Introspection, Updates and Belief Revision as Agent Processes

Fernando Zacarias Flores[1], Rosalba Cuapa Canto[2] Erick Madrid[3]
1,3 Computer Science – Autonomous University of Puebla,
2 Architecture Faculty – Autonomous University of Puebla

ABSTRACT:
A novel way of building intelligent agents is deployed. This proposal considers two aspects: In the first instance, our agents have a novel mechanism for updates, simple and easy of calculating and that guarantee our agents to stay always consistent. This mechanism was introduced and formalized in [13] and it is supported by the paradigm Answer Set Programming (ASP). ASP programs are written in the language of AnsProlog and its extensions [5, 11, 12, 14]. And, in a second plane, we generalize our kind of programs accepted in [13] and we accept general clauses, disjunctive clauses and augmented clauses. Considering these two aspects, we can develop systems to approach in a direct way to intelligent behavior.

KEYWORDS: *Agents, ASP, Belief revision, Introspection, Updates, AnsProlog, Consistent.*

I. INTRODUCTION

The agent paradigm has recently increased its influence in the research and development of computational logic-based systems. A clear and correct specification is made through Logic Programming (LP) and Non-nomotonic reasoning that has been brought (back) to the spot-light. Also, the recent significant improvements in the efficiency of ***LP*** implementations for Non-monotonic Reasoning [3] have helped to this resurgence. However, when we develop a real application, we need a friendly front-end for user. For this reason, we integrated both Answer Set Programming (ASP - defined in [5]) and Java (object-oriented programming language), eliminating the traditional high gap between theory and practice. We make use of several important results [12, 13, and 15] that we have obtained in the last years in the field of non-monotonic extensions to Answer Set Programming. This can represent an important added value to the design of rational agents. Let us remember that ASP is the realization of much theoretical work on Non-monotonic Reasoning and Artificial Intelligence (AI) applications of *LP* for the last 15 years. The two most known systems that compute answer sets are DLV[1]and SMODELS[2].As we mention previously, our proposal is based in integrating to our agents a novel mechanism (given in section 3, definition 2) for updates. We want our agent to stay consistent in all moment, so that it acts in a correct and opportune way. We introduced and formalize this mechanism in [14], it is supported by ASP and its extensions [5, 12, 13, and 15]. In this context in our proposal we make use of two principles pointed out by Daniel Kahneman, *Nobel economy reward 2002*. First, people expected samples to be highly similar to their parent population and also to represent the randomness of the sampling process [16, 17], and second, people often rely on representativeness as a heuristic for judgment and prediction [8, 9].

Kahneman is a pioneer on the integration of economy and psychology research about making decisions. His job has opened a new line of studies, discovering how the human judgment sometimes takes short cuts and amazing paths which are very different from probability basic principles and theories about complex reasoning. All this opens a new line of studies on the subject of reasoning in logic, where most of the times we want to develop reasoning and complex mechanisms. However, Kahneman's studies show the opposite. The taking of decisions escapes many times from probabilities, economy predictions and from reasoning. For this reason, in our proposal we incorporate to our rational agents an update process with a human behavior through our definition introduced in [14]. In the same way as in [14], given a theory T, its knowledge is understood as the formulas F such that F is derived in T using intuitionistic logic. This makes sense, since in intuitionistic logic according to Brouwer, *A* is identified with “I know *A*". We consider that an agent whose knowledge base is a theory T believes F if and only if F belongs to every intuitionistically complete and consistent extension of T by adding only negated literals. Take for instance: $\neg a \to b$. The agent knows $\neg a \to b$, $\neg b \to \neg\neg a$ and so on. The agent does not know however “a”. Nevertheless, one believes more than one knows.

[1] http://www.dbai.tuwien.ac.at/proj/dlv/
[2] http://saturn.hut.fi/pub/smodels/

But a cautious agent must have its beliefs consistent to its knowledge. This agent will then assume negated literals to be able to infer more information. Thus, in our example, our agent will believe ¬a, and so it can conclude b. It also makes sense that a cautious agent will believe ¬a or ¬¬a rather than to believe "a" (recall that a is not equivalent to ¬¬a in intuitionistic logic). This view seems to agree with a point of view stated by Kowalski, namely ``that Logic and Logic Programming need to be put into place: Logic within the thinking component of the observation-thought-action cycle of a single agent, and Logic Programming within the belief component of thought" [7]. See [12, 14, and 15] for a complete discussion on the role of intuitionistic logic in our approach, our proposal is focused to a real application.

