Appeared at the Second Workshop on Deep Knowledge Representation Challenge, Playa Vista, CA, July 2012

Process Interruption Reasoning

Vinay K. Chaudhri, Stijn Heymans, Neil Yorke-Smith
Artificial Intelligence Center, SRI International, Menlo Park, CA 94025, USA
{firstname.lastname@sri.com}

Abstract

Biological processes such as Cellular Respiration have an intricate structure that defines the ordering amongst different steps and the participants in each step. A person who understands the process is expected to be able to reason with how the process is affected if one or more steps is interrupted. In this paper, we analyze a family of questions about process interruption, and present reasoning patterns that an automated reasoner can use to answer them. Our reasoning patterns rely on the order of steps of the process and the participants of those steps. We suggest that this approach leads to more intuitive and simpler reasoning than an approach of based on theory of intentions [9,10], or an approach that relies on qualitative simulation [8].

Introduction

Biological processes such as cellular respiration and fermentation feature an intricate structure of steps and sub-steps, and intimate connections with related processes within the larger biological setting. A person who understands a biological process—for example, having digested the contents of a textbook—is expected to reason with how the process is affected if one or more steps is interrupted.

As concrete examples, recall high school biology [1] and consider the following questions:

1. In the absence of oxygen, yeast cells can obtain energy by which process?
2. What happens if rubisco production is blocked in plant cells?
3. The rate of reaction of the electron transport chain that functions in oxidative phosphorylation can be reduced by removing what substance?

We characterize the above questions as examples of *process interruption* questions because they ask about the behavior of the process if we introduce some form of interruption: remove normally available raw material to the process, stop some related process, or wish to change the behavior of the process.

To answer the first question, we would need to know what part of cellular respiration in a yeast cell could continue and produce energy without oxygen—or what alternative processes with the cell could produce energy without oxygen. To answer the second question, we would need to know what processes use as raw material the output from rubisco production which they could not obtain from elsewhere. To answer the third question, we would need to know removal of what substance would slow down the reaction of the electron transport chain.

We argue that answering the above family of process interruption questions can be done by *process description analysis*: that is, reasoning amongst the ordering of the steps and the entities that participate in those steps. This process description corresponds to a flow chart that describes a process. We give examples of such reasoning for the above three sample questions, and from there, induce an abstract algorithmic approach that could be applied to a variety of process interruption reasoning questions. We will evaluate our approach on a suite of over 100 process interruption reasoning questions to illustrate that the approach is general, tractable, and scales beyond the above three examples.

Representation of Process Structure

Our work is in the context of a system called *AURA* for representing knowledge in textbook and answering questions by deductive reasoning [2]. The AURA knowledge base consists of a set of hierarchically organized classes and a vocabulary of relations. The key relations that are of interest for representing process structure are as follows:

A subevent B	B is a step of process A
A first-event B	B is the first step of process A
A next-event B	Process B follows process A
A raw-material B	Process A uses B as input
A result B	Process A produces B as result
A has-function B	Process B is a function of entity A
A agent B	Entity B is the agent of process A
A site B	Entity B is location of process A

In addition to the relations listed above, we support the standard relations from qualitative process theory: positive influence, negative influence, directly proportional and inversely proportional [3].

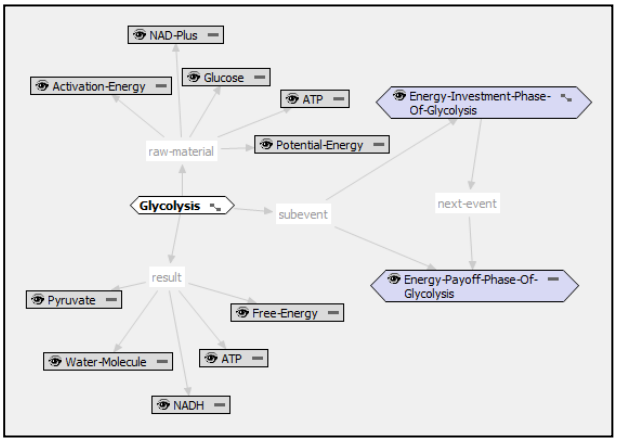

Figure 1: A partial representation of Glycolysis.

