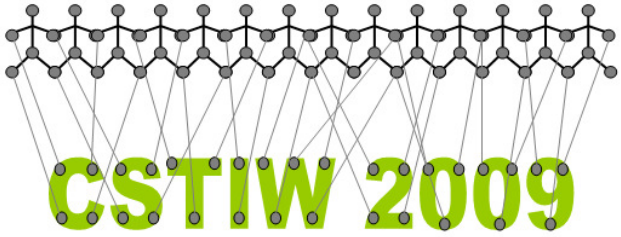
Fourth Conceptual Structures Tool Interoperability Workshop (CS-TIW 2009)

Madalina Croitoru, Jérôme Fortin, Robert Jaschke



Organization (Alphabetical Order)

Madalina Croitoru, LIRMM, University Montpellier II, France

Jérôme Fortin, LIRMM, University Montpellier II, France

Robert Jäschke, University of Kassel, Germany

Program Committee (Alphabetical Order)

Oscar Corcho, Manchester University, Great Britain

Raúl García Castro, UPM Madrid, Spain

Harry Delugach, University of Huntsville, USA

Guy Mineau, Université Laval, Canada

Heather D. Pfeiffer, New Mexico State University, USA

Simon Polovina, Sheffield Hallam University, UK

Uta Priss, Edinburgh Napier University, UK

Gerd Stumme, University of Kassel, Germany

The Fourth Conceptual Structures Tool Interoperability Workshop (CS-TIW 2009) was held in Moscow, Russia on the 26th of July 2009, collocated with the 17th International Conference on Conceptual Structures (ICCS 2009) "Leveraging Semantic Technologies". Information about the workshop is available at the webpage: http://www.kde.cs.uni-kassel.de/ws/cs-tiw2009.

The title of this year's workshop, "Vision and Applications" was chosen to emphasize the new challenges, problems and issues that have appeared in the context of knowledge representation that involve the visual manipulation of information by domain experts. Furthermore, such knowledge has to be manipulated with fast results within a logic based framework. In this context conceptual structure tools play a crucial role as their graph based foundations allow for (1) rigorous algorithmic reasoning and (2) user validation for both reasoning and representation. It is a unique environment for integrated research in discrete mathematics, human computer interaction, cognitive science and engineering.

The workshop brought together researchers from diverse communities with the common goal of a fruitful discussion on the above mentioned interoperability issues by raising mutual awareness of ongoing research and existing technologies from which each community could benefit.

We wish in full to express our appreciation to all the authors of submitted papers and to the members of the Program Committee for all their work and valuable comments.

July 2009,

Madalina Croitoru
Jerome Fortin
Robert Jäschke

Standard CGIF Interoperability in Amine

Adil Kabbaj[1], Ivan Launders[2] and Simon Polovina[3]

[1]INSEA, Rabat, Morocco
akabbaj@insea.ac.ma
[2]BT Global Services, PO Box 200, London, United Kingdom
ivan.launders@bt.com
[3]Cultural, Communication & Computing Research Centre (CCRC)
Sheffield Hallam University, Sheffield, United Kingdom
s.polovina@shu.ac.uk

Abstract. The adoption of standard CGIF by CG tools will enable interoperability between them to be achieved, and in turn lead to the interoperability between CG tools and other tools. The integration of ISO Common Logic's standard CGIF notation in the Amine platform is presented. It also describes the first steps towards full interoperability between the Amine CG tool (through its Synergy component) and CharGer, a representative CG tool that supports similar interoperability and for process (or 'active') knowledge as well as declarative knowledge. N-adic relations are addressed as well as semantic compatibility across the two tools. The work remarks on the successes achieved, highlighting certain issues along the way, and offering a real impetus to achieve interoperability.

1 Introduction

The adoption of a standard that will enable the actual interoperability of Conceptual Graphs (CG) tools both between them and other tools has been a long-standing interest of this community. Through such means the CG tools themselves will become more widely adopted as there is less 'lock in' to the limitations of a particular tool, and in turn collectively bring the power of CG into wider arenas such as the Semantic Web. It is in this context that the standard for the Conceptual Graph Interchange Format (CGIF), a fully conformant dialect of ISO Common Logic (CL) [1], describes a set of transformation rules. CGIF is a representation for conceptual graphs intended for transmitting CG across networks and between IT systems that use different internal representations. The design goal for CGIF was to define a minimal abstract syntax and notation to achieve:

- a proper foundation with translations to Knowledge Interchange Format (KIF) and permits extensions to be built upon;
- a rapid interchange of CG between CG tools.

Annex B [1] provides the CGIF syntax[1] and its mapping to CL. The Amine software platform [3,4] has accordingly been developed towards interoperability both in terms of syntax and semantics.

2 Integrating Standard CGIF in Amine for Interoperability

The integration of the Standard CGIF notation in Amine [3,4] (since Amine 6.0) involves many other integrations/additions (like a mapping for actors and a mapping for n-adic relations with n > 2), as follows.

2.1 From "Amine CGIF" to Standard CGIF notation

In previous versions, Amine adopted a proper subset of CGIF that we refer to "Amine CGIF". There are certain differences between the Amine CGIF and Standard CGIF notation. Amongst them there is the use of the "pseudo-designator" in "Amine CGIF". To explain, in Standard CGIF the coreferent has two functions:

- It is used in the CGIF notation (and CG's Linear Form (LF) notation) to specify occurrences of the same concept in the same CG. Such a coreferent exists neither in the Display form, nor in the internal representation of the CG.
- It is used to specify coreferent links between concepts that belong to embedded CG.

The Pseudo-designator is used in Amine CGIF to play the first function of a coreferent. Instead of pseudo-designator, standard CGIF notation makes a direct use of a coreferent. For instance, instead of the pseudo-designator #1 in this example:

```
[Work #1] [Person: Jane] [House : myHouse]
(agnt #1 Jane)  (near  #1  myHouse)  (ageOf  Jane  [Age = 30])
(poss  myHouse  Jane)
```

Standard CGIF notation [1] will use a coreferent:

```
[Work *p] [Person: Jane] [House : myHouse]
(agnt ?p Jane)  (near  ?p  myHouse)  (ageOf  Jane  [Age:30])
(poss  myHouse  Jane)
```

In Amine CGIF the designator and descriptor represents two different fields of a concept. In standard CGIF notation, coreferent, designator and descriptor share the same and one field; namely the field after the concept type. In Amine CGIF, a separator is used to differentiate between the designator and the descriptor. In standard CGIF notation, no separator is used to introduce and differentiate between the designator and the descriptor. The separator ":" is optional in Standard CGIF and there is great flexibility in ordering the coreferent, designator and descriptor; a coreferent can be placed before or after the designator and/or the descriptor.

[1] In describing the syntax of CGIF, B.1.2 [1] uses the Extended Backus-Naur Form (EBNF) notation as referenced in ISO/IEC 14977:1996. However the classic Sowa [5] examples are used in the paper as referenced in the standard for CGIF [1].

To assure backward compatibility with earlier versions of Amine 6.0 (i.e. that use just the Amine CGIF notation), Amine 6.0 adopts a CGIF syntax that integrates standard CGIF notation and Amine CGIF notation. A CG that was stored in Amine CGIF notation is thus still readable by Amine, but it will be reformatted into standard CGIF notation (i.e. a new save of the CG, in CGIF, will use standard CGIF notation).

Here is the syntaxic grammar of a concept in Standard CGIF notation in Amine (a complete description of the grammar of Standard CGIF notation in Amine is provided in the Amine Web Site[2]):

```
Concept ::= "[" [Comment] Conc_Type
[ParameterModes (generate in EndComment)] [CoReferent]
     [":"] [CoReferent] {Designator} [CoReferent] ["="] [Descriptor]
     [CoReferent] ["$" State (generate in EndComment)]
["&" Point (generate in EndComment)] [EndComment]     "]"
Comment ::= "/*", {(character-"*") | ["*", (character-"/")]}, ["*"],
"*/"
EndComment ::= ";", {character - ("]" | ")")}
```

Note that a concept can start with a comment and terminate with an end comment. A coreferent can be stated before or after the separator ":", before or after a designator, and before or after a descriptor. Separators ":" and "=" are optional. Parameter modes (appropriate to the Synergy component in Amine), States (appropriate to Synergy), and Point (appropriate to the CG display form) are also optional, and they can be specified inside the concept or in the end comment. Note also that the above syntaxic rule is used to define the syntax of concepts in both CGIF and LF notations. Thus, with Amine 6.0 we adopt the above syntaxic rule for the representation of concept in both LF and CGIF forms.

Example #1 To illustrate, Sowa's classical example ("John Goes to Boston by Bus") is shown in the LF notation, Standard CGIF, and Display Form (Figure 1):

LF notation:

```
[Go]-
   -instr->[Bus],
   -dest->[City :Boston ],
   -agnt->[Human :John ]
```

Standard CGIF notation (without graphical information):

```
[Go : *p1 ]
(instr ?p1 [Bus]) (agnt ?p1 [Human :John]) (dest ?p1 [City :Boston])
```

Standard CGIF notation (with graphical information):

```
[Go : *p1 ; & 235 89]
(instr ?p1 [Bus ; & 65 72]; (154 87)(235 97)(97 78)) (dest ?p1 [City
:Boston ; & 299 164]; (283 135)(253 106)(334 164)) (agnt ?p1 [Human
:John ; & 346 18]; (307 62)(255 89)(380 35))
```

(The graphical information, for both concepts and relations, are specified in the EndComment.)

[2] sourceforge.net/projects/amine-platform

Example #2 An example proposed by Sowa to illustrate nested CG:

LF notation:

```
[Believe] -
    -expr->[Human : "Tom" ],
    -thme->[Proposition : [Want] -
                    -expr->[Human : *x3 "Mary" ],
                    -thme->[Situation : [Marry] -
                                        -agnt->[Woman : ?x3 ],
                                         -thme->[Sailor]
                                ]
            ]
```

Standard CGIF notation:

```
[Believe : *p1 ]
(expr ?p1 [Human : "Tom" ]) (thme ?p1 [Proposition : [Want : *p2 ]
         (expr ?p2 [Human : *x3 "Mary" ])
         (thme ?p2 [Situation : [Marry : *p3 ]
                        (agnt ?p3 [Woman : ?x3 ]) (thme ?p3
[Sailor])
                                                    ])
                                        ])
```

2.2 Standard CGIF notation, Actors and "active concepts"

Concepts and relations capture declarative knowledge, but lack the capability for active or process knowledge. Actors[3] dynamically modify CG allowing active or process knowledge. Actors also provide the means to access external systems [2]. Actors constitute a standard element in CG theory, like concepts and relations. Standard CGIF [1] provides a precise syntax for their representation. Amine adopts another perspective: a concept can represent a static or a dynamic, executable entity (operation, function, process, etc.). Thus, concepts represent both “static concepts” and “dynamic concepts” (i.e. Actors). Amine aims to allow the integration of its Synergy engine with an ontology in order to perform actor-like processing[4].

As Amine integrates Standard CGIF notation (as of Amine 6.0), we can provide a mapping between “CG with Actors” and “Amine CG (with dynamic concepts)”. Thus a CG with Actors that is formulated in standard CGIF notation can be translated to an Amine CG. Hence, interoperability between Amine and other tools that use Actors is now possible, such as between Amine and the CharGer[5] CG software that has extensive support for Actors. Figure 2 is an example of this mapping: Figure 2.a shows CharGer visual display of a CG with Actors. CharGer provides the possibility to save a CG (with Actors) in standard CGIF (Figure 2.b). The file produced by CharGer can then be read and reformulated by Amine (Figure 2.c). However, to activate/execute the CG by Amine, semantic compatibility is required, in addition to the common use of standard CGIF notation. This point is discussed next.