On the other hand, our update process is safe and it maintains their knowledge base consistent, all this with the purpose that our agent gives reliable answer and in the right time. Later on, in a transparent way for the user, our agent carries out an introspection process. This process allows to the agent to refine their knowledge base, eliminating possible redundancies and restoring those independent beliefs to the new acquired knowledge. We use this introspect process supported by Kahneman's ideas. However, the question now arises is whether the result of an update process will depend on the particular set of sentences in the knowledge base, or only on the worlds described by this. We are interested in proposals that satisfy the fundamental principle of Irrelevance proposed by Dalal of syntax, that is, the meaning of the knowledge that results from an update must be independent of the syntax of the original knowledge, as well as independent of the syntax of the update itself. However, in our implementation we propose to reconsider the AGM postulates [1] under our new interpretation that considers "knowledge" and "belief". We use a new postulate which we call "Weak Irrelevance of Syntax" (WIS) and that we defined in [14]. This postulate, suggested by several authors [1, 4, 6], is satisfied by our update operator, as desired.The remainder of the paper is structured as follows: In section 2 we briefly recap the basic background used throughout the paper. In section 3, we present our new proposal on modeling of agents. Next, we present our application based in rational agents presented in section 4. Finally, in section 5, we give our conclusions and future work.

II. BACKGROUND

In this section, we give some general definitions for our theory. We define our theory about logic programs.

2.1. Preliminary

Rules are built from propositional atoms and the 0-place connectives T and ⊥ using negation as failure (¬) and conjunction (,). A rule is an expression of the form:

$$\text{Head} \leftarrow \text{Body} \qquad (1)$$

If Body is T then we identify rule (1) with rule Head. If a Head is ⊥ then we identify rule (1) with a restriction. A program is a set of rules. A logic program P is a (possibly infinite) set of rules. For a program *P*, *I* is a model of *P*, denoted $I \models P$, if $I \models L$ for all $L \in P$. As it is shown in [2], the Gelfond-Lifschitz transformation for a program *P* and a model $N \subseteq B_P$ (B_P denotes a set of atoms that appear in *P*) is defined by

$P^N = \{rule^N : rule \in P\}$ where $(A \leftarrow B_1, \dots, B_m, \neg C_1, \dots, \neg C_n)^N$ is either:

a) $A \leftarrow B_1, \dots, B_m$, if $\forall j \leq n: Cj \notin N$;

b) T, otherwise

Note that P^N is always a definite program. We can therefore compute its least Herbrand model (denoted as M_{PN}) and check whether it coincides with the model N which we started with:

Definition 1. [2] N is a stable model of P iff N is the minimal model of P^N

To avoid technical problems we omit a formal definition of ASP for arbitrary formulas with two kinds of negations and all the theoretical concepts of answer sets see [12, 13, and 15].

2.2. Extending our kind of programs

We use several kinds of clauses found in literature [12]. A free clause is built from a disjunction of literals in the head and a conjunction of literals in the body. Such that a clause has the form $h_1 \vee \dots \vee h_n \leftarrow b_1, \dots, b_m$. where each h_i and b_j is a literal. Either the head or the body of a free clause could be empty to denote a constraint or a fact. A general clause is a free clause that does not allow negation in the head, all literals in the head of the clause should be positive atoms. Finally, a disjunctive clause is a general clause with a non-empty head, i.e. it is not a constraint. Next, we give an example using a disjunctive program in the context of our application; it consists on a research environment for our scientific community.

Example 1. In our system is common to establish an appointment, let us see the following example:
$\leftarrow$ appointment(X), schedule-available(X),spot(X).
spot(a).
spot(b).
~schedule-available(X) $\leftarrow$ appointment(X), spot(X).
schedule-available(a) v schedule-available(b).

The interpretation is as follows: the first rule says that it is not possible to have an appointment, a spot and a schedule-available simultaneously. The third rule says that you don't have scheduled available X if you have an appointment X and a spot X. The last rule represents our schedule available, in this case, schedule-available(a) or schedule-available(b). As we can see, applying our definition given in the section 3 to this example we obtain that the program has two models: {schedule-available(a), spot(a), spot(b)} or {schedule-available(b), spot(a), spot(b)}, as desired. If our agent receives the following information: appointment(a) then, applying our definition 2 we obtain one model {schedule-available(b), spot(a), spot(b), appointment(a)}.