As a concrete example, we show in Figure 1 a portion of the representation of Glycolysis in the AURA knowledge base (KB), and give below its representation in first order logic.

∀x.[**instance-of(x, Glycolysis)** ⇒
  ∃ e1,e2,e3,n1,n2,**g**,a1,a2,w,p,eip,epp
 [instance-of(e1,Activation-Energy) ∧
  instance-of(e2,Potential-Energy) ∧
  instance-of(e3,Free-Energy) ∧
  instance-of(n1,NAD-Plus) ∧
  instance-of(n2,NADH) ∧
  instance-of(a1,ATP) ∧
  instance-of(a2,ATP) ∧
  instance-of(g,Glucose) ∧
  instance-of(p,Pyruvate) ∧
  instance-of(w,Water-Molecule) ∧
  instance-of(eip,
  Energy-Investment-Phase-of-Glycolysis) ∧
  instance-of(epp,
   Energy-Payoff-Phase-of-Glycolysis) ∧
  raw-material(x,e1) ∧
  raw-material(x,e2) ∧
  raw-material(x,a1) ∧
  raw-material(x,n1) ∧
  raw-material(x,g) ∧
  result(x,e3) ∧ result(x,w) ∧
  result(x,p) ∧ result(x,n2) ∧
  result(x,a2) ∧
  subevent(x,i) ∧ subevent(x,p) ∧
  next-event(i,p)]]

In the graph of Figure 1, the Glycolysis node, seen centrally in white, is universally quantified; the rest of the nodes are existentially quantified. The other nodes shown in the white background without border are relations drawn from our ontology. For example, we can read a part of the graph to mean: For every instance *x* of Glycolysis there must exist an instance *g* of Glucose, such that *g* is a raw-material of *x*. This corresponds to the bold lines in the fragment above.

The representation considered here is only a small subset of the representation supported in the *KM* knowledge representation system used in AURA [4].

To answer questions about the AURA representation, we *materialize* the knowledge base, by using skolem instances for each class. In particular, materializing a first order 'rule' such as that of Glycolysis would result in a set of ground literals (only partially shown):

```
instance-of(g1,Glycolysis)
instance-of(_Energy-Investment-Phase(g1),
            Energy-Investment-Phase)
instance-of(_ATP1(g1),ATP)
instance-of(_ATP2(g1),ATP)
subevent(g1,_Energy-Investment-Phase(g1)),
         raw-material(g1,_ATP1(g1))
result(g1,_ATP2(g1))
```

Intuitively, the variable in the head of the rule (*x*) is replaced by a skolem individual *g1*, and for each variable in the body of the rule we introduce a new skolem function linking it to *g1*. If this is performed for each rule in the KB (and thus for each concept in the KB), we have a finite materialization of the KB.

Note that this materialization is not complete with respect to reasoning. Indeed, we inevitably lose information by materializing rules only to up to one level. For example, ATP would be associated with its own definition but for the particular skolem function `_ATP1(g1)` introduced for Glycolysis this definition would not be explicitly materialized.

This finite materialization is called the *ABOX* in AURA (akin to the ABOX concept in Description Logics). A significant advantage of the ABOX is that it allows for the solving of simple conjunctive queries, possibly extended to unions of conjunctive queries with safe negation.

Intuitively, a *conjunctive query* (CQ) with safe negation is a conjunction of possibly negated literals where each variable occurs in a positive literal. We explain *safe negation* later with an example. Taking the disjunction of such conjunctive queries results in a *union of conjunctive queries* (UCQ) with safe negation. It is exactly in the realm of UCQs with safe negation that the process interruption queries we describe in this paper will reside.