---

[3] Not to be confused with Actors in the UML (www.uml.org)

[4] More information on the Synergy language can be found via the Amine website, amine-platform.sourceforge.net

[5] sourceforge.net/projects/charger

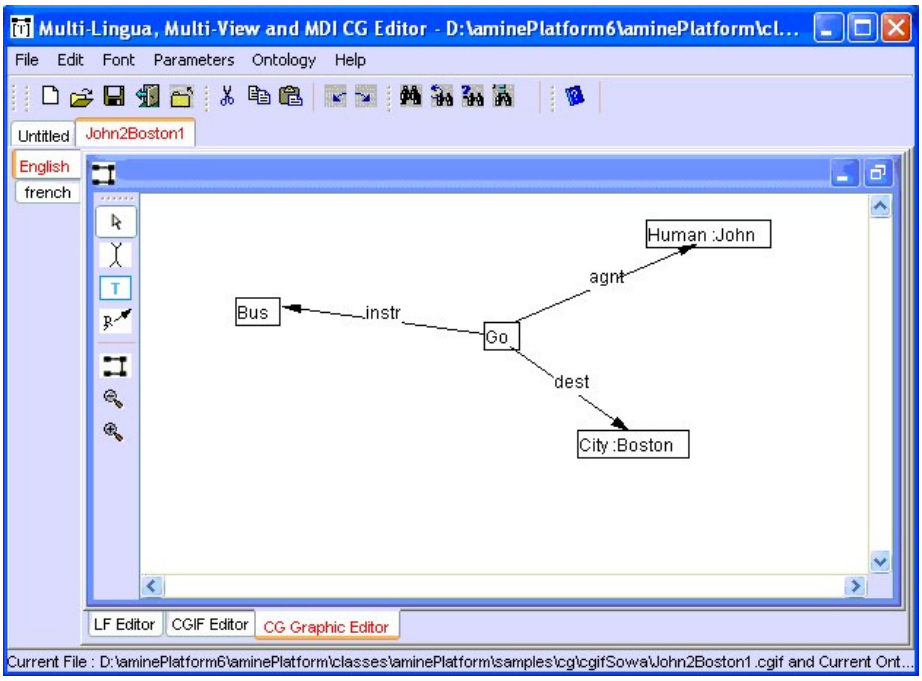


Figure 1: Diagram Form of a CG (in Amine)

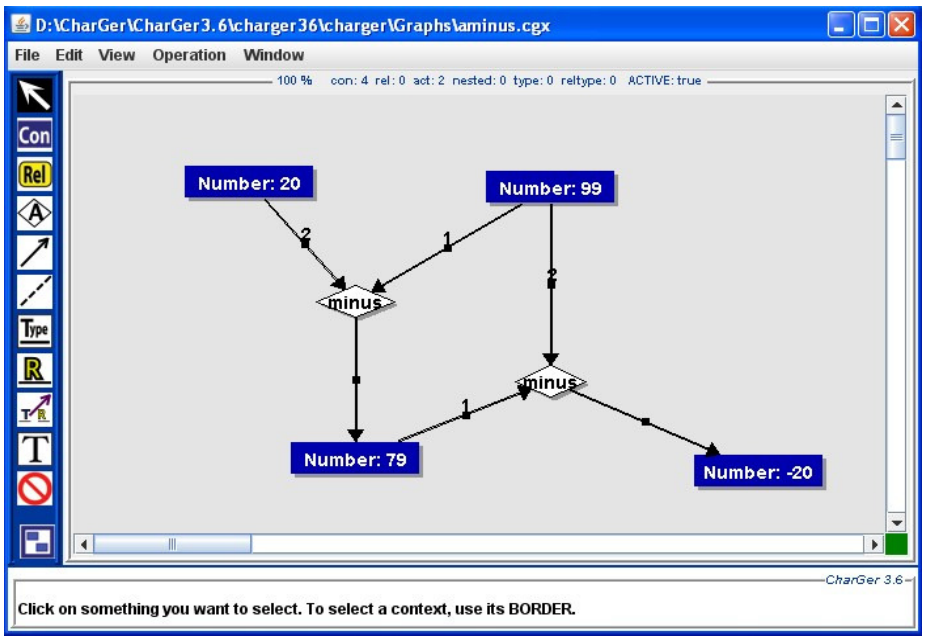


Figure 2.a: CharGer diagram notation of a CG with Actors

```
 [ /* <concept id="35979485995" owner="35979485989"> <type>
<label>Number</label> </type> <referent> <label>99</label>
</referent> <layout> <rectangle x="315.0" y="60.0" width="98.0"
height="25.0"/> <color foreground="255,255,255"
background="0,0,175"/> </layout> </concept> */ Number: 99]
...
 (/* <actor id="35979485996" owner="35979485989"> <type>
<label>minus</label> </type> <layout> <rectangle x="185.0" y="150.0"
width="60.0" height="25.0"/> <color foreground="0,0,0"
background="255,255,255"/> </layout> </actor> */ minus 99 20 | 79)
```

Figure 2.b: CharGer formulation of the above CG in Standard CGIF notation

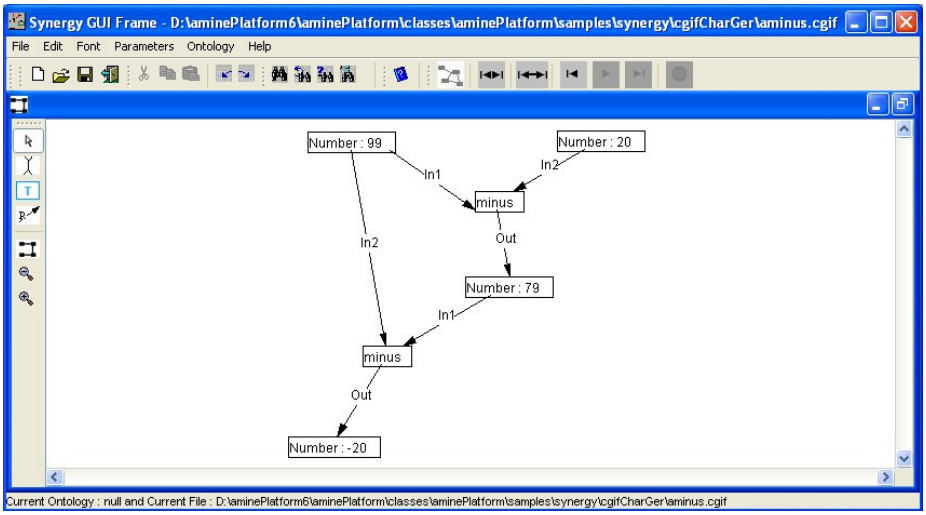


Figure 2.c: Amine's reformulation of a CG with Actors through standard CGIF translation

2.3 Standard CGIF, more than dyadic relations' reformulation in Amine

CG theory allows for conceptual relations that are more than dyadic, like the classical example of "Between" relation (Figure 3.a). Standard CGIF notation thus allows for the representation of more than dyadic relations. In Amine we consider dyadic relations only. However as Amine 6.0 integrates standard CGIF notation, we provide a mapping between "CG with more than dyadic relations" and "CG with dyadic relations only": CG with more than dyadic relations formulated in standard CGIF notation is thus translated to CG with dyadic relations only. This mapping is based on the representation of the more than dyadic relation by a concept and dyadic relations (Figure 3). With this mapping, we can now affirm that Amine handles n-adic relations (where n > 2) thereby providing interoperability with tools that allow the use of n-adic relations. Figure 3 provides the example of the "Between" relation (with "Betw" as a synonym). Figure 3.b shows how Amine 6.0 provides the possibility to use more than dyadic relation, through standard CGIF notation. But as illustrated by the diagram (Figure 3.c), the relation is reformulated in terms of dyadic relations.

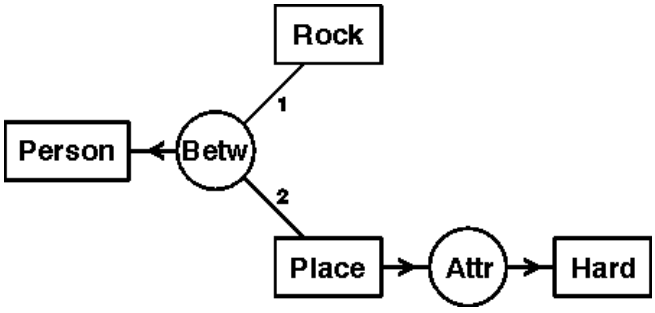


Figure 3.a: Triadic relation

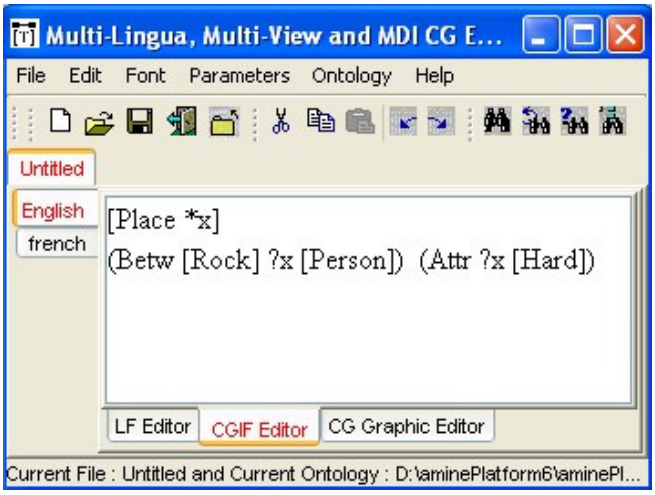


Figure 3.b: Triadic relation in standard CGIF

2.4 Semantic Compatibility

Sometimes, interoperability requires semantic as well as syntaxic compatibility. With the integration of standard CGIF in Amine 6.0, it is now possible for Amine to open and read CG that are created by CharGer. But this may not be enough; to enable a CG in CharGer to be executable by Synergy (and vice-versa), semantic compatibility may also be required. Synergy therefore provides semantic compatibility by:

- A mapping between CG with Actors (adopted by CharGer) and Amine CG with dynamic concepts (adopted by Amine);
- A mapping between CharGer's Actor identifiers with equivalent Synergy primitive operations. CharGer's Actor identifiers are added in Amine as synonyms for identifiers of Synergy primitive operations. For instance "plus" is added as a synonym of "Add", "minus as synonym of Sub", "multiply as synonym of Mul", etc.

- To guarantee CG that represent arithmetic expressions created by Synergy can be opened and activated by CharGer, CharGer's identifiers for arithmetic operations are preserved as the principal identifiers. Thus for instance, the Actor for addition will be expressed as [plus] instead of [Add].

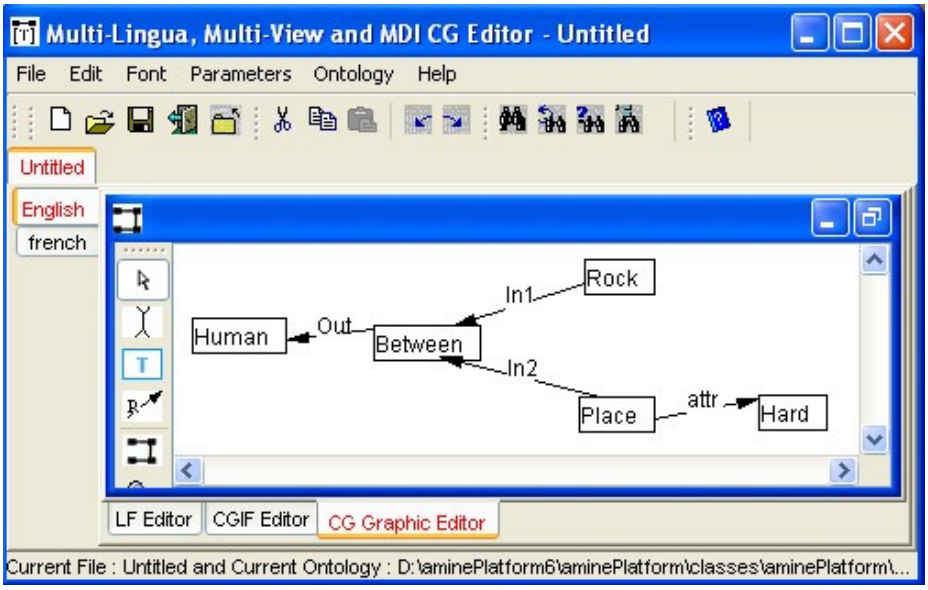


Figure 3.c: Reformulation of Triadic relation in Amine, through the CGIF translation

With the implementation of semantic compatibility between CharGer and Synergy, most of CharGer's CG can be activated and executed by Synergy. However since CharGer is equivalent to a subset of Synergy, certain CG with Amine/Synergy features cannot be activated by CharGer (such as Amine's CG that contain control relations, or CGs with defined concept types).

Thus, semantic compatibility remains partial because CharGer and Synergy consider a different semantic. Compatibility of the ontological level of the two tools is also partial: the ontology in CharGer corresponds to a type hierarchy, whilst ontology in Amine corresponds to a hierarchy of Conceptual Structures (Definition, Individual, Canon, Schema/Situation, Rule). Of course, a type can have no definition and no canon, and an individual can have no description. As the type hierarchy is a special case of Conceptual Structures hierarchy, any ontology used by CharGer can be automatically reformulated in Amine but not the inverse.

Semantic interoperability is also affected due to certain forms of CG that are presently unsupported in Amine; for instance "null" is not allowed as a concept type, a monadic expression is not allowed as a concept type, a monadic relation is not allowed (unless it can be re-expressed as a dyadic relation), and cardinality is not considered. Some of these features may be integrated in future versions of Amine, depending on how important it is considered by the CG and wider community that such functionality should be supported.

