(G) is consistent)} \\ 1 & otherwise \end{cases}$$

Then, we can give the definition of our global evaluation function.

Definition 3.5 *$eval$ is an evaluation function of chromosomes s.t.*

$$eval: \mathcal{G} \rightarrow \mathbb{N} \times \mathbb{N} \times \mathbb{R}$$
$$eval(G) = \langle f_0(G), f_1(G), f_2(G) \rangle$$

We can see that a chromosome G such that $eval(G) = \langle 0, 0, 0 \rangle$ corresponds to a default set satisfying all properties to be the generating default set of an extension (see subsection 4.1). This feature is the basis of our ordering process described in the next subsection.

3.3 Selection

The purpose of the selection stage is, starting from an initial population P, to generate a selected population P_{sel} containing chromosomes with the best rates according to the evaluation function. Genetic operators, which define the evolution of the population, will be applied on this intermediate population to get the next population deriving from the initial P. The selection process is based on an ordering of the individuals w.r.t. their evaluation. An ordering $<$ on $\mathbb{N} \times \mathbb{N} \times \mathbb{R}$ is defined as the natural lexicographic extension $(<, <, <)$ of the usual ordering $<$ on $\mathbb{N}$ and $\mathbb{R}$. Given a population P of size p_{size}, we built an ordered population

$$P_< = (G_i)_{i>1} \; such \; that \begin{cases} \forall i, j, \; i < j \Rightarrow eval(G_i) \leq eval(G_j) \\ \forall i, j, \; i \neq j \Rightarrow G_i \neq G_j. \end{cases}$$

The first condition implies that the chromosomes are ordered w.r.t. to their evaluation and the second condition implies that two identical chromosomes are represented only once in $P_<$. Note that if two chromosomes have the same evaluation value, they are ordered arbitrarily. We choose the ranking selection to generate the selected population. Remind that the population size is such that $\exists N, p_{size} = \frac{N(N+1)}{2}$, i.e. $p_{size} = \Sigma_{k=1}^{N} k$. The selected population P_{sel}, that will be used for crossover and mutation, is a multiset of chromosomes such that each $G_i, \; i < N$ in $P_<$ occurs $N - i + 1$ times in P_{sel}. This construction is required to preserve the maximum size of the population p_{size}.

3.4 Crossover and Mutation

As mentionned before, genetic operators are now applied on the selected population P_{sel}. Crossover is performed in the following way:

- select randomly two chromosomes in P_{sel}
- generate randomly a number $r \in [0, 1]$
- if $r < p_c$ then the crossover is possible;
  - select a random position $p \in \{1, \ldots, 2n-1\}$
  - the two chromosomes $(a_1, ..., a_p, a_{p+1}, ..., a_{2n})$ and $(b_1, ..., b_p, b_{p+1}, ..., b_{2n})$ are replaced by the two new chromosomes $(a_1, ..., a_p, b_{p+1}, ..., b_{2n})$ and $(b_1, ..., b_p, a_{p+1}, ..., a_{2n})$.
- if the crossover does not occur then the two chromosomes are put back in P_{sel}.

Mutation is defined as :

- For each chromosome $G \in P_{sel}$ and for each bit b_j in G, generate a random number $r \in [0, 1]$,
- if $r < p_m$ then mutate the bit b_j (i.e. flip the bit).

The population obtained after these evolution operations becomes the current population and will be the new input of the whole process described in subsection 3.2, 3.3 and 3.4. This full process is repeated to generate successive populations and one has to define the number of populations to be explored. The best chromosome of each population w.r.t. the evaluation function represents the current best solution to the problem. To resume, the architecture of our system GADEL is shown in figure 1. Remark that the initial population is randomly generated and, due to the acute definition of our evaluation function, we

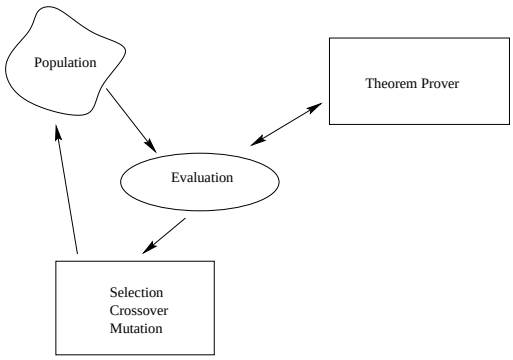

Figure 1. GADEL's architecture

get an additionnal stop criterion. This is not the case in general genetic algorithms. Here, when the evaluation of a chromosome is $\langle 0,0,0\rangle$ we can assert that its associated candidate extension is an extension of the default theory and stop the search. The validity of this criterion is proved in next section.

4 VALIDATION AND EXPERIMENTAL RESULTS

4.1 Theoretical validation

First, we give the following theoretical result that ensures the correctness of our computation methodology.