For example, suppose we wish to answer the question: *In the context of Glycolysis, what step follows the energy investment phase?* We can formalize this question as the following conjunctive query (where `?x` denotes an unbound variable *x*):

```
instance-of(?g,Glycolysis) ∧
instance-of(?i,Energy-Investment-phase) ∧
instance-of(?j,Event) ∧
subevent(?g,?i) ∧
next-event(?i,?j)
```

All variables in the query are assumed to be existentially quantified. When evaluated by a standard first order solver (e.g., [5]), on the logical formulation of Figure 1, the query will return the term:

```
next-event(_Energy-Investment-Phase-of-
Glycolysis(g1),_Energy-Pay-off-Phase-of-
Glycolysis(g1))
```

Here both `_Energy-Investment-Phase(g1)` and `_Energy-Pay-Off-Phase(g1)` are skolem terms of *g1* and are instances of the class Energy-Investment-Phase-of-Glycolysis and Energy-Pay-Off-Phase-of-Glycolysis respectively, and *g1* is an instance of Glycolysis that was used to ground the knowledge base, i.e., in the materialization process described earlier.

We have intentionally used a very simple example to illustrate the process representation and reasoning. In general, the evaluation of UCQs with safe negation query can involve arbitrary inference by inheritance reasoning, rule-based reasoning, or both. Despite the undecidability of UCQs with safe negation in the presence of expressive KBs (e.g., already CQs with safe negation are incomplete in the presence of $\mathcal{AL}$ axioms—$\mathcal{AL}$ is a basic Description Logic [6]).

We have plans to experiment with deeper materializations of the AURA KB, increasing the levels to five. While becoming slower, such a larger ABOX should deal with most incompleteness in the system that is of relevance for our prototypical user, and, importantly, would still retain decidability of querying.

Examples of Interruption Reasoning

In this section, we analyze the three sample questions we presented earlier.

Reasoning about participants: Energy in Yeast Cells

Let us consider the first of the sample questions in more detail: *In the absence of oxygen, yeast cells can obtain energy by which process?*

The model answer of this question, as provided by a Subject Matter Expert (SME), is: *In the absence of oxygen, yeast cells undergo Fermentation to produce energy, for example through Alcohol Fermentation or Lactic Acid Fermentation.* Based on a discussion between multiple SMEs, it was apparent that the essence of the answer is: *Fermentation*.

To answer this question, the reasoner would examine each process by which a yeast cell can produce energy, and identify those processes that can complete successfully without oxygen. Such reasoning can be declaratively stated using the following query:

```
instance-of(?p,Process) ∧
instance-of(?y,Yeast-Cell) ∧
instance-of(?o,Oxygen) ∧
instance-of(?e,Energy) ∧ site(?p,?y) ∧
result(?p,?e) ∧ not raw-material(?p,?o)
```

This query will return all processes that occur in a Yeast Cell that produce energy but do not require Oxygen as one of the raw-material inputs. The query will return `Fermentation` as one of the answers. With a suitable facility to describe a process in English, AURA would describe Fermentation in a manner close to the rest of the model answer.

This particular conjunctive query uses negation. This negation is however *safe*: the variables `?p` and `?o` occur also positively in the query. Hence, intuitively, one can execute the part of the conjunctive query that does not contain `not raw-material(?p,?o)` (the negation) and subsequently remove from the answers the solutions for which `raw-material(?p,?o)` holds.