3 Concluding remarks

We have described the integration of Standard CGIF notation in Amine, and the impact of this integration on the interoperability between Amine and, by using CharGer as one exemplar, other (CG) tools. It was most heartening to experience interoperability actually being accomplished through the adoption of standard CGIF. This actual experience was not discouraged by CharGer version 3.6, being the version we worked with in this exercise, being able to store a CG in standard CGIF but had no support for reading a CG written in standard CGIF. Rather, such issues can be overcome (for example in this case in the next version of CharGer), and that our experience will act as a key encouragement.

We have also found ways to support semantic compatibility through our experiences with Amine and CharGer as we described, using Amine to recognise the differing semantics in another tool (CharGer) and adapt to it in a principled way.

Of course we need continued effort from all sides to achieve heightened interoperability between Amine, CharGer and across all CG tools. By raising some of the issues involved in doing so, but at the same time demonstrating the success we have had, we have provided a real impetus to achieve the interoperability that the Conceptual Structures community ultimately desires.

References

1. International Standards Organisation (2007) International Standards for Business, Government and Society ISO/IEC 24707:2007 Information technology -- Common Logic (CL): a framework for a family of logic-based languages. Last accessed on 17 June 2009 at http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39175
2. Delugach, Harry (2009) "A Practitioner's Guide to Conceptual Graphs", personal communication (available from the author)
3. Kabbaj A., *Development of Intelligent Systems and Multi-Agents Systems with Amine Platform,* in 15th Int. Conf. on Conceptual Structures, ICCS'2006, Springer-Verlag, 2006.
4. Kabbaj A., *An overview of Amine*, to appear in « Conceptual Structures in Practice: Inspiration and Applications », H. Schärfe, P. Hitzler (eds), Taylor and Francis Group, 2009.
5. Sowa, J., (1984). 'Conceptual structures: information processing in mind and machine', Addison-Wesley.

Argumentation Map Generation with Conceptual Graphs: the Case for ESSENCE

Aldo de Moor[1], Jack Park[2], and Madalina Croitoru[3]

[1] CommunitySense, Cavaleriestraat 2, 5017 ET Tilburg, the Netherlands,
ademoor@communitysense.nl
[2] Knowledge Media Institute, Open University, UK
j.b.park@open.ac.uk
[3] LIRMM (CNRS and University Montpellier II)
croitoru@lirmm.fr

Abstract. Argumentation maps are visual representations of argumentation structures, making it possible to efficiently examine the cumulative results of protracted, distributed, and complex argumentation processes. Such visualizations can be useful to, for example, informally assess the status of public debates. Although the elicitation of argumentation maps is well supported, the support for the (1) analysis, (2) comparison, and (3) generation of maps relevant to particular stakeholders is still an open research problem. To develop such services, conceptual graph theory and tools can prove to be very useful. We analyze the argumentation needs of the ESSENCE (E-Science/Sensemaking/Climate Change) project, which aims to create useful argument maps to assess the current state of the global debate on climate change. We outline a public investigator service, which would allow policy makers, journalists, etc. to quickly zoom in on only those stakeholders, positions, and arguments they are interested in.

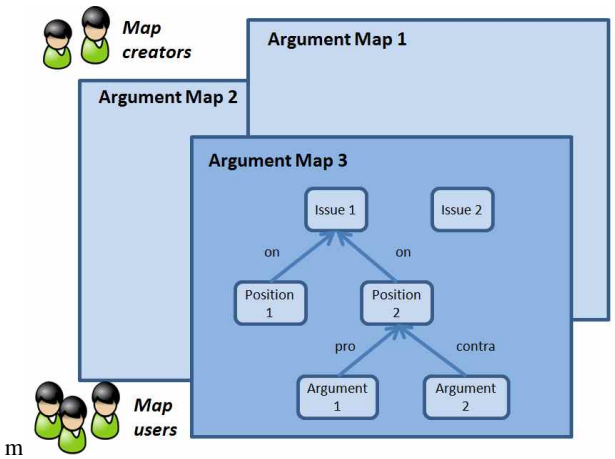
Key words: argumentation maps, conceptual graphs, services

1 Introduction

Argumentation maps are visual representations of argumentation structures, making it possible to efficiently examine the cumulative results of often protracted, distributed, and complex argumentation processes. Such visualizations can be very useful to, for example, informally assess the status of public debates. They help give an overview of the issues, positions, arguments pro and contra positions and other arguments of the stakeholders involved.

Although the elicitation of argumentation maps is well supported, support for the analysis, comparison, and generation of maps relevant to particular stakeholders is still an open research problem. Indeed, on the one hand the users of arguments are not necessarily the same as their creators. It is also very easy to get lost in the argument map space, especially when the maps grow in size, number, contributing authors, etc. Addressing such issues requires a careful pragmatic evaluation of collaborative technologies in their context of use, focusing on their *relevance* to the user community [5,

7]. Fig. 1 outlines a conceptual model of this pragmatic evaluation of argumentation maps.



m

Fig. 1. Argumentation maps in their context of creation and use

For instance, a policy maker in national government who is preparing new legislation about stricter carbon emission measures, may be especially interested in controversial issues, which are defined as those positions that business / environmental organizations have opposing arguments for / against. An argumentation map-based service should be able to present those parts of the overall climate change map which directly address this particular interest. Please note that the term "directly" is interpreted here in a somehow restrictive manner. In a first step we are interested in the **immediate** consequences and support arguments for a given issue that concerns the user. Once this service in place, it could be extended in order to include the **transitive closure** of pro / con arguments etc.

To improve the relevance of argument maps, automated semantic analysis can be useful. Argumentation maps are in fact semantic networks of argument nodes connected by meaningful relations. To analyze dependencies between map parts and answer queries, this basic semantic network structure can provide a useful starting point. Please note that in this paper we will not analyze the "semantics" of the relations or how they interact amongst each other (transitive closure of the same type of relation, what a "contradictory" relation means etc.). Instead, we will regard the relations as binary relations with a given pre-defined order.

Conceptual graph theory is a powerful formalism for knowledge representation and reasoning. Theory and tools can provide essential support especially in this application context. Intuitively this is due to the fact that Conceptual Graphs use graphs (a visual paradigm) for both representation and reasoning. This means that not only the main representational constructs of the language are graphs, but also that the operations that allow for reasoning (expressed in a subset of first order logic operations) are done via

graph based operations. Of course, the type of inference needed in the context of argumentation maps and first order logic subsumption provided by Conceptual Graphs have to be accordingly joined together.

To show how the worlds of argumentation maps and conceptual graphs can meet, we focus on addressing the argumentation needs of the ESSENCE (E-Science / Sensemaking / Climate Change) project[4]. This project aims to create argument maps to help stakeholders assess the current state of the global debate on climate change. In order to do so, stakeholders must be able to view highly customized maps which are those selections from the overall map which should most directly relate to their current information needs. To this purpose, we propose an architecture of a service based on a system of interoperating argument mapping and conceptual graph tools that allow such stakeholder-relevant maps to be dynamically generated from the overall argument map having been created at a particular moment in time. This service can be the basis for a range of stakeholder-specific services such as for policy makers or investigative journalists.

In Sect. 2, we explain the intuitive process behind the generation of relevant argumentation maps. Sect. 3 describes the role of conceptual graphs in providing the required service. Sect. 4 outlines an architecture of the ESSENCE argumentation map generator. We conclude the paper with a discussion and future directions of work.

2 Argumentation Mapping Support: From Elicitation to Generation

In this section, we introduce the ESSENCE case as a testbed for argumentation mapping support evaluation, and give a concrete example of the generation of relevant argumentation maps.

2.1 The ESSENCE Case

The ESSENCE project is the first major project that emerged out of the GlobalSensemaking Network[5]. It has a range of ambitious, but very important and interlinked goals [4]:

- **pilot software tools designed to help facilitate structured analysis and dialogue**. Map, summarise and navigate science and policy arguments underpinning the issues that people care about and collaborate with the development teams who are shaping these tools to meet the complex, systemic challenges of the 21st Century.
- develop a **comprehensive, distilled, visual map of the issues, evidence, arguments and options** facing the UN Climate Change Conference in Copenhagen, and being tackled by many other networks, which will be available for all to explore and enrich across the web.
- build a **definitive, public collective intelligence resource** on the climate change debate.

[4] `http://globalsensemaking.wik.is/ESSENCE`
[5] `http://www.globalsensemaking.net/`

– **support dialogue** that builds common ground, resolves conflict, and re-establishes trust

In May 2009, a kickoff workshop was held at the Knowledge Media Institute of the Open University in Milton Keynes, UK[6]. It was decided that a demonstrator will be built to show proof of concept of the usefulness of the argumentation mapping tools for helping make sense of the global climate change debate. These tools, however, will not be applied in isolation, but as part of a system of tools in a well-defined usage context. This socio-technical system is to provide a useful service to stakeholders involved in the debate.

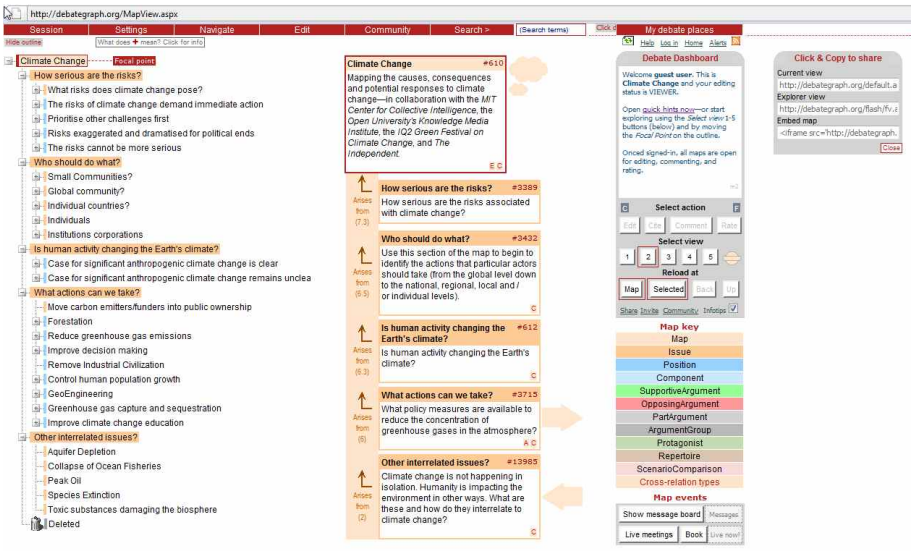


Fig. 2. The DebateGraph Climate Change argumentation map

To construct such a service, the initial focus is on the DebateGraph tool[7], as using this tool already quite a comprehensive map of the most important risks, causes, and effects is being produced[8], see Fig. 2. Furthermore, this map has already achieved considerable public visibility, as it has been embedded in the website of the UK quality newspaper The Independent [9] and as the DebateGraph tool has already been considered useful by the White House to map policy options relevant to the new U.S. administra-

[6] http://globalsensemaking.wik.is/ESSENCE/EVENTS/ESSENCE09_Workshop

[7] http://debategraph.org/

[8] http://debategraph.org/default.aspx?sig=610-610-2-0

[9] http://www.independent.co.uk/environment/climate-change/mapping-the-contours-of-climate-change-1640886.html

tion[10]. As such, the maps produced through this tool are prime candidates for exploring relevant argumentation mapping functionalities in a realistic context of creation and use.

2.2 Relevant Argument Map Generation: A Public Investigator Service

Assuming that using Debategraph, over time many climate change-related maps are being created. These would comprise the "reference map" currently being hosted at Debategraph and a number of related maps of particular interest to specific stakeholders. For example, one could be created by a EU policy making institute, one by the US government, one by a network of corporations interested in investigation carbon credit schemes, and so on. To be interoperable to at least some extent, all of these maps could be manually linked to relevant nodes of the reference map. However, to truly realize the policy making potential of these maps, relevant views need to be defined for *map users*.

When generating these views, rather than interpreting the content of the nodes, we could automatically interpret the argumentation *structure*, which has a restricted, yet well-defined and powerful semantics. When defining this restricted semantics and the knowledge operations based on them, the focus would be on the stakeholders, as this is what interests policy makers: what are the relative positions of organizations X, Y, and Z on the web of issues, positions, and arguments?

To illustrate what we mean, we introduce a "Public Investigator" service. Assume there is an investigative journalist who is interested in being kept up to date about controversial issues, which in our argumentation structure semantics could be defined as:

> "all the immediate *positions* on *issues* of which there is at least one *stakeholder* ("protagonist" in Debategraph terms) of a business-type that gives either a pro or a con *argument* and at least one *stakeholder* of the NGO-type who gives an *argument* with the _opposite_ *argument-relation* type (so, if the business stakeholder gives a pro-argument and the NGO-stakeholder a con-argument then we have a match, and thus a controversy, and vice versa)."