If later on we update with appointment(b), applying our definition again we obtain:
$\leftarrow$ appointment(X), schedule-available(X), spot(X).
spot(a)$\leftarrow$ ¬~spot(a). spot(b)$\leftarrow$¬~spot(b).
~schedule-available(X) $\leftarrow$ appointment(X), spot(X), ¬schedule-available(X).
schedule-available(a) v schedule-available(b)$\leftarrow$¬~schedule-available(b).
schedule-available(a) v schedule-available(b) ¬~schedule-available(a).
appointment(a) $\leftarrow$ ¬~appointment(a).
appointment(b).
Whose answer set is: {appointment(a), appointment(b), spot(a), spot(b), ~schedule-available(a), ~schedule-available(b)} as desired. This means that the rule five disappear, i.e., this rule doesn't have effect.

III. MODELING OF INTELLIGENT AGENTS

In this section, we present how our agents act in a dynamic environment. Before such situation, our agents should act in a correct and sure way, giving answers in an opportune way. This process is known as update process. We want this process to allow our agents to stay consistent in all moment. This guarantees us that our agents can always act in a reliable way.

3.1. Update definition

As part of our agents, we give our definition about update process. This definition was introduced in [14] and satisfies several properties of AGM postulates [1]. This gives to our agents an added value with respect to other proposals that don't satisfy them. One of the main aims of logic-constrained revision is to characterize suitable update operators through postulates like those formulated by AGM. In [4], the authors recapture these postulates and give their interpretation about AGM postulates in the update context. However, no such set of postulates would be adequate for every application [6]. Next, we present our update definition introduced in [14] and at once we present our new form of modeling agents. This form includes a novel mechanism that consists of the following three processes: Expansion, Update and Introspection.

Definition 2. Giving an update of two programs $P_\otimes = (P_1, P_2)$ over a set of atoms A, we define the update program $P_\otimes = P_1 \otimes P_2$ over A* consisting of the following items:

(i) all constraints in $P_1 \cup P_2$;
(ii) for each $r \in P_1$; $L \leftarrow B(r), \neg{\sim}L.$ if $H(r) = L$;
(iii)all rule $r \in P_2$.

As we can see, our proposal is inspired by both AGM postulates and the proposal presented in [4]. As it is shown in [4] the interpretation in belief revision and in update coincides in some of the postulates. Also, our proposal (definition 2) coincides with [4] in a wide family of programs. It is necessary to highlight the simplicity of our proposal, that allows our agent to be able to respond in a correct and opportune way, for later apply our introspection process. Our agent faces the problem of update it knowledge base with new information considering three aspects: First, if new knowledge of the world is somehow obtained, and it doesn't have conflicts with previous knowledge then this only expands knowledge (we will refer it as expansion [1]). Second, if, on the contrary, new knowledge is inconsistent with the previous knowledge, and we want knowledge to be always consistent so that our agents can act in all moment, we should solve this problem somehow. We point out that new information is incorporated into the current knowledge base subject to a causal rejection principile, which enforces that, in case of conflicts among rules, more recent rules are preferred and older rules are overridden. Third, once the agent applies our definition 2 formalized in [14] it can respond in a quick and opportune way, and our agent is in a new state. Since this moment our agent can apply its introspective process. This introspective process will allow our agent to revise the implications that our update could have generated. For instance, recapturing the example 1 we can see that is not necessary to weaken all rules, for example: spot(*a*) and spot(*b*).

3.2. Expansion

Next, we present a first example that shows the expansion process of new knowledge, i.e., if new knowledge of the world is somehow obtained, and it doesn't have conflicts with the previous knowledge then this only expands knowledge base.

Example 2. The next example shows the expansion process that our agent executes when to he/she is presented knowledge without consistency problems with previous knowledge base.

Let P be: r1: mail-revision ← computer-on, internet-on.
r2: another-task ← ¬mail-revision.
r3: computer-on.
r4: internet-on ← computer-on, ¬power-failure.
Let P_1 be: r5: class-preparation ← ¬there-are-students.
r6: research ← ¬busy.

In this case, as we can see the new knowledge doesn't have conflicts with previous knowledge. In this case, our agent only applies the expansion process. Then, the answer set of $P \otimes P_1$ is: {computer-on, mail-revision, internet-on, class-preparation, research} as desired.