Reasoning about sub-step interruption: Rubisco Production

Next, let us consider the second sample question. This question hinges not on inputs to a process but on sub-steps of a process: *What happens if rubisco production is blocked in plant cells?*

This question is more involved than the first question, as demonstrated by the SME-provided model answer: *Rubisco is a key enzyme that functions at the beginning of the Calvin Cycle in photosynthesis, and in a competing reaction of photorespiration. Both reactions would cease without rubisco, meaning the plant would not be able to manufacture sugar and would lose some photo protection. Additionally, side products of the Calvin Cycle, NADP+ and ADP, would not be produced, so the Light Reaction that requires these would be negatively impacted. Over time, photosynthesis would cease.*

Our approach to this question begins by recognizing that the question can be rewritten as: *In the absence of rubisco, what processes in plant cells are impacted?* In fact, a more explicit version of the question would be: *In the absence of rubisco, what processes in plant cells are negatively impacted?* SMEs confirmed that this rewriting is permissible and maintains the intent of the question.

Hence, the question requires two reasoning steps, each of which can be formulated using a logical query. The first step is to determine processes that are affected when the rubisco production is blocked. We can formulate this task as the following query:

```
instance-of(?p,Process) ∧
instance-of(?c,Plant-Cell) ∧
instance-of(?r,Rubisco) ∧ site(?p, ?c) ∧
  ( raw-material(?p,?r) ∨ agent(?p, r) ∨
    has-function(?p,?r) )
```

Note the presence of disjunction in the query; we can easily rewrite this as a disjunction of conjunctions and thus as a UCQ.

With the above query, we identify the processes negatively affected by the absence of Rubisco. When we evaluate this query on the current KB, we get the answer: Carbon Fixation in which Carbon Dioxide, Ribulose Bisphosphate, and ATP are used to generate Three-Phosphoglycerate.

The second step is to describe the implications of the negatively affected processes. When a process is negatively affected, the larger process of which it is a part, and the steps that follow it, are also negatively impacted. This can be formulated using the following query:

```
instance-of(?p,Process) ∧
instance-of(?p1,Process) ∧
( sub-event(?p1,?p) ∨ next-event(?p,?p1) )
```

When evaluated on the current knowledge base, this query should return: Carbon Fixation is a step of the Calvin Cycle, which will result in NADP+, ADP, and Phosphate Ion, which are used in the Light Reaction, the earlier step in Photosynthesis.

Thus, using a combination of the two queries—i.e., in practice, rewriting the two as one UCQ—gives us an answer that comes very close to the model answer.

Reasoning about qualitative influences: Electron Transport Chain Reaction

Let us now consider the third sample question, which is an example of what we term *rate reasoning*, i.e., it concerns rates of reactions: *The rate of reaction of the electron transport chain that functions in oxidative phosphorylation can be reduced by removing what substance?*

The model answer is: *NADH, $FADH_2$, and O_2 are important reactants in the electron transport chain pathway and its reactions, so removing them would stop the pathway.*

Note that this is a non-elementary form of a rate reasoning question. A more simple form would be: *Will a decrease of NADPH production decrease the rate of the Calvin cycle?* An intermediate form would be Question 9 of this year's Deep KR Challenge: *How does the oxidation of NADH affect the rate of glycolysis?*

As before, the first step in our approach is to examine the sub-steps of the process for their inputs. In the absence of influence relation information in the knowledge base, we make the assumption that if the inputs of a process are reduced, then the outputs are reduced, i.e., a direct proportionality relationship. In general, this need not be the case, for example because one of the inputs to a step in a process might be an inhibitor, whose removal or reduction thus accelerates the process, leading to potentially greater outputs. If influence or proportionality relationships are present in the knowledge base, we will take them into account.

The following query expresses what we are seeking:

```
instance-of(?p,Process) ∧
instance-of(?p1,process) ∧
instance-of(?e,Entity) ∧
  ( raw-material(?p,?e) ∨
    (subevent(?p,?p1)
      ∧ raw-material(?p1,?e)) )
```

As before, we can easily rewrite this query as a UCQ. Instantiating `?p` by Electron-Transport-Chain and solving for every `?e` (i.e., multiple solutions) obtains NADH, $FADH_2$, and O_2 as the result.