The resulting positions plus the relevant argumentation parts with their associated stakeholders could then be presented in a simple web page, with clickable links to supporting contact information and documentation. An example of an argument map that should match with this query is given in Fig. 3.

In the rest of the paper, we outline in broad strokes the theory behind and implementation of a service that could deliver such matches.

3 Using Conceptual Graphs to Analyze Argumentation Map Semantics

3.1 The IBIS formalism

At the core of our argumentation semantics is the Issue-Based Information Systems (IBIS) paradigm. The IBIS formalism [6, 8] has been reduced to practice by Jeff Con-

[10] `http://www.whitehouse.gov/blog/Open-Government-Brainstorm-Collaboration-in-Action/`

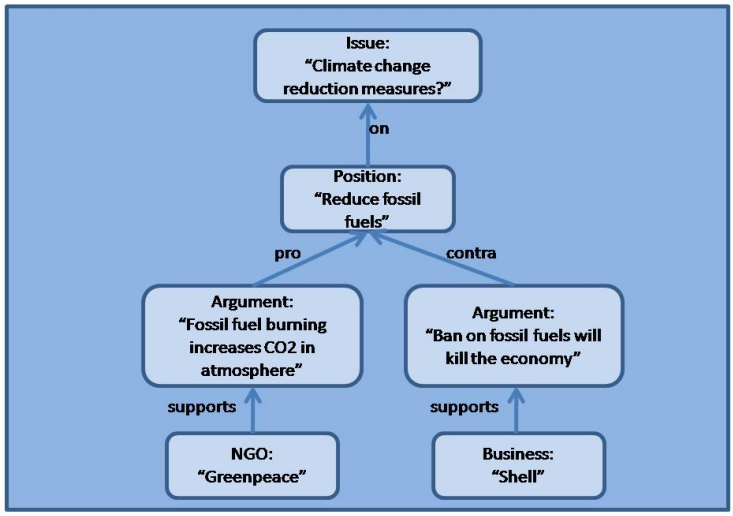


Fig. 3. An example of a matching argument map

klin [3]; a small ontology that defines node types and inter-node relationships is available and used in ESSENCE platforms such as Compendium[11], DebateGraph, and Deliberatorium[12]. The principal nodes of interest are as follows:

- *Map* - serves as an envelope that lends context to the other nodes, including map nodes.
- *Issue* - provides a container to ask a question and to explain the question with further details. These nodes can be answered by issue nodes that refine the question, or by Position nodes which provide answers.
- *Position* - provides a container to respond to a question with an answer, or to refine an earlier position.
- *Pro* Argument - a node that presents an argument that supports a position, together with details that explain the argument.
- *Con* Argument - a node that presents an argument that refutes a position, together with details that explain the argument.

The node types we present represent an IBIS ontology that has been found to be common among the three IBIS platforms mentioned above. Each platform is capable of providing other node types, some of which may enter the common ontology. For instance, a Reference node is useful in documenting a position or argument, though citations can presently be included in the details.

[11] `http://compendium.open.ac.uk/institute/`
[12] `http://18.36.1.44:8000/ci/login`

A typical IBIS conversation can begin with either an issue declaration, an opening question that sets the context for the remainder of the conversation, or the conversation can be opened with a topic for discussion, typically represented as a position node. An opening topic invites multiple issues to be raised. For instance, a conversation could begin with an issue such as "What is the role of carbon dioxide in climate change?", which defines a narrow context; a different conversation could simply begin by stating that "Climate Change" is the topic of the conversation, which would invite questions along the lines of causes, consequences, and so forth.

In a typical IBIS conversation in DebateGraph, IBIS statements take the form of short natural language sentences. The IBIS formalism adds context to those statements through the structure the conversation takes.

3.2 The CG formalism

Conceptual Graphs were introduced by Sowa (cf. [9, 10]) as a diagrammatic system of logic with the purpose "to express meaning in a form that is logically precise, humanly readable, and computationally tractable". In this paper we use the term "Conceptual Graphs" to denote the *family of formalisms* rooted in Sowa's work and then enriched and further developed with a graph-based approach in [2].

Conceptual Graphs encoded knowledge as graphs and thus can be visualized in a natural way:

- The vocabulary, which can be seen as a basic ontology, is composed of hierarchies of concepts and relations. These hierarchies can be visualized by their Hasse diagram, the usual way of drawing a partial order.
- All other kinds of knowledge are based on the representation of entities and their relationships. This representation is encoded by a labeled graph, with two kinds of nodes, respectively corresponding to entities and relations. Edges link an entity node to a relation node. These nodes are labeled by elements of the vocabulary.

The **vocabulary** is composed of two partially ordered sets: a set of concepts and a set of relations of any arity (the arity is the number of arguments of the relation). The partial order represents a specialization relation: $t' \leq t$ is read as "t' is a specialization of t". If t and t' are concepts, $t' \leq t$ means that "every instance of the concept t' is also an instance of the concept t". If t and t' are relations, then these relations have the same arity, say k, and $t' \leq t$ means that "if t' holds between k entities, then t also holds between these k entities").

A **basic graph** (BG) is a bipartite graph: one class of nodes, called *concept* nodes, represents entities and the other, called *relation* nodes represents relationships between these entities or properties of them. A concept node is labeled by a couple $t : m$ where t is a concept (and more generally, a list of concepts) and m is called the marker of this node: this marker is either the generic marker, denoted by $*$, if the node refers to an unspecified entity, otherwise this marker is a specific individual name. BGs are used to represent assertions called *facts*. They are also building blocks for more complex kinds of knowledge (such as rules, or nested graphs). In this paper we only detail rules as they are of direct interest to the framework we are proposing.

A **rule** expresses implicit knowledge of form "if *hypothesis* then *conclusion*", where hypothesis and conclusion are both basic graphs. Using such a rule consists of adding the conclusion graph (to some fact) when the hypothesis graph is present (in this fact). There is a one to one correspondence between some concept nodes of the hypothesis with concept nodes of the conclusion. Two nodes in correspondence refer to the same entity. These nodes are said to be *connection nodes*. The knowledge encoded in rules can be made explicit by applying the rules to specific facts.

These graphical objects are provided with a **semantics in first-order-logic**, defined by a mapping classically denoted by Φ in conceptual graphs [10]. First, a FOL language corresponding to the elements of a vocabulary $\mathcal{V}$ is defined: concepts are translated into unary predicates and relations of arity k into predicates of arity k. Individual names become constants. Then, a set of formulas $\Phi(\mathcal{V})$ is assigned to the vocabulary. These formulas translate the partial orders on concepts and relations: if t and t' are concepts, with $t' < t$, one has the formula $\forall x(t'(x) \rightarrow t(x))$; similarly, if r and r' are k-ary relations, with $r' < r$, one has the formula $\forall x_1 \ldots x_k(r'(x_1 \ldots x_k) \rightarrow r(x_1 \ldots x_k))$. A fact G is naturally translated into a positive, conjunctive and existentially closed formula $\Phi(G)$, with each concept node being translated into a variable or a constant: a new variable if it is a generic node, and otherwise the constant assigned to its individual marker. The logical formula assigned to a rule R is of form $\Phi(R) = \forall x_1 \ldots x_p\ ((hyp) \rightarrow \exists y_1 \ldots y_q\ (conc))$, where: hyp et $conc$ are conjunctions of atoms respectively translating the hypothesis and the conclusion, with the same variable being assigned to corresponding connection nodes; $x_1 \ldots x_p$ are the variables assigned to the concept nodes of the hypothesis; $y_1 \ldots y_q$ are the variables assigned to the concept nodes of the conclusion except for the connection nodes.

More importantly, first order logic subsumption can also be translated in a graphical operation: homomorphism. A homomorphism from G to H is a mapping between the node sets of G to the node sets of H, which preserves the adjacency between nodes of G and can decrease the node labels. If there is a homomorphism (say π) from G to H, we say that G *maps* to H (by π).

3.3 Linking IBIS and CGs

In this section we present the advantages of using Conceptual Graphs to complement the IBIS architecture. Intuitively speaking, the semantic network generated by IBIS does not differentiate amongst any of its nodes / relations. We will try to address this shortcoming by using Conceptual Graphs. However, this is not a trivial problem given the nature of "inference" (where by inference we mean making implicit facts explicit in the domain of interest) used by argumentation maps and by Conceptual Graphs.

Conceptual Graphs, in a first step, will provide a vocabulary for the types of nodes / relations of this semantic network. Indeed, what we propose is to organise the principal nodes of interest in a concept hierarchy as follows: the *Map, Issue, Position* and *Argument* nodes will be direct subconcepts of the node *Top*, while *Pro Argument* and *Con Argument* will be subconcepts of *Argument*. While debatably, on a first glance, this can only provide us with a quicker indexing method, these concept types will also be taken in consideration by relation signatures. For instance, the relation *argumentPro* will have *(Pro Argument, Position)* as signature. Furthermore, this pre order is easily

extended with different types of concepts / relations as need may be. In the context of a specific domain the concept hierarchy will also be enriched with the different roles that the "actors" participating at the debate will play (e.g. non- profit organisation, company, etc.)

Second, the projection will provide us with a basic mechanism for performing reasoning. As already mentioned in the previous section we can search for:

> "all the immediate *positions* on *issues* of which there is at least one *stakeholder* ("protagonist" in Debategraph terms) of a business-type that gives either a pro or a con *argument* and at least one *stakeholder* of the NGO-type who gives an *argument* with the _opposite_ *argument-relation* type (so, if the business stakeholder gives a pro-argument and the NGO-stakeholder a con-argument then we have a match, and thus a controversy, and vice versa)."

Please note the use of "immediate" in the context of the previous query. The rationale for using "imediate" is the fact that if we have a long chain of arguments which are pro each other (p_1 *supports* p_2 ... *supports* p_n) we can deduce the transitive closure of all these arguments. This is no longer true for instance when the relation between arguments is *"contradicts"*. Moreover, if we have p_1 *supports* p_2 which *contradicts* p_3, or we p_1 *contradicts* p_2 which *supports* p_3 we cannot deduce anything using Conceptual Graphs projection either. However, potentially, all these arguments could be useful in the context of the query mentioned above. This is why, in a first step we will only address immediate positions on issues. Heuristics are to be developed that will integrate first order logic subsumption with argumentation scheme manipulation.

4 A Proposal for the ESSENCE Argumentation Map Generator

As already explained, a missing piece of the puzzle so far has been a sophisticated semantic analysis service within and across these maps. This service could be based on, for instance, the Debategraph node/link primitives and possibly (within bounds) an analysis of the content of such nodes. The LIRMM conceptual graphs tools, in particular the reasoning engine Cogitant[13], would be perfectly capable of doing this. The problem is that so far, no hosted web service is available for manipulating (representation and / or reasoning with) conceptual graphs, but this could be realized at a relatively short notice.

4.1 Objectives

What we envision is using:

1. Debategraph to host a collection of growing argumentation maps
2. A well-designed service in which these argumentation maps are created and used by relevant stakeholders such as policy makers
3. CG analysis to generate new maps (subsets of original maps) based on user queries

[13] Available at: `http://cogitant.sourceforge.net/`

4. The results of these queries to be presented in a useful (Web page) format so that users can benefit best from the results.

This could be operationalized by having Debategraph send a set of queries with relevant parts of its argument maps to the Cogitant engine. This CG engine would process those queries, and return the results to Debategraph, which could then render the new, customized maps that are partial views highly relevant to *particular* stakeholders. Depending on their preference, the stakeholders would never even know that CGs have been used to help generate relevant views on the knowledge contained in the potentially many and very large maps having been created. With CGs, it is possible to create the highly customized and relevant map views, which can then be displayed in various formats.

