Super increasing sequences and learning of default rules

Mathieu Serrurier, Didier Dubois, Henri Prade

Super increasing sequences and learning of default rules

Mathieu Serrurier
UPS, IRIT,
118 route de Narbonne,
31062 Toulouse, France
serrurie@irit.fr

Didier Dubois
UPS, IRIT,
118 route de Narbonne,
31062 Toulouse, France
dubois@irit.fr

Henri Prade
UPS, IRIT,
118 route de Narbonne,
31062 Toulouse, France
prade@irit.fr

Abstract

This paper points out that the same mathematical structure, called "super-increasing sequence" appears in various disguises in different areas such as nonmonotonic reasoning (in relation with the probabilistic representation of default rules), cryptography (as an application of the knapsack problem), or uncertainty modeling (when dealing with probability/possibility transformations). Apart from bridging research trends, which are usually considered separately, this note takes advantage of this parallel for discussing different approaches to the learning of default rules.
Keywords: Super increasing sequence, default rules, learning.

1 Introduction

Default reasoning, as a way of handling if-then rules with potential exceptions, is a basic form of reasoning for dealing with commonsense knowledge. As such, it has been extensively studied in the last 25 years in AI, and applied to inference from expert rules, e.g. [8]. However, it is worth noticing that default rules can be encountered in other contexts such as algorithmic procedures, or machine learning. Indeed, this note starts from the remark that greedy algorithms are very close in spirit to default reasoning. This can be emphasized on the basis of a classical example of a greedy procedure, namely the one for giving the change for a given amount of money. The efficiency of greedy methods on this problem (which would be NP-complete in the general case) relies on a particular mathematical property of the distribution of the values of the money units (coins or banknotes) that are available. Interestingly enough, a similar mathematical structure, known as "super-increasing sequence" is at work in the knapsack crypto-systems used for key encoding, and is also underlying a finite probabilistic semantics, called big-stepped probabilities, in nonmonotonic reasoning.

The paper is organized as follows. The next section provides a short background on super-increasing sequences and its link with greedy algorithms, while Section 3 gives a brief reminder on big-stepped probabilities, linear possibilities, and their role in nonmonotonic reasoning semantics. Section 4 illustrates the parallel between the two contexts by presenting different approaches to the learning of default rules either based on the identification of a big stepped probability distribution, or on the use of possibilistic layered rules.

2 Super increasing sequences and greedy algorithms

Optimization algorithms usually go through a series of steps, where choices between different options must be made. Greedy algorithms always choose the best local option with the hope that the algorithm will lead to an optimal solution. These algorithms are very efficient, from a computational point of view, but do not find the global optimum in general. Let us consider, as an example, the standard versions of the knapsack

problem. Given an increasingly ordered sequence $X = x_1, ..., x_p$ and a value v, We can define two different knapsack problems. The *decision* one consists in determining if it exists a collection of p binary variables $\delta_1, ..., \delta_p \in \{0, 1\}$ such that

$$\sum_{i=1}^{p} \delta_i * x_i = v.$$

This problem is NP-complete. The *search* one is to determine a collection of p binary variables $\delta_1, ..., \delta_p \in \{0, 1\}$ such that

$$\sum_{i=1}^{p} \delta_i * x_i \leq v$$

and

$$\sum_{i=1}^{p} \delta_i * x_i$$

is maximal. In general, this problem is NP hard, and then greedy algorithms are not optimal for any of the two problems. It is known that with some particular sequences X, called super increasing sequences, the complexity of the decision problem becomes polynomial and even linear. These sequences are defined by :

Definition 1 *Let $X = x_1, ..., x_p$ be a sequence of real numbers. X is a super increasing sequence iff*

$$\forall 1 < i \leq p, \quad x_i > \sum_{j=1}^{i-1} x_j.$$

In this case, the greedy procedure consists, at each step, in choosing $\delta_i = 1$ for i such that the value x_i that is immediately lower than the rest of the value of v to be covered. At the end, if an i is selected twice or no x_i is lower than the rest of the value of v there is no solution. Otherwise, the greedy algorithm find the unique solution.

There are two well-known uses of super increasing sequences for the knapsack problem. The first one corresponds to the knapsack public key crypto-system [7]. The designer publishes a public key that is composed by a sequence $X = x_1, .., x_p$. A message binary $(\delta_1, ..., \delta_p)$ is encoded by $mes = x_1 * \delta_1 + ... + x_p * \delta_p$. So, knowing mes and X, decoding the message is a knapsack problem which is NP-complete. But the designer, who may receive such a message, possesses two secret keys (obeying some technical constraints) M and W such that $M * x_1, ..., M * x_p \ mod \ W$ is a super increasing sequence. Then, the designer can easily decode the message.