A Generic Template for Reasoning

An analysis of over one hundred questions in the process interruption reasoning test suite [7] suggests a triage of the questions into three categories:

1. Questions that in essence do not concern process interruption — can be answered by another means.
2. Questions that concern process interruption, that can in principle be answered with the content from the current AURA knowledge base — we call these *reasoning-bound*.
3. Questions that concern process interruption, that cannot be answered with the content from the current AURA knowledge base — we call these *representation-bound*.

The first category is uninteresting for the purpose of this paper. The third category, by contrast, contains some questions that are beyond the current scope of what we can expect AURA to adequately answer. Some of these will be answerable if the ontology coverage was extended. Others, however, contain fundamental representational issues, such as secondary influences. Thus we have focused most of all on the second category, reasoning-bound questions. Nonetheless, it should be noted that our approach applies across all three categories and across the whole of the test suite.

Based on examples such as the three sample questions described above, and on our analysis of the process interruption reasoning test suite, we think abstractly of these questions as having three main components: a *context*, an *interruption*, and a *query*. We have enumerated the possible values for each of these components, and can systematically generate process interruption reasoning questions using a cross product of those values.

Hence, to answer each question requires providing a formal description of literals for each value of the component, and then simply taking their conjunction to construct a query—along the lines of those in the previous section for the three sample questions.

In the remainder of this section we enumerate possible values for each of the three components of the query, apply it to the three examples we considered earlier, and then give data on how broadly is the approach applicable to the full test suite.

A ***context*** may be one of the following: an entity, a process (or its sub-process), a function or functional role, or a location.

An ***interruption*** may be one of the following: a raw-material is added, removed, increased, or decreased; a process or sub-process is blocked, inhibited, removed or accelerated; a process or step of a process is substituted (by another process or step); or the rate of a process is increased or decreased.

A ***query*** may ask for one of the following: parts of the process that are affected or unaffected, effect on the output or the rate of the process, the cause of interruption, or what substitution could be done to achieve the same effect if there was no interruption.

A process interruption reasoning question then has the following generic form: In ***context***, if ***interruption***, then ***query***. Note that in some questions, the context may be omitted altogether.

To illustrate, let us first formulate the three questions we considered earlier using this general template. We treat the first question in detail. Here, the context is a Yeast Cell, the interruption is that Oxygen is no longer available as raw material, and we wish to query for which process could be substituted to output energy. Hence we can formulate a query as the conjunction of the three parts:

context: `instance-of(?y,Yeast-Cell) ∧ site(?p, ?y)`
interruption: `instance-of(?o,Oxygen) ∧ not raw-material(?p,?o)`
query: `instance-of(?p,Process) ∧ instance-of(?e,Energy) ∧ result(?p,?e)`

Note that some care needs to be taken when we talk for illustration purpose of combining queries or, as above, taking a conjunction of parts. Indeed, in order to obtain the well-understood format of unions of conjunctive queries with safe negation, just conjoining the different terms is in general too naive. However, we can rewrite such a conjoining to obtain a valid UCQ.

For example, if we have on the one hand a UCQ:

```
( instance-of(?p,Process) ∧
  instance-of(?p1,Process) ∧
  sub-event(?p1,?p) ) ∨
( instance-of(?p,Process) ∧
  instance-of(?p1,Process) ∧
  next-event(?p1,?p) )
```

and on the other hand the query

```
instance-of(?p,Process)
instance-of(?r,Rubisco)
raw-material-of(?p,?r)
```

We obtain, after suitable rewriting, the UCQ:

```
( instance-of(?p,Process) ∧
  instance-of(?p1,Process) ∧
  sub-event(?p1,?p) ∧
  instance-of(?r,Rubisco) ∧
  raw-material-of(?p,?r) ) ∨
( instance-of(?p,Process) ∧
  instance-of(?p1,Process) ∧
  next-event(?p1,?p) ∧
  instance-of(?r,Rubisco) ∧
  raw-material-of(?p,?r) )
```

This rewriting presumes that variables in both original queries actually refer to the same instance when the same variable is used (`?p` in the above example). If this is not the case, then a simple rewrite to a UCQ will not deliver the required result. Essentially, while we can split up the templates in different logical parts, we need still to take care that we appropriately determine the variables on which to 'join' the queries. In the above example, the process that has a super event, and the process with raw material Rubisco, are intended to be the same process—and as such any combination of those queries has to take this into account (by for example properly renaming variables).