4.2 The Argumentation-Conceptual Graph Converter

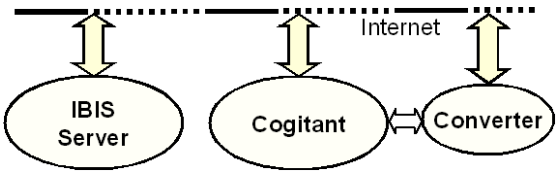


Fig. 4. IBIS Converter Extension to Cogitant

The prototype converter is planned to be a plug in agent for Cogitant as illustrated in Figure 4. In another ESSENCE-related project one of the authors is creating an IBIS serialization format that allows IBIS conversation platforms, including Compendium, DebateGraph, Deliberatorium, and Cohere from the Knowledge Media Institute's hypermedia discourse research [1], to share IBIS documents (conversations) with each other. It is that serialization format that will be read by the converter. The serialization format represents a subset of the XML Document Type Definition (DTD) used by Compendium. The serialized nodes are those listed in the Section above. All IBIS conversations persisted by the IBIS Server are thus available to the converter agent. The Converter serves as an agent to interact through Web services calls to the IBIS Server; fetching new IBIS conversations as they appear, and converting them to CG documents internal to Cogitant. The IBIS server provides an XML serialization that represents nodes and links from the IBIS conversation. As a simple example, a fragment of a serialization of the IBIS graph illustrated in the fragment below, where protagonist relations are modeled in the link structure:

```
<nodes>
<node id="1000" type="question" label="Climate change reduction measures?" />
<node id="1001" type="answer" label="Reduce fossil fuels" />
<node id="1002" type="pro" label="Fossil fuel burning increases CO2 in
 atmosphere" />
<node id="1003" type = "con" label="Ban on fossil fuels will kill the economy" />
<node id="1004" type="answer" label="NGO: Greenpeace" />
```

```
<node id="1005" type="answer" label="Business: Shell" />
</nodes>
<links>
<link id="2100" type="answers" label="Responds To" fromId="1001"
 toId="1000" />
<link id="2101" type="pro" label="Argues For" fromId="1002"
toId="1001" />
<link id="2102" type="con" label="Argues Against" fromId="1003"
toId="1001" />
<link id="2103" type="protagonist" label = "Supports" fromId="1004"
toId="1002" />
<link id="2104" type="protagonist" label="Supports" fromId="1005"
toId="1003" />
</links>
```

Operational steps presume that the IBIS Server is aggregating IBIS conversations from a variety of sources:

- Cogitant, through its IBIS Converter, acquires IBIS conversations through Web services calls to IBIS Server, and builds a CG database from those maps. Conversations are acquired in the XML form as illustrated above, and are then mapped to conceptual graphs.
- Cogitant receives queries submitted by Web services calls to the IBIS Converter and processes them as described below in Section 4.2

4.3 The Argumentation Map Generator Service

The envisaged functionality for Cogitant is to act as a stand alone web service. Once this is accomplished the query obtained from the IBIS Web Server will be run against the knowledge base and relevant projections will be retrieved. From an expressivity view point the query answers will allow for the users to retrieve certain parts of the maps they are interested in. These projections will be mapped into relevant subsets of the IBIS map and then rendered, for example in Debategraph, to the user.

As mentioned before if we are searching for issues and positions which are not "immediate" one to another certain manipulations should be performed in order to integrate the argumentation scheme with first order logic subsumption. These heuristics will also be implemented at this level, leaving projection to act on its own as a basic CG operation.

5 Discussion and Conclusions

Argumentation maps are visual representations of argumentation structures, making it possible to efficiently examine the cumulative results of protracted, distributed, and complex argumentation processes. Argumentation mapping tools could play an important role in improving the quality of societal debate and policy making processes, for example in the key domain of climate change. By generating relevant views on large argumentation maps, stakeholders can be helped in making better sense of new issues, positions, and arguments that pertain to their interests. The ESSENCE project investigates how such systems of tools in realistic contexts of use can be applied.

Conceptual graphs tools can be instrumental in performing the semantic analysis and reasoning needed for producing such relevant argumentation maps out of large argumentation knowledge bases. To show proof of concept, we outlined an architecture of a Public Investigator service built on a combination of DebateGraph, an argumentation support tool, and Cogitant, a concepual graphs engine.

What we propose is only a small step on a long road of research and development. Some directions for improvement:

- In our example, we only used a few argumentation semantics primitives: maps, issues, positions, arguments, and stakeholders. However, there are many more primitives, some only supported by specific tools. By including richer sets of argumentation primitives, more advanced services can be developed.
- Most argumentation supprort focuses on a single map at a time. However, in realistic usage scenarios, hundreds or thousands of related maps may need to be combined. How to classify the maps and their linkages is still very much an open question.
- In this paper, we only looked at the semantics of the argumentation structure, considering the semantics of the node contents a black box. Using the meaning of these nodes as well, would allow for more advanced argumentation support services. However, the natural language processing issues involved are still daunting and would require much more research.
- Many argumentation support tools are often considered in isolation. Still, in the real world, such tools are parts of much larger tool systems, including wikis, blogs, and social network sites such as Facebook. We are still only at the beginning of understanding technical, semantic, and pragmatic interoperability issues of such tool systems in their many different usage contexts
- We gave an example of one possible service, a "Public Investigator"-service. Many other (types of services) are conceivable. However, before we can implement these, basic argumentation semantics interoperability issues such as alluded to in this paper need to be resolved first.

Both argumentation mapping and conceptual structures tools are coming of age. The need for sophisticated semantics-based argumentation support is greater than ever with the many wicked problems threatening our global society. The ESSENCE project can be a testbed for their alignment and integration. We start with only one tool from each class, providing a relatively simple argumentation map generation service. In future work, this proof of concept will be expanded in many different directions. Through their combined forces, argumentation tools meeting semantic analysis tools can at last start delivering their fascinating potential on a very large scale.

References

1. S. Buckingham Shum. Hypermedia discourse: Contesting networks of ideas and arguments. In L. N. in Computer Science, editor, *Conceptual Structures: Knowledge Architectures for Smart Applications*, volume 4604, pages 29–44. Springer, 2007.
2. M. Chein and M. Mugnier. *Graph-based Knowledge Representation: Computational Foundations of Conceptual Graphs*. Springer, 2009.
3. J. Conklin. *Dialogue Mapping: Building Shared Understanding of Wicked Problems*. Wiley, 2005.
4. A. de Moor. Making ESSENCE work, presentation at the First ESSENCE workshop, May 5-6, 2009 KMi, The Open University, Milton Keynes, UK, http://www.slideshare.net/ademoor/making-essence-work.
5. M. Gurstein. Effective use: a community informatics strategy beyond the digital divide. *First Monday*, 8(12), 2003.
6. W. Kunz and H. Rittel. Issues as elements of information systems. Technical report, Institute of Urban and Regional Development, University of California, Berkeley, 1970.
7. A. D. Moor. The pragmatic evaluation of tool system interoperability. In *Proceedings of the Second Conceptual Structures Tool Interoperability Workshop (CSTIW 2007)*, pages 1–19, Sheffield, UK, 2007.
8. H. Rittel and M. Webber. Dilemmas in a general theory of planning. *Policy Sciences*, 4:155–169, 1973.
9. J. F. Sowa. Conceptual Graphs. *IBM Journal of Research and Development*, 1976.
10. J. F. Sowa. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, 1984.

Data Conversion and Interoperability for FCA

Simon Andrews

Communication and Computing Research Centre
Faculty of Arts, Computing, Engineering and Sciences
Sheffield Hallam University, Sheffield, UK
s.andrews@shu.ac.uk

Abstract. This paper proposes a tool that converts non-FCA format data files into an FCA format, thereby making a wide range of public data sets and data produced by non-FCA tools interoperable with FCA tools. This will also offer the power of FCA to a wider community of data analysts. A repository of converted data is also proposed, as a consistent resource of public data for analysis and for the testing, evaluation and comparison of FCA tools and algorithms.

1 Introduction

Most tools for Formal Concept Analysis require data to be in a particular format, usually representing a formal context and/or concept lattice. Unfortunately, most publicly available data sets, and most of the data produced by non-FCA applications, is not in this format. To make them interoperable with FCA tools, they need to be converted. Furthermore, there is a variety of data set formats and data types, each requiring different treatment. Converting data sets for FCA can be a time consuming and awkward task. A further problem arises in that a particular data set may be interpreted in different ways, resulting in inconsistent conversions. This can lead to different analyses of the same data and can make the comparison of FCA tools and algorithms more difficult. These problems have been pointed out by this author [2] and by Kuznetsov and Ob"edkov [5]:

> "We would like to propose the community to reach a consensus w.r.t. databases to be used as testbeds. Our idea is to consider two types of testbeds. On the one hand, some "classical" (well-recognised in data analysis community) databases should be used, with clearly defined scalings if they are many-valued. On the other hand, we propose to use "randomly generated contexts"... The community should specify particular type(s) of random context generator(s) that can be tuned by the choice of ... parameters."

This paper, therefore, has two proposals to improve the interoperability and consistency of use of non-FCA format data sets:

1. A 'To-FCA Format' Data Converter, that will provide FCA practitioners with an efficient and consistent means of converting a range of non-FCA data set formats into a format suitable for FCA.
2. An FCA Data Repository, that will provide FCA practitioners with a resource of public data sets in FCA format.

2 Data Conversion

Public data set repositories, such as the UCI Machine Learning Repository [3] and Amazon Public Data Sets [1], provide a useful resource for FCA. Several data sets from the UCI Repository have become familiar to FCA. Four of these are listed in Table 1 and are useful to illustrate issues in data conversion. The table lists the name of the data set, its format, number of objects, number of attributes, the data type/s of the attributes and the number of attributes once converted into a formal context. Some of these have a question mark indicating that there are different interpretations possible. For example, the Mushroom data set has been quoted variously as having 125 [2], 119 [4] and 120 [9] attributes.

Table 1. Some UCI Repository Data Sets

Name	Format	Objs	Atts	Att Type	FCA Atts
Mushroom	data only	8124	22	Categorical	125?
Adult	data only	48842	14	Categorical, Integer	96?
MS Web	DST Sparse	32711	294	N\A	294
Internet Ads	data only	3279	1558	Real, Relational	1555?

There are several issues to consider when converting data sets into a formal context:

Data Set Format Different 'standard' data set formats require different treatment. Some contain data values only, others include additional information, such as the names of the attributes or the classes of objects. Many contain data in tabular form, with rows representing objects and columns representing attributes. The type of the attribute (categorical, integer, real or relational) will determine the conversion technique required. Many data sets are multivariate, having a mixture of attribute types. Some data sets do not have a tabular format, such as the DST Sparse Data Format and Sparse ARFF, where relational data is represented by attribute/object pairs; clearly suitable for FCA, but requiring a different method of conversion. Similarly, RDF files [6] would require a different method of conversion, where an RDF subject becomes an FCA object. Databases are another important consideration; RDBMS data files would require a significantly different approach to the treatment of flat-file data.

Categorical Attributes Categorical (many valued) attributes are the most common type and can be converted by creating a formal context attribute for each of the values. The attribute *cap-surface*, for example, in the Mushroom data set, has four values: *fibrous*, *grooves*, *scaly* and *smooth*. In a formal context, this becomes four attributes: *cap-surface_fibrous*, *cap-surface_grooves*, *cap-*

surface_scaly and *cap-surface_smooth*. However, different interpretations are possible if an attribute has only two values; it may be said that not having one value implies having the other, thus leading to a single attribute in the formal context. This is particularly so if the values are opposites. For example, in the Mushroom data set, there is an attribute called *bruises?* that has values *bruises* and *no*. Should this be interpreted as a single attribute, *bruises*, or two: *bruises?_bruises* and *bruises_no*? If, for a particular attribute, there is no object with a particular value, or all objects have the same value, how should this be interpreted? In the Mushroom data set, for example, none of the mushrooms has a universal veil; all have a partial veil. Should the veil attribute be ignored or interpreted as one or two attributes?

Other Attribute Types A table of Boolean values is used by some data sets to indicate the relationship between attributes and objects, such as the majority of the data in the Internet Ads data set. Such data can be translated, one-to-one, into a corresponding context. Attributes with integer or real types are less easily dealt with. Should they be ignored or should some form of scaling be used to create context attributes with ranges of values? Attributes that have free text values (a person's address, for example) are the least convertible and will almost certainly be omitted in a conversion.

Missing Values Many data sets contain missing values. Should they be treated as a 'has not' relationship or should objects with missing values be removed?

Classes Some data sets contain classes of data. Should these be ignored in the conversion, treated as attributes, or should a separate context be created for each class?

Clearly, a useful tool for data conversion must deal with all of these issues.

2.1 A Conversion Example

A simple example will help illustrate some of the issues outlined above, and visualise possible input and output files of an FCA data converter. Figure 1 is a miniature version of the UCI Mushroom data file, `mushroom.data`; a data only flat-file of comma separated values, and a format very suitable for input to an FCA data converter. The first column gives the mushroom class, the other four are mushroom attributes. Each is nominally valued in the following way:

column 1 class: edible = e, poisonous = p
column 2 bruises?: bruises = t, no = f
column 3 gill-size: broad = b, narrow = n (missing value = ?)
column 4 veil-type: partial = p, universal = u
column 5 ring-number: none = n, one = o, two = t

```
e,t,b,p,n
e,t,n,p,t
p,f,n,p,n
e,t,?,p,o
p,f,n,p,n
```

Fig. 1. Miniature version of `mushroom.data`

Figure 2 is an interpretation of the mushroom data as a formal context. The following decisions have been made in the interpretation:

- The mushroom class is not used.
- The attribute 'bruises?' is a single formal attribute.
- All other attributes have a formal attribute for each of their possible values.