Theorem 2 *Let (W,D) be a default theory, G a chromosome and a candidate generating default set $\Delta = CGD(W,D,G)$.*
(W,D) has an extension $E = Th(W \cup cons(CGD(G)))$ if and only if $eval(G) = \langle 0,0,0\rangle$.

Proof : $\longrightarrow$: Let $E = Th(W \cup cons(\Delta))$ be an extension of (W,D). Since E is an extension, it is consistent [12], and since Δ is its generating default set E [13], it is obviously grounded. Thus, $eval(G) = \langle 0,\_,0\rangle$.

Let us suppose that $eval(G) > \langle 0,0,0\rangle$. Then, according to the definition of our evaluation function f_1 (see table 1), it means that there exists a default $\delta = \frac{\alpha:\beta_1,\ldots,\beta_n}{\gamma} \in D$ for which a penalty has been assigned. Let us examine the two possible cases:

- $\delta \in \Delta$: penalties can arise from cases 2, 3 or 4, but no one of them is possible since $E \vdash \alpha$ and $E \nvdash \beta_i, \forall i = 1,\ldots,n$ by definition of a generating default set
- $\delta \notin \Delta$: penalties can arise from cases 5, 9, or 13, but no one of them is possible since it would indicate that δ should be a generating default of E.

Thus $eval(G) = \langle 0,0,0\rangle$.
Proof : $\longleftarrow$: Let $\Delta = CGD(W,D,G)$ such that $eval(G) = \langle 0,0,0\rangle$ and $E = Th(W \cup cons(\Delta))$. Since $f_2(G) = 0$, it means that Δ is grounded. So, we can order it like $\Delta = (\delta_1,\ldots,\delta_p)$ and we have $\forall i = 1,\ldots,p, W \cup cons(\{\delta_1,\ldots,\delta_{i-1}) \vdash pre(\delta_i)$ that is equivalent to $\forall i = 1,\ldots,p, pre(\delta_i) \in Th(W \cup cons(\{\delta_1,\ldots,\delta_{i-1}))$ from which we can build the sequence

$$\begin{aligned} E_0 &= W \\ E_{i+1} &= Th(E_i) \cup \{cons(\delta_i)\}, \forall i = 0,\ldots,p-1 \end{aligned}$$

Because of the groundedness of Δ, we obtain

$$\begin{aligned} E_0 &= W \\ E_{i+1} &= Th(E_i) \cup \{cons(\delta_i) | E_i \vdash pre(\delta_i)\} \\ &\quad \forall i = 0,\ldots,p-1 \end{aligned}$$

Since $f_2(G) = 0$, we can deduce : $\forall \beta \in jus(\delta_i), E \nvdash \neg\beta$ and then

$$\begin{aligned} E_0 &= W \\ E_{i+1} &= Th(E_i) \cup \left\{ cons(\delta_i) \;\middle|\; \begin{array}{l} E_i \vdash pre(\delta_i) \\ E \nvdash \neg\beta, \forall \beta \in jus(\delta_i) \end{array} \right\} (*) \\ &\quad \forall i = 0,\ldots,p-1 \end{aligned}$$

From $f_2(G) = 0$, we can also deduce that for all other defaults $\frac{\alpha:\beta_1,\ldots,\beta_n}{\gamma} \in D \setminus \Delta$, we have either $E \nvdash \alpha$, either $\exists j, E \vdash \neg\beta_j$. So, in $(*)$ we can delete the explicit reference to i in the defaults and we can extend the sequence for all positive integer. So we have

$$\begin{aligned} E_0 &= W \\ E_{k+1} &= Th(E_k) \cup \{cons(\delta) | E_k \vdash pre(\delta), \beta \in jus(\delta), E \nvdash \beta\} \\ &\quad \forall k > 0 \end{aligned}$$

Finally, let us remark that by construction E is exactly the set $\bigcup_{k=0}^{\infty} E_k$. Thus we have obtain here the pseudoiterative characterization of an extension given in Reiter's theorem 1, and we can conclude that E is an extension of (W,D). □

4.2 Experimental results

The GADEL system is implemented in Sicstus Prolog 7.3.1. Due to the lack of space we give only few computation times in table 4.2 where problems are of two kinds. The first lines are about a taxonomic default theory "*people*" described in example 4.1. Each line corresponds to the common part of W augmented with one of the specified formula (boy or $girl$ or ...). The last lines are about the well known Hamiltonian cycle problem as it has been described and encoded in [4].