A second application is based on the unbound search knapsack problem. In this case, we consider an increasingly ordered sequence $X = x_1, ..., x_p$ and a value v and we search a collection of integers $n_1, ..., n_p$ such that $v = \sum_{i=1}^{p} n_i * x_i$ and $\sum_{i=1}^{p} n_i$ is minimal. The problem is NP-hard in general, be it can be polynomially resolved by the same kind of greedy algorithm than the one described previously when X is a complete super increasing sequence, whose definition is now recalled.

Definition 2 *Let $X = x_1, ..., x_p$ be a sequence of real numbers. X is a complete super increasing sequence iff $x_1 = 1$ and X is a super increasing sequence and it exists no x and i such that $X = x_1, ..., x_i, x, x_{i+1}, ..., x_p$ is still a super increasing sequence.*

The direct application of complete super increasing sequences corresponds to the problem of giving the change for a given amount of money v with a minimum number of money units. This problem is a typical unbound search knapsack problem. It is the reason why, in general, money units constitute complete super increasing sequences. Other classical examples of the use of super increasing sequences is the writing of numbers in base 2, 10,...Each step of a greedy algorithm can be represented as a set of default rules. Conversely, reasoning with sets of default rules can be viewed as a greedy procedure, since it consists in always applying the most specific applicable rule. Let us take the example of a machine providing change. Let $X = x_1, ..., x_p$ be the complete increasing sequence of values of the money units that are available and v be the change to give back.

```
while v>0
 v>xp -> give back xp and v=v-xp
 ...
 v>xi -> give back xi and v=v-xi
```

```
 ...
 v>x1 -> give back x1 and v=v-x1
end
```

Clearly the rules are presented here in a decreasing specificity ordering. If $x_1 = 1$ the algorithm terminates for any integer v. If X is a super increasing sequence, as it is the case in practice for money, the above algorithm always provides an optimal solution in terms of a minimal number of money units that are given back. For instance, with the non super increasing sequence $X = \{300, 250, 249, 1\}$ and $v = 499$, the above algorithm gives back 200 money units (one time 300 + 199 times 1), while the optimal solution only uses 2 money units (250 and 249).

3 Big-stepped probabilities and nonmonotonic reasoning

Kraus, Lehmann and Magidor [6] have proposed an approach to nonmonotonic reasoning based on postulates that a nonmonotonic consequence relation should satisfy. On the basis of these postulates, a preference entailment system has been defined (called system P). Several semantics have been proposed for this system. Among them, a probabilistic, finite semantics, and a possibility theory-based one have been provided in [3].

More precisely, it has been proved that a default rule $\alpha \rightarrow \beta$ belongs to the inference closure of system P iff for all probability measures $Prob$ belonging to a set of a so-called big-stepped probabilities we have $Prob(\beta \wedge \alpha) > Prob(\neg\beta \wedge \alpha)$(or equivalently $Prob(\beta/\alpha) > \frac{1}{2}$). Big-stepped probabilities are defined by

Definition 3 *$Prob$ is a big-stepped probability distribution on the ordered set of mutually exclusive events $\omega_1, ..., \omega_n$ iff*

$$\forall 1 \leq i < p, Prob(\omega_i) > \sum_{j=i+1}^{p} Prob(\omega_j)$$

In fact, it is allowed that the two last non-zero terms be equal, when representing a nonmonotonic consequence relation. Big-stepped are also known as atomic-bound probabilites [13]. Clearly, big-stepped probabilities corresponds to a super increasing sequence in the sense of the previous section ($X = Prob(\omega_p), ..., Prob(\omega_1)$).

Note that the following inequalities can be derived from definition 3

$$\forall i = 1, \ldots, n-2, Prob(\omega_i|A_i) > 0.5 \qquad (1)$$

with $A_i = \{\omega_i, \ldots, \omega_n\}$.

Moreover, $\Pi(\omega_{i+1}) = \sum_{j=i+1}^{p} Prob(\omega_j)$ for $1 \leq i < p$ and $\Pi(\omega_p) = Prob(\omega_p)$ defines a basic probability/possibility transformation, first introduced in [4], such that $\Pi(\omega_i) > Prob(\omega_i)$ for all i. Thus, we have the following bounds

$$\Pi(\omega_i) > Prob(\omega_i) > \Pi(\omega_{i+1}).$$

Then, the $\Pi(\omega_i)$'s form a linear possibility distribution, which is the basis of another equivalent semantics for system P, namely we have $\Pi(\alpha \wedge \beta) > \Pi(\alpha \wedge \neg\beta)$ for any default rule $\alpha \rightarrow \beta$ (with $\Pi(\alpha) = max\{\pi(\omega), \omega \models \alpha\}$).