Mushroom	bruises	gill-size-broad	gill-size-narrow	veil-type-partial	veil-type-universal	ring-number-none	ring-number-one	ring-number-two
mushroom1	×	×		×		×		
mushroom2	×		×	×				×
mushroom3			×	×		×		
mushroom4	×			×			×	
mushroom5			×	×		×		

Fig. 2. Mushroom context

Figure 3 is the Mushroom context in the Burmeister cxt file format, a common FCA context format used by a number of FCA tools and one that could be output from an FCA data converter. The 'B' at the start of the file possibly stands for 'Burmeister'; it appears to be a tradition of the format! The numbers that follow are the number of objects and the number of attributes, respectively. The names of the objects and attributes are then listed, followed by the incidence vectors of the objects, consisting of dots and 'X's.

How the object and attribute names are obtained will be a factor in the design of an FCA data converter, and how far this process can be automated will need to be considered.

```
B

5
8

mushroom1
mushroom2
mushroom3
mushroom4
mushroom5
bruises
gill-size-broad
gill-size-narrow
veil-type-partial
veil-type-universal
ring-number-none
ring-number-one
ring-number-two
XX.X.X..
X.XX...X
..XX.X..
X..X..X.
..XX.X..
```

Fig. 3. `mushroom.cxt`

3 Proposal 1: A 'To-FCA Format' Data Converter

Figure 4 is a simple, high-level view of the proposed 'To-FCA Format' data converter. The proposed output of the converter is a file in the Burmeister cxt format.

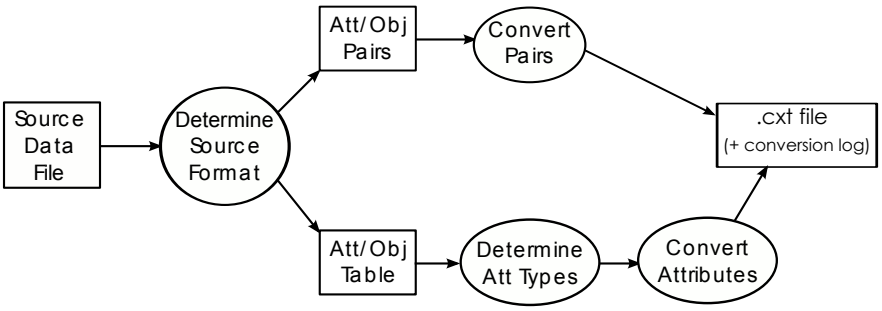


Fig. 4. Proposed 'To-FCA' Data Converter

After selection of the source data file format, the main process is determined by whether the data is in the form of attribute/object pairs or in the form of a table. In the latter case, the tool must carry out the appropriate conversion depending on the type of each attribute. Information will be required by the tool,

concerning the number and names of objects and attributes, the type of each attribute and its categories, if appropriate. This information may be obtained by the tool from the source data file, or from the user, depending on the source format, and will be output in a conversion log, along with information regarding decisions made in the conversion process, such as any original attributes not converted.

There are a number of FCA context formats, other than cxt, used by a variety of FCA tools and applications. The data converter could be expanded to output these formats, too, but FcaStone[1] [7, 8], already exists that easily converts one commonly used FCA file type into another. The proposed data converter could integrate with FcaStone, making non-FCA format data interoperable with a much wider range of FCA tools and applications (Figure 5). In conjunction with Graphviz[2], an open-source graph visualisation tool, FcaStone can produce a range of graph formats for the production of concept lattices, thus expanding further the range of FCA tools interoperable with the original data.

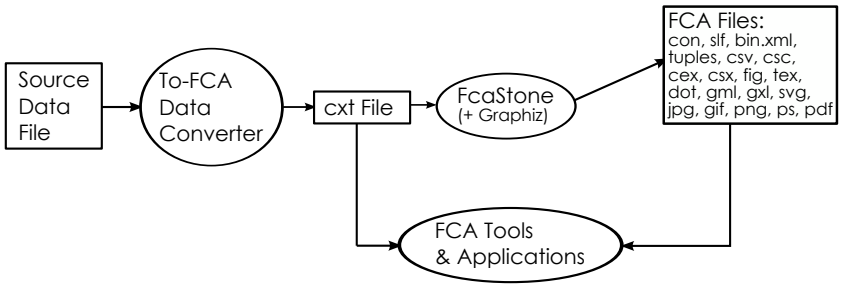


Fig. 5. Possible Integration of 'To-FCA' Data Converter and FcaStone

4 Proposal 2: An FCA Data Repository

Figure 6 is a diagram of the proposed, web-based, FCA data repository. It is proposed that the data converter tool and FcaStone are incorporated into the repository allowing users to convert data sets as well as to access the stored converted data sets. Users will also be able to donate data sets. A converted data set will be stored along with, where possible, its original data file, information about the original data (probably from the original data source), a link back to the original data source, the conversion log, and FCA information, such as context density and number of concepts.

To address needs outlined in the introduction of this paper, the repository will also provide access to stored random data and incorporate a random data

[1] FcaStone: http://fcastone.sourceforge.net
[2] Graphviz: http://www.graphviz.org

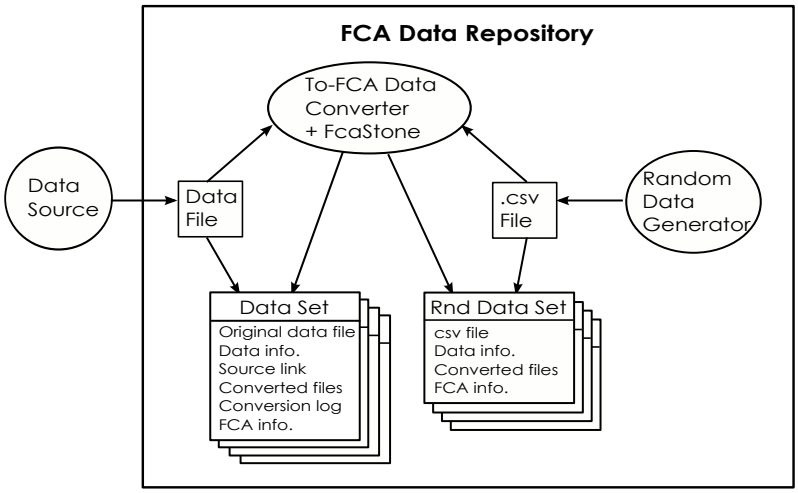


Fig. 6. Proposed FCA Data Repository

generator, the initial output of which will be a file of comma-separated object number, attribute number pairs. The csv file can then be converted into the required format. The user will be able to determine the number of attributes, number of objects and the density. It is proposed that the user will also select one of a small range of random number generator seeds; thus the same data set will be generated by any given seed/data parameters.

In this way, the repository can act as a bench-marker for the comparison of tools and algorithms by providing citeable random data as well as converted 'real' data sets.

5 Development of the Proposals and Conclusion

The To-FCA data converter will be developed as an open-source software. An initial prototype is under development, converting 'vanilla' data sets (such as the UCI Mushroom data set) and should be ready as a demonstrator tool in June/July 2009. A Link has been formed with the JISC Information Environment Demonstrator Project[3] to provide possible dissemination vehicles for the development. It is also hoped that participation in an ICCS 2009 workshop comparing the performance of FCA algorithms[4] will provide useful steering regarding data formats.

The UCI Machine Learning Repository has kindly given the author permission to reformat and make their data sets available online. The FCA data

[3] JISC IE Demonstrator Project: http://www.jisc.ac.uk/whatwedo/programmes/reppres/iedemonstrator.aspx

[4] Comparing performance of FCA algorithms: http://iccs09.org/forum

repository is likely to initially take the form of a web service offering a small selection of converted UCI data sets. The addition of random data sets and the incorporation of tools will be an incremental development. If data sets are to be encouraged from donors, a repository librarian will be required to validate them, ensuring that they are in required format and that the necessary (and verifiable) supporting information is provided.

The final vision is of an interactive set of web-services, offering FCA tools interoperability with a wide range of data, providing a resource of useful collections of 'real' and random data sets in a wide variety of FCA context and lattice formats, and offering users the facility to create, convert and donate data sets of their own. Conversion between non-FCA and FCA formats will also open a way to the wider use of FCA, offering its power to those currently outside the FCA community.

References

1. Amazon Web Services: Public Data Sets on AWS [http://aws.amazon.com/publicdatasets/] (2009)
2. Andrews, S.: In-Close, a Fast Algorithm for Computing Formal Concepts. To be presented at the Seventeenth International Conference on Conceptual Structures (2009).
3. Asuncion, A., Newman, D. J.: UCI Machine Learning Repository [http://www.ics.uci.edu/∼mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science (2007).
4. Krajca, P., Outrata, J., Vychodil, V.: Parallel Recursive Algorithm for FCA. In: Belohlavek, R., Kuznetsov, S.O. (eds.), *Proceeding of the Sixth International Conference on Concept Lattices and their Applications*, pp. 71-82, Palacky University, Olomouc (2008).
5. Kuznetsov, S.O., Ob"edkov, S.A.: Comparing Performance of Algorithms for Generating Concept Lattices. In: *Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 14, pp. 189-216 (2002).
6. Passin, T. B.: Explorer's Guide to the Semantic Web. Manning Publications Co., Greenwich, CT 06830, USA (2004).
7. Priss, U.: FcaStone - FCA File Format and Interoperability Software. In: Croitoru, M., Jaschkë, R., Rudolph, S. (eds.), Conceptual Structures and the Web, *Proceedings of the Third Conceptual Structures and Tool Interoperability Workshop*, pp. 33-43 (2008).
8. Priss, U: FCA Software Interoperability, In: Belohlavek, R., Kuznetsov, S. O. (eds.) *Proceeding of the Sixth International Conference on Concept Lattices and Their Applications*, pp. 133-144 (2008).
9. Wang, J., Han, J., and Pei, J.: CLOSET+: searching for the best strategies for mining frequent closed itemsets. *Proceedings of the Ninth ACM SIGKDD international Conference on Knowledge Discovery and Data Mining*, pp. 236-245, ACM, New York (2003).

Building Conceptual Graphs for Articles Abstracts in Digital Libraries

Michael. Y. Bogatyrev[1], Olga A. Mitrofanova[2], Vitaliy V. Tuhtin[1],

[1] Tula State University, Lenin ave 92, 300600 Tula, Russia
[2] St. Petersburg State University, University emb. 11, 198134 St. Petersburg, Russia

Abstract. An approach which is based on joining acquisition of conceptual graphs and analyzing them for further implementations is presented. The goal of conceptual graphs analysis is to find informative conceptual graphs for storing them in digital library. This analysis is based on learning links between sentences in abstracts by using two instruments: conceptual graphs clustering and passive voice grammar form as a possible pointer to the link between sentences. All results of acquisition of conceptual graphs and their clustering can be visually illustrated using modeling framework.

Keywords: knowledge representation, conceptual graphs, clustering, visualization.

1 Introduction

In Digital Libraries conceptual graphs (CGs) [14], [15] become a competent part of stored data together with library texts. Conceptual graphs may serve as data for creating more general structures, for example ontologies, to form knowledge bases [11]. In this connection the problem of creating CGs automatically from library text is important. If CGs are acquired automatically then another problem of sufficient quantity of stored conceptual graphs appears. Actually, conceptual graph is a model of semantics of a sentence but not every sentence in library texts is informative for solving all these problems which use CGs.

In this paper an approach which is based on joining acquisition of conceptual graphs and analyzing them for further implementations is presented.

This approach is applied in digital library project devoted to store and process scientific papers. Conceptual graphs are acquired only from papers abstracts instead of processing the whole texts. This is based on the assumption that an abstract is short and clear essence of a paper.

There are two instruments for analysis links between sentences in an abstract discussed in the paper. The first is based on the assumption that passive voice grammar form in a sentence often serves as a pointer to another sentence. Some peculiarities of conceptual graph acquisition from sentences in passive voice are illustrated. The second way of checking links between sentences is realized by conceptual graphs clustering.

Both these ways are experimental and help an expert to select informative conceptual graphs for further implementations. Visualization of graphs and clusters is applied as crucial tool for making decision.

2 Conceptual Graphs in Digital Library Project

Digital libraries, unlike usual ones, are capable to present principally new functions to their users. Among them there is a function of knowledge representation when a user issuing a query to the library gets output as a structure of terms and notions related to that query. To support this function, a library must store and process some knowledge model. This high level model, for example ontology, represents knowledge inherent to library texts.