3.3. Update

Next, we recapture the previous example and show our update process using our definition 2, which coincides in a wide family of programs with the one presented in [4].

Example 3. This example shows the update process. This happens if by the contrary, new knowledge is inconsistent with the previous knowledge base, and we want that our agent maintains its knowledge always consistent so that our agents can act in all moment.

Let P_1' as follows: r7: ~computer-on ← power-failure.
r8: power-failure.

Then, applying our previous definition 2 we obtain the following program:

r1: mail-revision ← computer-on, internet-on, ¬~mail-revision.
r2: another-task ← ¬ mail-revision, ¬~another-task.
r3: computer-on ← ¬~computer-on.
r4: internet-on ← computer-on, ¬power-failure, ¬~internet-on.
r7: ~computer-on ← power-failure.
r8: power-failure.

The answer set of $P \otimes P_1$' is: {~computer-on, power-failure, another-class} as desired.
As we can observe, in this example the rules r7 and r8 have problems of consistency with previous knowledge. In this sense, the most recent rules are preferred and oldest ones are overridden. This proposal gives to our agents a quick and efficient mechanism.

3.4. Update

Here, we present our proposal related with update process called “introspective process”, that it can be seen more near to belief revision. This process is executed for our agent after having applied our update process. Also, this process can be used to analyze short messages in mobile telephony. Next, we recapture the previous example and consider that our knowledge base P includes the rules from r1 to r6, and the new knowledge consists of rules r7 and r8. The first step consists in applying our update process. However, the rules r5 and r6 are independent of the rest of the program. Our agent will make an introspective analysis evaluating if it was necessary to weaken all the rules, made in the update process. This is an excellent process that our agent has and that allows itself to restore all those rules that are not involved with the new knowledge (in this case, the rules r5 and r6).

Example 4. This example shows the introspective process. This happens after applying our definition 2. Then, the answer set of P $\otimes$ P_1' is: {~computer-on, power-failure, another-class, class-preparation, research} as desired.

Our conceptual solution to modeling introspection is as follows:

- Keep a history of the last *n* updates done by our agent as well as its original theory representing its sets of beliefs.
- Verify step by step each update. Here, we have two major concerns. First, to correct only the ``relevant'' rules (recall in example 3 that we modified all rules). Second, if the belief base becomes inconsistent (a non-typical situation) we need to apply a form of belief revision in order to achieve consistency. We have partial theoretical results reported in [11, 12, 13, 14, and 18].
- If a major problem can not be resolved by (2). Our idea is that the system should request a discussion group (by e-mail) with others agents (possibly humans) such that all together can find a solution.

However, is required to continue working in this research line.

IV. Application based on intelligent agents

In this section we present our application based in rational agents. This proposal integrates both ASP and Java, eliminating the traditional high gap between formal theory and practice. We use java as front-end, in which we develop the interfaz for user and the database administration. While in the declarative part, we use answer set programming through DLV. This paradigm gives the formal support to our agents. Nowadays, the agent paradigm has recently increased its influence in the research and development of computational logic based systems. However, the development of "rational" agents is a hard task, due to this involves both the updates and beliefs revision process. Besides, when we think in rational agents, we should consider some human characteristics like: reasoning, planning, and acting in a dynamic world. Our application consists in a work environment (figure 1) that integrates necessary tools for correct performance of our scientists, incorporating a new formalization about update process. Our application consists of: an intelligent calendar that has as main ingredient the negotiation of meetings among multiple members of our scientific community via rational agents; a chat, that allows virtual meetings reducing the big distances; electronic mail and short messages via mobile telephone, used as our general communication channel in the negotiation meetings by our agents; an intelligent editor, that counts with a rational agent that assists in papers writing and a knowledge base on the scientists' belief. In the figure 1, we present the main interfaz of our application. As we can observe one of the main tasks is the presentation of the schedule of each one of the researchers in our community. All this with the following purpose: that any member of the community can request a meeting with any of the researchers. The acceptance or not of a meeting depends on the beliefs that our agent has with respect to the required researcher. These knowledge bases are formed for beliefs, knowledge and preferences. Obviously, these knowledge bases change in a dynamic way according to the researcher's dynamic behavior.