Thus, a set of default rules can be equivalently viewed as :

i) associated with big stepped probability distributions, which provide a statistical flavor to them;

ii) associated with a layered possibilistic [2] logic base, which may be assimilated to a greedy reasoning procedure, viewing the clauses as rules.

This provides two directions for learning or mining default rules

4 Learning

Let us now consider the problem of finding of set of default rules that cover a finite number of examples.

We first recall a data mining approach based on the building of big stepped probabilities from the data. In this context, we can associate a default rule with the probability corresponding

to the proportion of examples covered by the rule. A default set of rules is optimal if the rules cover all the examples and they are underlain by big stepped probability distributions. In this case, these rules are genuine default rules in the sense of system P. This is the base for the data mining algorithm described in [1]. We consider a set of examples $E = \{w_1, ..., w_n\}$ where each $w_i =< l_1, \ldots, l_q >$ is a binary vector corresponding to q different attributes under consideration. First, the algorithm determines a partition of $E = E_1 \cup \ldots \cup E_p$ such that $|E_1|, \ldots, |E_P|$ is a super increasing sequence (where $||$ means cardinality). The partition must be minimal, i.e. any subset of E cannot be split while keeping the super increasing properties. A partial order on partitions is defined in order to express preference for partitions that group examples having a maximal number of identically valued attributes. When a partition has been discovered, a set of default rules can be identified. For identifying rules, the subsets of E are considered from E_p to E_1. The "if then" rule associated with the subset E_i is such that the attributes involved in its antecedent are discriminant with respect to the examples in E_j, $j > i$, while the attributes appearing in conclusion are those which are identically valued in E_i. By using a super increasing sequence (which corresponds to a big stepped probability when reasoning with proportion of examples), the set of default rules found is granted to satisfy the requirements of the system P. It means in practice that the new default that could be deduced from the found default rules, using system P would still agree with the database it start from.

The use of super increasing sequences would be also worth investigating in the scope of machine learning. In particular, it could be interesting in relation with the recently developed algorithms [11, 10] based on possibilistic logic in the framework of inductive logic programming (ILP) [9]. However in the ILP settings, all the rules forming the hypotheses to be induced are supposed to be Horn clauses with the same predicate in their head. It is still interesting to layer the rules in order to handle inconsistency, even if it correspond to a very special type of set of default rules.

4.1 Learning from a set examples to be stratified

The authors of the present paper have proposed an algorithm for learning from a possibilistic database [11]. A possibilistic ILP database is composed of two sets : B_p and E_p. B_p is named background knowledge and contains ground facts and formulas associated with necessity degrees. E_p is the set of examples that describes the concept to be learnt. It contains ground facts pertaining to the same predicates (the one that appears in the head of the rules to be induced), and are also associated with necessity levels. Theses necessity levels are supposed to reflect the importance of the example with respect to is nature, namely it is more important to cover typical examples rather than exceptional ones. The goal is then to find a possibilistic hypothesis H_p such that

$$B_p \cup H_p \models_p E_p,$$

where $\models_p$ is the possibilistic entailment. Then, the necessity levels on the examples correspond to priority levels for covering them. The examples that have the highest level must be covered in priority. On the contrary, the examples with a low level of necessity are not covered in priority and the rules may make error on them. The necessity levels make a partition of E (and B in the same way). Namely we have $E = E_{\alpha_i} \cup \ldots \cup E_{\alpha_p}$ where $\alpha_1 < \ldots < \alpha_p$ and E_{α_p} contains all examples that have α_i as necessity level. The method is based on a classical ILP algorithm that learns hypotheses from increasing sets of examples. It starts with E_{α_p} and B_{α_p} for computing H_p, then it computes H_{p-1} from $E_{\alpha_p} \cup E_{\alpha_{p-1}}$ and $B_{\alpha_p} \cup B_{\alpha_{p-1}}$ and so on. The hypotheses found are merged and necessity levels are associated with rules with respect to the levels of their exceptions. This allows the algorithm to learn both general rules with some exceptions with respect to the less prioritized examples, and more specific rules without exceptions. The major limitation of this method is that we have to have assigned the necessity levels of the examples. They may be obtained from a set Δ of expert default rules (if available), by computing the level of inconsistency of each example with the

set of stratified rules associated with Δ using the ranking procedure in [2]. Then, the priority of the example would be all the greater as the level of inconsistency is low.