We will have genuine knowledge representation system when its high level knowledge model – ontology - can be created or at least corrected from low level models – for example from conceptual graphs. This problem of building ontologies automatically is very complicated, not yet solved and may be could not be solved completely. As it is shown in [5], it is possible to go quite far in knowledge representation by using conceptual graphs. So on the way of creating ontologies, conceptual graph may serve as elementary start semantic model needed to generate next level models.

Solutions of two standard problems of clustering and aggregation on conceptual graphs constitute principal foundation of creating next level models [2], [11]. Clusters of conceptual graphs correspond to possible topics of library texts and aggregated conceptual graphs represent a frame for library catalogue.

We develop the digital library research project [1] where conceptual graphs are applied. It is a library of scientific papers collecting data from two sources: papers in English language from the CiteSeer portal [13] and papers in Russian language from the portal of RCDL conferences [6]. The project is being realized as an open framework containing subsystem of conceptual graph acquisition from text. It also contains subsystem for conceptual graphs clustering.

2.1 Building conceptual graphs for abstracts

There are many approaches to conceptual graph acquisition. One can find references to existing approaches in [4], [8], [16]. To summarize, we highlight the following points of general algorithm of conceptual graph acquisition.

1. Lexical analysis of the source text for conceptual graph acquisition. Source text may be from natural language or domain specific text. If the last take place then domain lexicon is considered.
2. Morphological analysis of the source text which includes identification, analysis and description of the structure of words in sentences. Word formation paradigm from certain language is applied. Here concepts of conceptual graph have been created.
3. Semantic analysis. This point is mostly complicated and determines all variety of existing solutions. It is directed to create conceptual relations. As a

rule, conceptual relations are not represented by words from analyzing sentence. So external resources (such as WerbNet and WordNet) have been applied in analysis.

At first glance it seems that these three points of general algorithm can be applied to conceptual graph acquisition from a text in any language because they constitute standard grammar of language; in practice it is by no means so.

To acquire conceptual graph from a sentence we use existing approaches to realize lexical, morphological and semantic analysis. We apply Semantic Roles Labelling [7] as the main instrument for building relations. Some special decisions were made for texts in Russian language and discussed below.

For every article in the library only its abstract is processed for building conceptual graphs. This is reasonable since an abstract is short and clear essence of an article.

2.1.1. Classifying abstracts

It is evident that not every sentence in an abstract really contains actual information for processing in the library. So it is not needed to acquire conceptual graph from every sentence of an abstract. We classify all abstracts into two types: abstracts of *declarative* and *narrative* types. As a rule, declarative abstracts are short and contain weakly connected sentences. Oppositely, narrative abstracts are quite long and contain sequences of closely connected sentences.

The most of real abstracts are declarative. Narrative abstracts are rare and may contain *new ideas* expounded sequentially. But namely narrative abstracts most probably have "useless" sentences. "Useless" sentences are often isolated, i.e. semantically not connected with others.

To detect narrative abstracts we applied two criteria mentioned above: existing many sentences in the abstract – more than 6 (experimental result) and having high level of *semantic similarity* of sentences. We applied hierarchical clustering technique as experimental way to check links between sentences. The checking algorithm is the following.

1. Find a long abstract.
2. Acquire conceptual graphs from all sentences of an abstract.
3. Build a clustering for conceptual graphs.
4. Analyse the structure of clusters: CGs which constitute separate clusters are marked for further analysis as "useless".

The central problem in any clustering is a measure of similarity of clustering objects. We apply three similarity measures for conceptual graphs: two measures similar to Dice coefficients [2], [11] – conceptual and relational measures, and semantic measure similar to one applied in [16]. To calculate semantic measure we apply WordNet system for English texts.

Figure 1 demonstrates clustering of conceptual graphs for two abstracts. One of them is the abstract in reference 9 which is definitely narrative having 11 sentences (numbers 0 – 10 on Figure 1) and being as a little story. The second abstract is very different from the first one and is consisted of only one sentence constituting its own cluster.

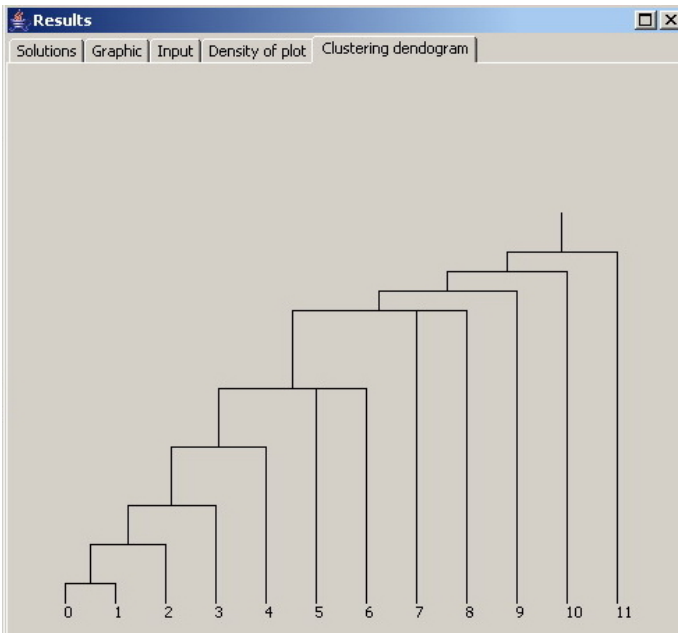


Fig. 1. Clustering dendrogram of conceptual graphs for two abstracts having 11 and 1 sentences and using relative similarity measure.

Clustering algorithm applied in the system uses special modified relative similarity measure based on the Dice coefficients. We detect *narrativeness* as specific nesting in clusters hierarchy shown on Figure 1. This kind of nesting with its slightly different visible variants is typical for all narrative abstracts. Nevertheless, clustering dendrogram could not directly help to find "useless" sentences in an abstract. For example the abstract from article [9] contains the sentence "*At such extremes our intuition breaks down*" (number 4 on the graph on Figure 1) which is not appropriate source for conceptual graph and was manually detected as "useless".

Using more detailed visualization which represents map of clusters we can find that this sentence is modeled by CG which has relations of *attribute*, *agent.* These relations occur in other CGs and therefore the first CG does not constitute the single cluster.

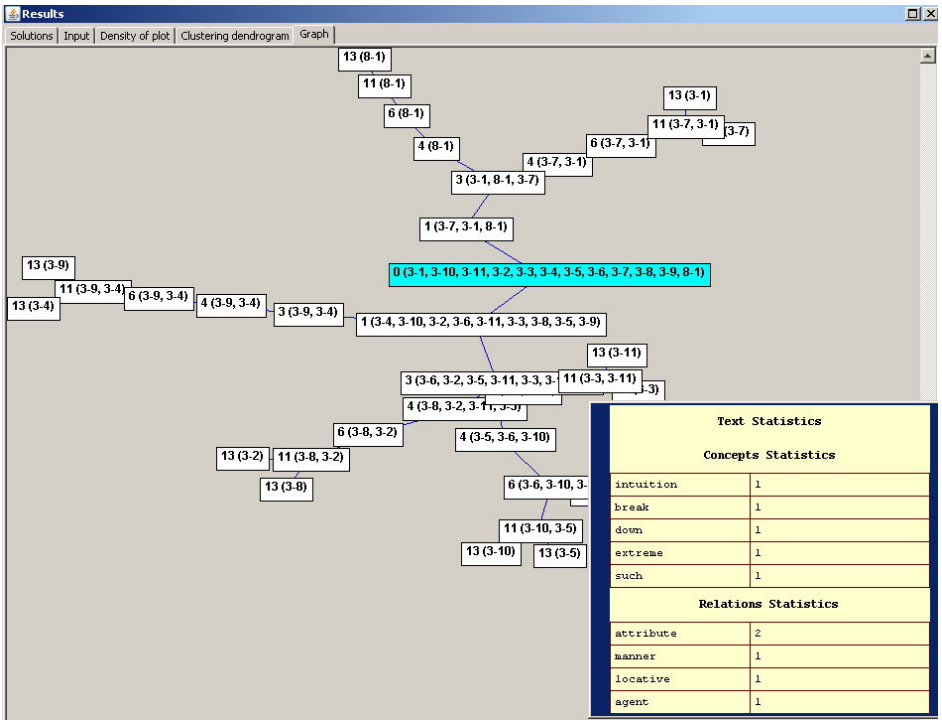


Fig. 2. Map of clusters with statistics of concepts and relations. Notation $n(m\text{-}k, \ldots, s\text{-}r)$ means that on the n – th step of clustering the cluster contains k –th sentence from m-th abstract and so on up to r– th sentence from s-th abstract.

2.1.2. Isolated concepts and multilingual passive voice problem

Very often a text of a sentence in an abstract contains no direct useful information but a reference to other sentences. This feature is often realized grammatically by passive voice. There are two ways of building conceptual graph for a sentence in passive voice: create conceptual graph for equivalent sentence in active voice or create special tool for representing passive voice. Since a sentence in passive voice may be a pointer to external objects – concepts or conceptual graphs, - we have selected the second way.

Working with passive voice in Russian and in English languages, one faces with many peculiarities of realizing passive voice in Russian language. English language, being more analytical than Russian language, has strict grammatical rules of expressing passive voice.

Using our algorithm of conceptual graphs acquisition to English sentences in passive voice we applied the following solution: resulting conceptual graph has isolated concept corresponded to the verb “to be”. So the isolated concept “to be” serves as an indicator of passive voice.

Unfortunately it is impossible to have such regular solution for texts in Russian language in passive voice. So we undertook special efforts to create algorithm of conceptual graphs acquisition for Russian sentences in passive voice. The instrument of Automated Lexical Analysis [10] was useful here. As a result an actor "passive" appears in the graphs denoting passive voice for Russian texts. Using visualization we can illustrate processing of sentences in passive voice by the following example. Consider two sentences about contemporary people: *"Adam gave the apple to Eve"* and *"The apple was given to Eve by Adam"*. Figures 3, 4 illustrate corresponding conceptual graphs in English and Russian languages.

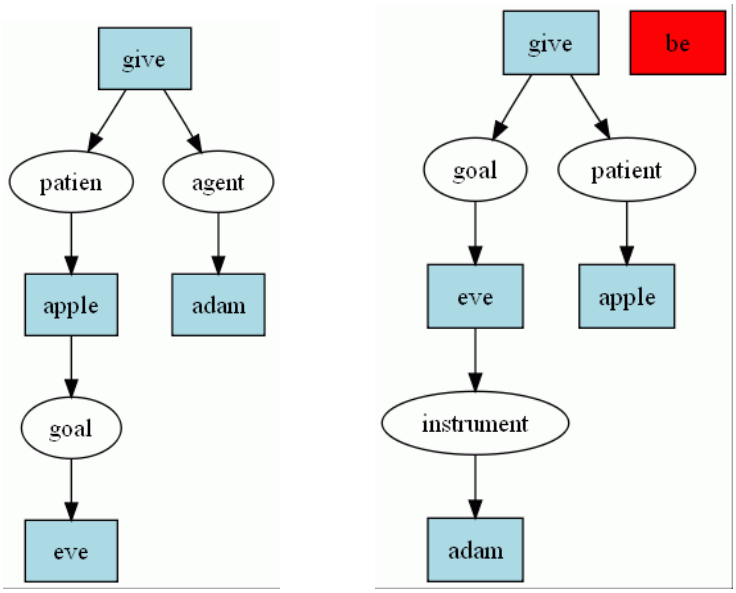
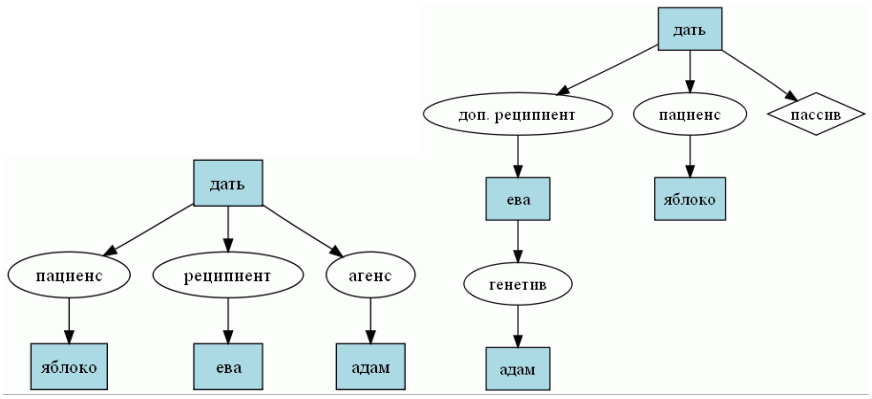


Fig. 3. Conceptual graphs for sentences in active (a) and passive (b) voice.