After this extended discussion about the formulation of the first sample question, we treat the remaining two questions more briefly. In the second question, the context is a plant cell, the interruption is that the rubisco is no longer produced, and we wish to query which other processes will be affected. In the third question, the context is oxidative phosphorylation, the interruption is the reduction in the rate of the electron transport chain reaction, and we wish to query for the cause that might lead to the interruption.

A complete set of process interruption reasoning test suite questions is available online [7]. Of the test suite, only about 10% of questions are representation-bound; the remainder are reasoning-bound or non-PI questions. Excluding the representation-bound questions which are beyond the current scope of AURA, we estimate that 85% of the test suite can be addressed using our generic template approach.

Default assumptions

Implicit in the approach just described are a set of assumptions. We saw one earlier in the direct proportionality relationship assumed unless the KB states otherwise. We conclude this section making explicit our assumptions. Some derive from the KB and representation within AURA; others stem essentially for simplicity for our reasoning approach.

- processes are independent, unless there is an explicit link between them
- processes do not contain infinite cycles
- steps in a process are atomic
- steps within a process are not temporally ordered, unless there is an explicit temporal or causal relation between them
- increases of the quantity of a raw-material increases the quantity of the result and vice versa
- introduction of (superfluous) raw-materials (of a new substance) to a step or process has no impact upon the process
- a process in an organism or cell could get a raw-material input from any other process that produces that raw-material as result, assuming that the process can happen in that cell or organism
- unless stated otherwise, the outputs of successive iterations of a step or process are cumulative: if one iteration produces k instances of a resultant, then n iterations produce nk instances
- if a raw-material for a step is removed, that step will not happen
- if a step is interrupted (for any reason), its result will not be produced
- a step in a process produces all its outputs in full exactly at its conclusion, and none before
- a process is successful only if all of its steps are successful
- provision of (sufficient) of each of the inputs of a process in necessary but not sufficient for the success of the process

Such default assumptions are key to correctly answering process interruption questions with our generic template and conjunctive query approach. AURA must be smart enough to figure out when such default assumptions are violated and do something different in those cases; this will be future work.

Comparison to Related Work

In this section, we consider two alternative approaches to process interruption reasoning recently proposed.

Bredeweg and colleagues [8] argue that we should build qualitative models of processes in a biology textbook, and in response to a process interruption, generate a complete qualitative simulation to answer a question. Our approach agrees with their proposal in that a qualitative representation needs to be built but differs in the need for a complete qualitative simulation for the following reasons. First, textbooks rarely provide knowledge necessary to run a qualitative simulation needed for their approach. Second, even if it were possible for a knowledge engineer to construct a qualitative model, running a simulation is not *necessary* because textbooks typically enumerate the specific states that teachers expect a student to learn about. Such states could be simply captured and described using the sort of process descriptions we have considered in our work. Finally, qualitative simulation deals with only qualitative influences, proportionalities, and correspondences, and thus is significantly more limited in dealing with the full range of interruptions in the question suite.

An approach based on qualitative simulation has the advantage that it helps us think through the completeness of knowledge, and it can help clarify and deepen the understanding of the domain knowledge. We argue that this advantage is, however, far outweighed by the complexity, cost, and the cognitive distance between the output from a qualitative simulation and the knowledge in the textbook.