Example 5. This example shows how our agent acts when it faces new knowledge. Suppose that knowledge base is as follows:

workday(mon). workday(tue). workday(wed). workday(thu). workday(fri).
coffee-hour(1600,Day) ← workday(Day).
call-home(1930,Day) ← workday(Day). research-time(1200,1400,mon,alone).
research-time(1200,1400,wed,group). research-time(1600,1800,fri,alone).
superior(Super, Person) ← boss(Super, Person).
superior(Super, Person) ← boss(Super, Someone), superior(Someone, Person).

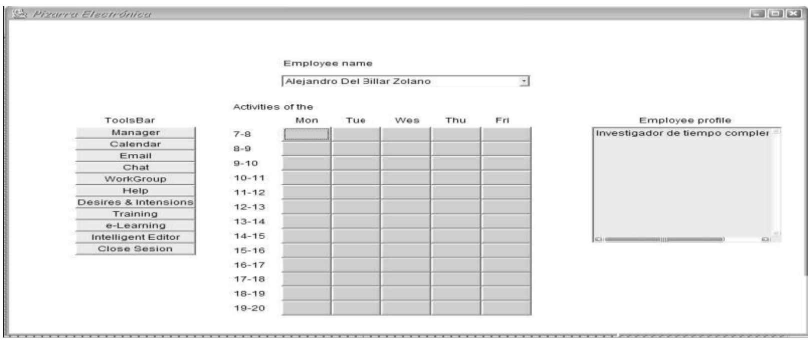

Figure 1. Electronic board

Now, suppose that our agent receives the new following information:
~coffee-hour(1600,Day) ← workday(Day). tee-hour(1600,Day) ← workday(Day).
In this case, our agent doesn't apply the expansion process, because the new information has conflicts with the previous information. Therefore, applying our definition about belief revision we obtain:

workday(mon) ← ¬~workday(mon). workday(tue) ← ¬~workday(tue).
workday(wed) ← ¬~workday(wed). workday(thu) ← ¬~workday(thu).
workday(fri) ← ¬~workday(fri).
coffee-hour(1600,Day) ← workday(Day), ¬~coffee-hour(1600,Day).
call-home(1930,Day) ← workday(Day), not ~call-home(1930,Day).
research-time(1200,1400,mon,alone) ← ¬~research-time(1200,1400,mon,alone).
research-time(1200,1400,wed,group) ← ¬~research-time(1200,1400,wed,group).
research-time(1600,1800,fri,alone) ← ¬~research-time(1600,1800,fri,alone).
superior(Super, Person) ← boss(Super, Person), ¬~superior(Super, Person).
superior(Super, Person) ← boss(Super, Someone), superior(Someone, Person),
¬~superior(Super, Person). ~coffee-hour(1600,Day) ← workday(Day).
tee-hour(1600,Day) ← workday(Day).

In this case, the answer set is:
{workday(mon), workday(tue), workday(wed), workday(thu), workday(fri), ~coffee-hour(1600,mon), ~coffee-hour(1600,tue), ~coffee-hour(1600,wed), ~coffee-hour(1600,thu), ~coffee-hour(1600,fri), call-home(1930,mon), call-home(1930,tue), call-home(1930,wed), call-home(1930,thu), call-home(1930,fri), research-time(1200,1400,mon,alone), research-time(1200,1400,wed,group), research-time(1600,1800,fri,alone), tee-hour(1600,mon), tee-hour(1600,tue), tee-hour(1600,wed), tee-hour(1600,thu), tee-hour(1600,fri)} as desired.

As we can observe, the new information is related with the coffee hour only. Then, our agent can apply the introspection process, i.e., our agent can execute its introspective process. Considering that it is totally unaware to the new knowledge. In this case, we can consider following knowledge obtaining:
research-time(1200,1400,mon,alone). research-time(1200,1400,wed,group).
research-time(1600,1800,fri,alone). superior(Super, Person) ← boss(Super, Person).
superior(Super, Person) ← boss(Super, Someone), superior(Someone, Person).
On the other hand, our agent should make a more exhaustive analysis with respect to the most relevant knowledge with the purpose of not modifying the original semantics.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we considered a new proposal for modeling agents. In this proposal we include three very important characteristics that give to our agent a human behavior. This behavior is very similar to the human behavior considering Kahneman's ideas. Humans don't always make very exhaustive reasoning, mainly in situations where we should give answers in a quick, opportune, and correct way. Also, our application is based in rational agents whom lean in an update process that is independent from syntax and whose theoretical support is presented in [14]. With respect to future work, we will continue our research in belief revision. In particular form, we will continue our analyses about introspective process. Because, we consider that this process is adapted for when one wants that our agents to give opportune answers in time and it forms. Also, another future objective is to endow our agents of new capacities that allow them to analyze the short messages among mobile telephones, with the objective of determining a reception of 100 percent of them.