	GADEL			DeRes
problem	p_{size}	$n.p.$	time	time
boy	325	3	16	>7200
$girl$	325	3	16	>7200
man	325	5	26	>7200
$woman$	325	3	15	>7200
$man \wedge student$	1275	91	1349	>7200
$woman \wedge student$	1275	65	1202	>7200
$ham.board\_3, 2, 0, 0, 1, 0, 0\_$	465	2	4	0.56
$ham.board\_4, 2, 0, 0, 1, 0, 0\_$	1275	74	444	19.48
$ham.board\_5, 2, 0, 0, 1, 0, 0\_$	2485	-	>7200	566.45

Table 2. Experimental results

Example 4.1 $W = \{\neg boy \vee \neg girl, \neg boy \vee kid, \neg girl \vee kid, \neg human \vee male \vee female, \neg kid \vee human, \neg student \vee human, \neg adult \vee human, \neg adult \vee \neg kid, \neg adult \vee \neg male \vee man, \neg adult \vee \neg female \vee woman, \neg academic \vee adult, \neg academic \vee diploma, \neg doctor \vee academic, \neg priest \vee academic, \neg prof \vee academic, \neg bishop \vee priest, \neg cardinal \vee bishop, \neg redsuit \vee suit, \neg whitesuit \vee suit, \neg blacksuit \vee suit, \neg redsuit \vee \neg whitesuit, \neg whitesuit \vee \neg blacksuit, \neg redsuit \vee \neg blacksuit\} \cup \{boy\}or \cup \{girl\}or \cup \{man\}or \cup \{woman\}or \cup \{man, student\}or \cup \{woman, student\}$

$$D = \{\frac{human:name}{name}, \frac{kid:toys}{toys}, \frac{student:adult}{adult}, \frac{student:\neg employed}{\neg employed}, \frac{student:\neg married}{\neg married}, \frac{student:sports}{sports}, \frac{adult:\neg student}{employed}, \frac{adult:\neg student,\neg priest}{married}, \frac{adult:car}{car}, \frac{adult:\neg academic}{\neg toys}, \frac{man:\neg prof}{beer}, \frac{man:\neg vegetarian}{steak}, \frac{man:coffee}{coffee}, \frac{man \vee woman:wine}{wine}, \frac{woman:tea}{tea}, \frac{academic:\neg prof}{\neg employed}, \frac{academic:\neg priest}{toys}, \frac{academic:books}{books}, \frac{academic:glasses}{glasses}, \frac{academic:\neg priest}{late}, \frac{doctor:medicine}{medicine}, \frac{doctor:whitesuit}{whitesuit}, \frac{prof:employed}{employed}, \frac{prof:grey}{grey}, \frac{prof:tie}{tie}, \frac{prof:water}{water}, \frac{prof:conservative}{conservative}, \frac{priest:male}{male}, \frac{priest:conservative}{conservative}, \frac{priest:\neg cardinal}{blacksuit}, \frac{cardinal:redsuit}{redsuit}, \frac{car:mobile}{mobile}, \frac{tie:suit}{suit}, \frac{wine \wedge steak \wedge coffee:\neg sports}{heartdisease}, \frac{sports:man}{football \vee rugby \vee tennis}, \frac{sports:woman}{swim \vee jogging \vee tennis}, \frac{toys \wedge (football \vee rugby):ball}{ball}, \frac{toys:boy}{weapon}, \frac{toys:girl}{doll}\}$$

Column p_{size} gives the initial number of chromosomes in the population, $n.p.$ is the average number of populations needed to find an extension. The last two columns give CPU time in seconds on a SUN E3000 $(2 \times 250Mhz)$.

At this time, we have only compared GADEL with DeRes [4] because both systems accept any kind of closed default theories. Furthermore, we have focused on non stratified default theories since they are more difficult to handle. GADEL has very good performances on our taxonomic example [3] whereas DeRes does not solve it, even if we use its local prover. We can see an increase of the number of generated populations when $student$ is added to the set W (fifth and sixth lines vs above lines in table 4.2). In all cases the default theory has only one extension, but when $student$ is present the generating default contains fourteen defaults and two others are grounded. Whereas, when $student$ is not in W, the generating default set contains only five defaults and there is no other grounded defaults and that is why it is easier for our methodology to find a solution. We need also note that GADEL has poor average performances on Hamiltonian problems[4]. We think that it is because we take into account the groundedness into our evaluation function, only in a second time in the evaluation of the chromosomes that are sorted firstly acording to f_1 and secondly to f_2 (see 3.2). In the Hamiltonian problem, a solution is exactly one "chain"[5] of defaults, but, there is a lot of potential solutions (with $f_1(G) = 0$) based on two, or more, chains of defaults. The only criterion to discard these candidate generating default sets is the groundedness property that they do not satisfy $(f_2(G) > 0)$. Conversely, in people example, a solution is a set of non conflicting defaults, but at most four defaults are chained together, and so the groundedness property is less important to reach a solution. These two kinds of results illustrate the twofold difficulty in default reasoning : to respect justifications of each applied default and to find a set of defaults that are well "chained". A future improvement of GADEL is to better take into account these two aspects by defining a new global evaluation function merging f_0, f_1 and f_2 in a more efficient way, maybe in an evolutive manner during the search.