In an alternative proposal based on a modular action language [9,10], Gelfond and colleagues argue in favor of extending the knowledge base with a theory of intentions. We consider here two example axioms from that theory:

- If an agent intends a process to occur, and we know that it is not false that the process has not occurred, then the process will occur.

  occurs(A, I) ← intend(A, I), not ¬occurs(A, I)

- If an agent intends a process to occur, and the process has not occurred, and the agent does not intend for the process to not occur in the next situation, then the agent intends the process to occur in the next situation.

  intend(A, I1) ← intend(A, I), ¬occurs(A, I),
  not ¬intend(A, I1), I1 = I + 1

Gelfond and colleagues have shown that such a theory of intentions can be used to answer a question about cell cycle; they have not yet shown its generality to the full suite of process interruption reasoning questions. But the primary disadvantage of this approach is that biology teachers do not think about the processes described in a textbook in terms of agentive intentions (as formulated above), and hence there is almost no hope for us to get a biologist to verify the validity of a theory of intentions, and more importantly, to understand the explanation behind the reasoning performed by the system.

Summary and Outlook

This paper presented a generic approach for answering process interruption questions. Such questions naturally arise in the context of a textbook description of processes. Our approach is based on creating a template of process interruption reasoning of three components: a context, an interruption, and a query of interest. This abstract template enables us to translate such questions into logical queries that can be evaluated by a conventional first order reasoner. We described how this approach generalizes to over 100 questions from a test suite. Our current work is the implementation and empirical evaluation of our approach.

Acknowledgments

This work was funded by Vulcan Inc. We thank Andrew Goldenkranz for providing the questions and model answers for the process interruption reasoning test suite. We thank Atalay Ozgovde for undertaking an initial analysis of the process interruption questions. We thank the Subject Matter Experts who assisted with the analysis and provided additional model answers.

References

[1] Jane B. Reece, Lisa A. Urry, Michael L. Cain, Steven A. Wasserman, Peter V. Minorsky, and Robert B. Jackson. *Campbell Biology*, 9th ed. Benjamin Cummings, 2011.

[2] David Gunning, Vinay K. Chaudhri, Peter Clark, Ken Barker, Shaw-Yi Chaw, Mark Greaves, Benjamin Grosof, Alice Leung, David McDonald, Sunil Mishra, John Pacheco, Bruce Porter, Aaron Spaulding, Dan Tecuci, and Jing Tien. Project Halo Update — Progress Toward Digital Aristotle. *AI Magazine*, 31(3), 33-58, Fall 2010.

[3] Ken Forbus. Qualitative Process Theory. *Artificial Intelligence*, 24, 85-168, 1984.

[4] Peter E. Clark and Bruce Porter. *Knowledge Machine Users's Guide*. Technical Report, University of Texas at Austin.

[5] Riccardo Rosati. *The Limits of Querying Ontologies*. Proceedings of International Conference on Database Theory (ICDT'07), 164-178, 2007.

[6] Lawrence C. Paulson. The Foundation of a Generic Theorem Prover. *Journal of Automated Reasoning*, 5(3), 363-397, 1989.

[7] Andrew Goldenkranz. *A Test Suite of Process Interruption Reasoning Questions*. Online at: http://www.ai.sri.com/halo/public/2012-qr/Process%20Interruption%20questions%20v5.xls

[8] Bert Bredeweg et al. *Qualtiative Reasoning for Cellular Respiration*. Proceedings of 2012 Qualitative Reasoning Workshop, 2012.

[9] Daniela Inclezan and Michael Gelfond. *Representing Biological Processes in Modular Action Language ALM*. Proceedings of AAAI 2011 Spring Symposium on Common Sense Reasoning, 2011.

[10] Michael Gelfond and Daniela Inclezan. *Reasoning about Dynamic Domain using Modular Action Language ALM*. Technical Report, Texas Tech University. Online at: http://www.webpages.ttu.edu/dincleza/ALM/ALM-technical-report-2010.pdf