In case no expert knowledge is available, as often, it would be interesting to assess the necessity levels of the examples by determining a partition associated with a super increasing sequence. This would ensure, with a correct and complete ILP algorithm, to find an hypothesis made of genuine default rules. In this scope, the ideal partition of the set of examples is the one that describes a partition that corresponds to a super increasing sequence, with respect to the number of examples in each set of the partition. Moreover, the idea is also to build a partition $E = E_1 \cup \ldots \cup E_p$ of the examples, where the example in E_1 are as similar as possible, and then the examples in $E_1 \cup E_2$ remain similar as much as possible, and so on for all $\bigcup_{j=1}^{i} E_j$. Formally, let consider $E = E_1 \cup \ldots \cup E_p$ such that $|E_p|, \ldots, |E_1|$ is a super increasing sequence (i.e. $\forall j, |E_j| \geq \sum_{i=j+1}^{p} |E_i|$ and d is a distance between examples (in the case of relational learning some distances have been proposed, see [5] for instance). Given a set of example $E_i = \{e_1, \ldots, e_n\}$, the average value of the distance between examples in E_i is

$$D(E_i) = \frac{\sum_{k=1}^{n} \sum_{j=k+1}^{n} d(e_k, e_l)}{\frac{|E_i|*(|E_i|-1)}{2}}.$$

According to the above discussion, a good partition for the sets of examples may be the super increasing one that minimizes the value

$$\sum_{i=1}^{p} |E_i| * D(\bigcup_{j=1}^{i} E_j). \qquad (2)$$

It is obvious that, without any information, finding an optimal solution is a NP-complete problem. A possible heuristics may be first to fix the shape of the super increasing sequence as the "extremal" one in the sens of equation 1, i.e, $|E_i| = \frac{[\,|E - \bigcup_{j=1}^{i-1} E_j|\,]}{2} + 1$ where $[\ ]$ denotes the integer part. Then, as long as the criterion (eq. 2) decrease, we swap pairs of examples in between two subsets of the partition (Alg 1). Remark that this approach is not optimal. Better results with respect to the criterion could be achieved by also allowing the move of an example from one subset to another subset; this will require the use of meta-heuristics such as simulated annealing or genetic algorithms.

Alg. 1 Building a partition

Require: E the set of examples
Require: d a distance between examples
1: let $E = E_1 \cup \ldots \cup E_p$ a random "extremal" sequence
2: improvement=true
3: **while** improvement **do**
4: **for all** possible pairs of examples $(e_i \in E_i, e_j \in E_j)$ **do**
5: **if** it exists a pair (e_i, e_j) that decreases the criterion when they are swapped **then**
6: choose the pair (e_i, e_j) that minimizes the criterion
7: swap e_i and e_j
8: **else**
9: improvement=false
10: **end if**
11: **end for**
12: **end while**

4.2 Learning stratified set of rules from classical examples

In [10], possibilistic logic is used for dealing with exceptions, assimilated as inconsistency, in a classical ILP setting. Due to the monotonicity of first order logic consequence relation, hypotheses in ILP cannot deal with exceptions (misclassification). In particular, classical hypotheses accumulate all exceptions of the rules that appear in them and if a rule has an exception, there is no way to compensate by with another rule. In [10], We have proposed to use possibilistic logic in order to manage the exceptions. Thus, given a classical background knowledge B and a set of examples E, a stratified set of rules H_p is to be induced such that it exists a stratification of E, named E_p, such that

$$B \cup H_p \models_p E_p.$$

By contrast with the previously described approach, the stratification of the examples is not given and depends on the hypothesis. The stratification of rules allows the algorithm to deal with multiple classification of examples

and to compensate the errors of a rule at a given level by rules at higher levels. The possibilistic hypothesis are learned directly by adapting an ILP algorithm based on simulated annealing [12]. The stratification is induced in maximizing the accuracy and does not take into consideration an intended default rule meaning. In order to do that, a measure could be introduced for guiding the learning process in order to induce hypotheses that describe super increasing sequences of examples.