REFERENCES

[1] C.E. Alchourron, P. Gardenfors, and D. Makinson. On the logic of Theory Change, Partial Meet Functions for Contraction and Revision Functions. Journal of Symbolic Logic, 50:510-530, 1985.

[2] G. Brewka, J. Dix, and K. Knonolige, nonmonotonic reasoning, an overview. CSLI Publication Eds. Leland Stanford Junior University, 1997.

[3] D. De Schreye, M. Hermenegildo and L.M. Pereira. Paving the Roadmaps: Enabling and Integration Technologics. http://www.compulog.org/net/Forum/Supportdocs.html.

[4] T. Eiter, M. Fink, G. Sabattini, and H. Thompits. Considerations on Updates of Logic Programs. In M.O. Aciego, LP. de Guzmßn, G. Brewka, and L.M. Pereira, editors, Proc. Seventh European Workshop on Logic in Artificial Intelligence JELIA 2000, vol. 1919 in Lecture Notes in Artificial Intelligence. LNAI, Springer 2000.

[5] M. Gelfond, V. Lifschitz. The stable model semantics for logic programs. Proceedings of the Fifth International Conference on Logic Programming 2, MIT Press. Cambridge, Ma. pp.1070-1080.

[6] H. Katsumo and A.O. Mendelzon. On the difference between updating a knowledge base and revising it. in: J.A. Allen, R. Fikes and E. Sandewell. eds.. Principles of knowledge representation and reasoning: Proceedings of the Second International Conference (Morgan Kautmann. San Mateo. CA. 1991) pp. 387--394.

[7] R. Kowalski. Is logic really dead or just sleeping. In Proceedings of the 17th International Conference on Logic Programming, pages 2-3, 2001.

[8] D. Kahneman and A. Tversky. Subjective probability: A judgment of representativeness, Cognitive Psychology, vol 3, pp. 430--454, 1972.

[9] D. Kahneman and A. Tversky. On the Psychology of prediction, Psychological Review, vol. 80, pp. 237-251, 1973.

[10] V. Lifschitz, D. Pearce, and A. Valverde. Strongly equivalent logic programs. ACM Transactions on Computational Logic, 2:526-541, 2001.

[11] M. Osorio and J.C. Acosta. Towards modelling an intelligent calendar agent with LUPS, IASTED International Conference on Artificial Intelligence and Applications, (AIA'2003) Innsbruck, Austria, February, pp. 10--13, 2003.

[12] M. Osorio, J.A. Navarro and J. Arrazola. Applications of Intuitionistic Logic in Answer Set Programming, Journal of Theory and Practice of Logic Programming, TPLP , 2003.

[13] M. Osorio, J.A. Navarro, and J. Arrazola. Equivalence in Answer Set Programming, Proceedings of LOPSTR 01, LNCS 2372, pp. 57-75, Springer-Verlag, Paphos, Cyprus, November 2001.

[14] M. Osorio and F. Zacarias, Irrelevance of Syntax in updating answer set programs, Proceedings Of Fourth Mexican International Conference On Computer Science Enc'03, pp.183-188, Eds. J. H. Sossa, and E. Perez, México, 2003.

[15] D. Pearce. From Here to There: Stable negation in Logic Programming, in D. Gabbay, H. Wansing (Eds.) What is Negation? Kluwer Academic Publishers, Dordrecht,.

[16] A. Tversky and D. Kahneman. Belief in the law of small numbers. Psychological bulletin, Vol. 2, pp. 105--110, 1971.

[17] A. Tversky and D. Kahneman. Judgment Ander uncertainty: Heuristics and biases, American association for the Advancement of Science, vol. 1, pp. 1124--1131, 1971.

[18] Fernando Zacarías, Mauricio Osorio, José Arrazola. Updates based on Structural Properties –USP-. Gests international transactions on computer science and engineering, pp. 61-72, issn: 1738-6438, isbn: 89-953729-5-8, October 2005.

.