We have also in mind that in the area of logic programming and non monotonic reasoning there exist others systems (Smodels [11], DLV [5]) able to compute stable models of extended logic program. Since this task is equivalent to compute an extension of a default theory it seems interesting to compare GADEL to these systems. But, even if DLV has the advantage to accept formulas with variables which are instanciated before computation, this system does not accept theories like our people example. On its part, Smodels does not deal with this default theory because it can not be represented by a normal logic program without disjunction. Because we have the objective to deal with every kind of propositional formulas, GADEL spends a lot of time in theorem proving and it seems not realistic to compare it with those two systems. But it will be very interresting to work on GADEL's architecture in order to improve its performances on particular classes of default theories.

[3] Even if the time is not so good: all the implementation is written in Prolog, the number of generation is encouraging.

[4] The average CPU time is more than 7200 seconds for the $ham.board\_5, 2, 0, 0, 1, 0, 0\_$ problem but we get some solutions in less time.

[5] We say that δ is chained to δ' if $cons(\delta) \vdash pre(\delta')$.

5 CONCLUSION

The basic problem we wanted to address in this paper was to find an extension of a given default theory. The GADEL system we have designed shows that genetic algorithms provides an efficient framework for this particular search. As an immediate side effect, this system can be used as a complementary part of a default logic theorem prover to check if a proof scheme generated by the prover can be valid in an extension. The defaults necessary to achieve the proof would define a persistent characteristic of the chromosomes in every population (i.e. positions protected from mutation and crossover). GADEL would then be used to generate an extension w.r.t. these restrictions. Global improvement of the system could be explored in two different ways. On one hand, one can improve the performance of the genetic algorithm by introducing parallelism in the management of the population : evaluation, selection and genetic operations. On the other hand, one could introduce other heuristics in our search issued from local optimization methods [1] (simulated annealing, tabu search ...) to get an hybrid algorithm combining evolution and local search.

REFERENCES

[1] *Local Search in Combinatorial Optimization*, eds., E. Aarts and J.K. Lenstra, John Wiley and Sons, 1997.

[2] G. Antoniou, *Nonmonotonic Reasoning*, MIT Press, 1997.

[3] P. Besnard, *An Introduction to Default Logic*, Symbolic Computation — Artificial Intelligence, Springer Verlag, 1989.

[4] P. Cholewiński, V. Marek, A. Mikitiuk, and M. Truszczyński, 'Computing with default logic', *Artificial Intelligence*, **112**, 105–146, (1999).

[5] T. Eiter, N. Leone, C. Mateis, G. Pfeifer, and F. Scarcello, 'The kr system dlv:progress report, comparisons and benchmarks', in *Proceedings of the Sixth International Conference on the Principles of Knowledge Representation and Reasoning*, eds., A. G. Cohn, L. Schubert, and S. C. Shapiro, pp. 406–417. m-k, (1998).

[6] G. Gottlob, 'Complexity results for nonmonotonic logics', *Journal of Logic and Computation*, **2**(3), 397–425, (June 1992).

[7] J.H. Holland, *Adaptation in Natural and Artificial Systemes*, University of Michigan Press, 1975.

[8] W. Łukaszewicz, 'Considerations on default logic — an alternative approach', *Computational Intelligence*, **4**, 1–16, (1988).

[9] M. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer Verlag, 1996.

[10] I. Niemelä, 'Towards efficient default reasoning', in *Proceedings of the International Joint Conference on Artificial Intelligence*, ed., C. Mellish, pp. 312–318. Morgan Kaufmann Publishers, (1995).

[11] I. Niemelä and P. Simons, 'Smodels - an implementation of the stable model and well-founded semantics for normal logic programs', in *Proceedings of the 4th International Conference on Logic Programming and Nonmonotonic Reasoning*, eds., DIX J., Fuhrbach U., and Nerode A., volume 1265 of *Lecture Notes in Artificial Intelligence*, pp. 420–429. Springer Verlag, (1997).

[12] R. Reiter, 'A logic for default reasoning', *Artificial Intelligence*, **13**(1-2), 81–132, (1980).

[13] V. Risch, 'Analytic tableaux for default logics', *Journal of Applied Non-Classical Logics*, **6**(1), 71–88, (1996).

[14] T. Schaub, *The Automation of Reasoning with Incomplete Information: From semantic foundations to efficient computation*, volume 1409 of *Lecture Notes in Artificial Intelligence*, Springer Verlag, 1998.

[15] C. Schwind, 'A tableaux-based theorem prover for a decidable subset of default logic', in *Proceedings of the Conference on Automated Deduction*, ed., M. Stickel. Springer Verlag, (1990).