Given a stratified hypothesis $H_p = h_1, ..., h_n$ (h_1 is the most prioritized rule), we note $supp(h)$ the proportion of examples to which the rule apply (even it leads to wrong classification). We call effective support of a rule the proportion of examples to which the rule effectivelly apply when considering the priorities in possibilistic logic. More formally, the effective support is then $suppEff(h_i) = supp(h_i) - suppEff(h_{i-1})$ with $suppEff(h_1) = supp(h_1)$. Thus, a stratified hypothesis will be considered as acceptable from the non monotonic reasoning point of view, if the sequence $\{suppEff(h_1), \ldots, suppEff(h_n)\}$ is a super increasing sequence. In order to guide the algorithm toward stratified hypothesis that describes a super increasing sequence and that have an high accuracy, we need to measure the distance between the sequence of the effective supports and the closest super increasing sequence (as define in algorithm 2). Then the function to maximize is

$$val(H_p) = acc(H_p) - c * distIS(H_p)$$

where $acc(H_p)$ is the accuracy of H_p and c is a constant. The constant c is used for the control of the relative weight of accuracy vs. the distance of the hypothesis to the closest super increasing sequence. We consider that the most general hypothesis H_g (i.e. the hypothesis that classes all examples in the majority class) is as acceptable as the most specific hypothesis H_s (i.e. E). Thus $c = \frac{acc(H_s)*distIS(H_g) - acc(H_g)*distIS(H_s)}{distIS(H_s) - distIS(H_g)}$. Since $distIS(H_g) = 0$, we can take c such as $c = acc(H_g)$.

Alg. 2 distIS

Require: $H_p = h_1, ..., h_n$ a stratified hypothesis

$cumulsupport = 1$
$diff = 0$
$\epsilon = \frac{1}{|E|}$
$i = n$
while $cumulsupport > 0$ **do**
  if $i > 0$ **then**
    $currentsupport = suppEff(h_i)$
  else
    $currentsupport = 0$
  end if
  if $currentsupport \leq (1 - cumulsupport)/2$ **then**
    $diff = diff + (1 - cumulsupport)/2 - currentsupport + \epsilon$
    $currentsupport = (1 - cumulsupport)/2 + \epsilon$
  end if
  $cumulsupport = cumulsupport - currentsupport$
  $i = i - 1$
end while
return $diff$

5 Conclusion

In this paper, we have outlined the links that exists between non-monotonic reasoning and greedy algorithms by pointing out the similarity of the two mathematical structures underlying them : super increasing sequences and big-stepped probabilities. This link echoes the two-sided nature of the representation of default rules either in statistical terms or as properly layered sets of classical formulas. It suggests that learning a minimal default rules coverage of a set examples amounts to identify hypotheses that induce super increasing sequences of subsets of examples.

References

[1] S. Benferhat, D. Dubois, S. Lagrue, and D. Prade. A big-stepped probability approach for discovering default rules. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 11, suppl.:1–14, septembre 2003.

[2] S. Benferhat, D. Dubois, and H. Prade. Representing default rules in possibilistic logic. In *Proceedings of the 11th International Conference of on Principles of Knowledge Representation and Reasoning KR92*, pages 673–684, 1992.

[3] S. Benferhat, D. Dubois, and H. Prade. Possibilistic and standard probabilistic semantics of conditional knowledge bases. *Journal of Logic and Computation*, 9(6):873–895, 1999.

[4] D. Dubois and H. Prade. On several representations of an uncertain body of evidence. In M.M. Gupta and E. Sanchez, editors, *Fuzzy Information and Decision Processes*, pages 167–181, 1982.

[5] W. Emde and D. Wettschereck. Relational instance-based learning. In L. Saitta, editor, *Proceedings of the 13th International Conference on Machine Learning*, pages 122–130. Morgan Kaufmann, 1996.

[6] S. Kraus, D. Lehmann, and M. Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *AI*, 44:167 – 207, 1990.

[7] R. C. Merkle and M. E. Hellman. Hiding information and signatures in trapdoor knapsacks. *IEEE Transactions on Information Theory,*, 24:525–530, 1978.

[8] L. Morgenstern and S. Moninder. An expert system using nonmonotonic techniques for benefits inquiry in the insurance industry. In *IJCAI (1)*, pages 655–661, 1997.

[9] S.-H. Nienhuys-Cheng and R. de Wolf. *Foundations of Inductive Logic Programming*. Number 1228 in LNAI. Springer, 1997.

[10] M. Serrurier and H. Prade. Coping with exceptions in multiclass ILP problems using possilistic logic. In *Proc of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 1761–1764, 2005.

[11] M. Serrurier and H. Prade. Possibilistic inductive logic programming. In L. Godo, editor, *Proceedings of European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'05) - LNAI 3571, Barcelone*, pages 675–686, Berlin Heidelberg, 2005. Springer-Verlag.

[12] M. Serrurier, H. Prade, and G. Richard. A simulated annealing framework for ILP. In R. Camacho, R. King, and A. Srinivasan, editors, *Proceedings of ILP 2004*, pages 288–304. LNAI 3194, Springer, 2004.

[13] P. Snow. Diverse confidence levels in a probabilistic semantics for conditional logics. *Artif. Intell.*, 113(1-2):269–279, 1999.