Conceptual graph on Figure 3 - b) seems not correct because actually Adam serves as an instrument for delivering apples to Eve. In Russian variant Adam and Eve are relatives.

For both languages we have special markers for passive voice – isolated concept "be" and an actor "passive". They can be exploited not only visually but also in programs and help to select CGs for storing in digital library.



a) b)

Fig. 4. Conceptual graphs for sentences in active (a) and passive (b) voice for Russian texts. Translations: дать = to give, яблоко = an apple, Ева = Eve, Адам = Adam, пациенс = patience, агенс = agent, реципиент = recipient, генетив = genitive, пассив = passive (actor).

3 Conclusion

We have presented approach to building conceptual graphs for abstracts of papers in digital library which uses analysis of abstract sentences. The goal of such analysis is to find informative conceptual graphs for storing them in digital library. We applied the framework for conceptual graphs acquisition and for clustering them. Presented results which illustrate proposed approach are preliminary. They may be expanded by new results of investigating various similarity measures in clustering problem and new results of learning *semantic expressiveness* of conceptual graphs against sentences they model.

References

1. Bogatyrev, M. Latov, V. Stolbovskaya, I.: Application of Conceptual Graphs in Digital Libraries. In: Digital libraries: Advanced Methods and Technologies, Digital Collections. Proceedings of the Ninth Russian National Research Conference - Pereslavl-Zalesskij: Pereslavl University. P.p. 109-115 (in Russian) (2007)
2. Bogatyrev, M.Y. Tuhtin, V.V.: Solving Some Text Mining Problems with Conceptual Graphs. In: Digital libraries: Advanced Methods and Technologies, Digital Collections. Proceedings of the Tenth Russian National Research Conference RCDL'2008 Dubna, JINR p.p. 31-36. (in Russian). (2008).
3. Bogatyrev, M.Y., Tuhtin, V.V.: Creating Conceptual Graphs as Elements of Semantic Texts Labeling. Computational Linguistics and Intellectual Technologies. Papers

from the Annual International Conference “Dialogue 2009”. Issue 8 (15). P.p. 31-37 (in Russian). (2009)
4. Boytcheva, S. Dobrev, P. Angelova, G.: CGExtract: Towards Extraction of Conceptual Graphs from Controlled English. Lecture Notes in Computer Science № 2120, Springer Verlag. (2001).
5. Chein, M., Mugnier, Marie-Laure: Graph-based Knowledge Representation. Computational Foundations of Conceptual Graphs. Series: Advanced Information and Knowledge Processing, Springer. (2008)
6. Digital libraries: Advanced Methods and Technologies, Digital Collections. Digital Collection of Proceedings: http://rcdl.ru/
7. Gildea D., Jurafsky D.: Automatic labeling of semantic roles. Computational Linguistics, 2002, v. 28, p.p. 245-288. (2002)
8. Hensman, S, Dunnion, J.: Automatically building conceptual graphs using VerbNet and WordNet. In: Proceedings of the 3rd International Symposium on Information and Communication Technologies (ISICT), Las Vegas, June 16-18, 2004, pp.115-120. (2004).
9. Horn, J, Goldberg, D.: Genetic Algorithm Difficulty and the Modality of Fitness Landscapes. In: Proc. of the Foundations of Genetic Algorithms (FOGA) 3 Workshop, 1994 Estes Park, Colorado, USA.
10. Mitrofanova, O. A., Zakharov V. P.: Automatic Analysis of Terminology in the Russian Text Corpus on Corpus Linguistics. Computational Linguistics and Intellectual Technologies. Papers from the Annual International Conference “Dialogue 2009”. Issue 8 (15) p.p. 321-328. (in Russian) . (2009).
11. Montes-y-Gomez, M., Gelbukh, A., Lopez-Lopez, M.: Text Mining at Detail Level Using Conceptual Graphs. Lecture Notes In Computer Science; Vol. 2393. P. 122 – 136. (2002)
12. Rajman, M., Besançon, R.: Text Mining - Knowledge extraction from unstructured textual data In: Proc. of 6th Conference of International Federation of Classification Societies (IFCS-98), p.p. 473 – 480. (1998).
13. Scientific Literature Digital Library and Search Engine: http://citeseer.ist.psu.edu/
14. Sowa, J.F.: Conceptual Graphs for a Data Base Interface. IBM Journal of Research and Development 20(4): 336-357 (1976)
15. Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, London, UK (1984)
16. Zhong, J., Zhu H., Li J., Yu Y.: Conceptual Graph Matching for Semantic Search. In: Proceedings of the 10th International Conference on Conceptual Structures: Integration and Interfaces, Springer-Verlag p.p. 92–106. (2002)

RDF to Conceptual Graphs Translations

Calculating Shatterproof Transponds

Jean Francois Baget[1,2], Michel Chein[2], Madalina Croitoru[2], Jérôme Fortin[2], David Genest[3], Alain Gutierrez[2], Michel Leclere[2], Marie-Laure Mugnier[2], and Éric Salvat[4]

[1] INRIA Sophia Antipolis, 2004 Route des Lucioles 06902 Sophia Antipolis, France
jean-francois.baget@inria.fr
[2] LIRMM (CNRS & Université Montpellier II), 161 rue Ada, F–34392 Montpellier Cedex 5, France
{chein, croitoru, fortin, gutierre, leclere, mugnier}@lirmm.fr
[3] LERIA, Université d'Angers, 2, boulevard Lavoisier, 49045, Angers, France
genest@info.univ-angers.fr
[4] IMERIR, Avenue Pascot, 66004 Perpignan, France
salvat@imerir.com

Abstract. In this paper we will discuss two different translations between RDF (Resource Description Format) and Conceptual Graphs (CGs). These translations will allow tools like Cogui and Cogitant to be able to import and export RDF(S) documents. The first translation is sound and complete from a reasoning view point but is not visual nor a representation in the spirit of Conceptual Graphs (CGs). The second translation has the advantage of being natural and fully exploiting the CG features, but, on the other hand it does not apply to the whole RDF(S). We aim this paper as a preliminary report of ongoing work looking in detail at different pro and the cons of each approach.

1 Introduction and motivation

In this paper we will discuss the different translations between RDF (Resource Description Format) and Conceptual Graphs (CGs). We aim this paper as a preliminary report of the ongoing work of the authors looking in detail at different problems raised by the translation. We will give the overview of two possible translations and explain the pro and the cons of each one of them.

The objective of this work is to have a detailed translation from RDF(S) to COGXML and vice-versa. This translation will allow tools like Cogui [5] and Cogitant [6] to be able to import RDF(S) documents and to export RDF(S) documents. The translation between RDF and Conceptual Graphs is an important problem to be addressed for both Conceptual Graphs and Semantic Web communities:

– For the **Conceptual Graphs** people there is an obvious interoperability benefit (adhering to a well known standard). One consequence of this benefit is the fact that large RDF(S) benchmarks are available, hence ready for use in the context of testing CG algorithms.

[5] http://www.lirmm.fr/cogui/
[6] http://cogitant.sourceforge.net/

– For the **Semantic Web** people
  - We provide an editor for RDF that performs representation and reasoning at the same time (this is not possible yet with the other editors).
  - Another benefit for Semantic Web people will the translation of RDF into a data structure that is relying on a support hence potentially improving the performances of the algorithms manipulating RDF (subsumption).
  - And finally, the RDF community could benefit from the ideas behind the CG extensions.

A first translation has been provided by Tim Berners Lee [1]. While an important step towards the RDF - CG translation this paper remains at an intuitionist level. More in detail the translation has been addressed by [2] or [3] but the different technical difficulties encountered have been addressed within an implementation oriented setup. The authors of [4] also address the problem of converting CGs to RDF but certain practical cases are not addressed in their framework. A more theoretical approach has been taken by [5] and this translation will be discussed further on in the paper.

This paper discusses two different RDF(S) - CG translations. The first translation, following the work of [5] is sound and complete from a reasoning view point but is not a visual representation or a representation in the spirit of Conceptual Graphs (the support is flat).

The second translation has the advantage of being natural and fully exploiting the CG features (for instance, the RDFS constructs subClassOf and subPropertyOf are translated into partial orders, which are automatically taken into account by projection). On the other hand, it does not apply to the whole RDF(S), but only to RDF(S) documents conforming to a strict separation between concepts (or classes), relations (or properties) and individuals. In other words, all RDF(S) triples leading to meta reasoning are out of the scope of this translation. Note that for a given RDF(S) document, there may be several maximal subsets of triples satisfying the property. A simple way of choosing one maximal subset is to process the triples in the given order and to discard triples that contradict the separation according to the triples already processed. The separability condition is in accordance with usual assumptions in knowledge representation and reasoning, for instance in description logics (see namely OWL-DL). Moreover, it seems that most RDF(S) documents fulfill this condition in practice (which remains to be experimentally checked).

On the other hand, the first translation allows to process the whole RDF(S), but it does not allow to fully benefit from CG features (for instance, the RDFS constructs subClassOf and subPropertyOf are translated into CG relations, and rules are necessary to ex- press the transitivity of these relations (cf. rules 5 and 11 in Figure 2), which leads to a loss in algorithmic efficiency).

2 Preliminary notions

In this section, we will quickly recall the main basic structures of RDF(S) and Conceptual Graphs (CGs) in order to have a coherent notation throughout the paper. Since in this paper we will focus on the translation between RDF(S) and will first present RDF(S) and then CGs.

2.1 RDF(S)

General notions The Resource Description Framework (RDF) has been defined by the World Wide Web Consortium (W3C) as a metadata data model. It allows to make statements about resources in the form of subject–predicate–object expressions called RDF triples. The subject denotes a resource, the predicate gives the relationship between the subject and the object. Any RDF statement can be both stored in a XML format or in Notation 3 (or N3). A set of RDF statements can be dispayed by a graph.

Syntax The subject of an RDS statement can be a Uniform Resource Identifier (URI) or blank node. An URI is a a string of characters which identifies a resource which can be a locator (URL), a name (URN), or both. A blank node represents an anonymous resource which is not directly identifiable. The predicate is a URI which represents a relationship. The object can be an URI, a blank node or a Unicode string literal. Note that a given URI can be subject of one expression, predicate of another and object of a third one.

A predefined property, `rdf:type` is provided to classify entities into different categories. RDF permits to represent groups of entities. The first way to group things is the use of some containers. The three kinds of containers are `rdf:Bag`, `rdf:Seq` and `rdf:Alt`, depending if one wants to have a group of distinct things or not and if the order of the given elements is relevant or not. Their use permits to list entities that are part of the container, but there is no way to specify that no other item can be part of the group modelled by the container. Collections has been created for this purpose. A collection is a list structure based on the the predefined types `rdf:List`, the predefined properties `rdf:first` and `rdf:rest`, and the predefined resource `rdf:nil`. RDF Reification provides a built-in vocabulary used to describe RDF statements. This vocabulary consist in the type `rdf:Statement`, and the properties `rdf:subject`, `rdf:predicate`, and `rdf:object`.

The main field of a structured value is given by the `rdf:value` property.

`rdf:XMLLitteral` is used for the sake of simplicity when some litteral contains XML notation.

2.2 RDF Schema

General notions RDF Schema (RDFS) is a specification that explains how to describe RDF vocabularies. It provides a kind of type system for RDF.

Syntax Some classes of things can be defined in RDFS to model some categories. A class is defined as being a resource which has a `rdf:type` property for which the value is `rdfs:Class`. A resource can be an instance of sevral classes and every class is an instance of the class `rdfs:Resource`. The `rdfs:subClassOf` property induce a specialization relationship between classes. This relation is transitive. One can define some properties of classes: `rdf:Property` is the superclass of all properties. The properties `rdfs:domain`, `rdfs:range`, and `rdfs:subPropertyOf` are special

kinds of properties. `rdfs:domain` is used to express that a given property is designed to describe a particular class. `rdfs:range` ensures that the value of a property ranges in a given class. The `rdfs:subPropertyOf` property induces a specialization of two properties. The class `rdfs:Datatype` is used to identify an URIref as datatype. `rdfs:Literal` defines the class of all literal values such as strings and integers. Other built-in properties exit in RDFS. For example, `rdfs:comment` and `rdfs:label` to provide a human readable description or name of a resource.

RDFS Rules The RDFS rules, as detailed in `http://www.w3.org/TR/rdf-mt/` are available in Figure 1. The rules should be read: "if one finds this information (as detailed in column 2) then the following information (column 3) should be added".

Rule Name	If E contains:	then add:
rdfs1	uuu aaa lll. where lll is a plain literal (with or without a language tag).	_:nnn rdf:type rdfs:Literal . where _:nnn identifies a blank node allocated to lll by rule rule lg.
rdfs2	aaa rdfs:domain xxx . uuu aaa yyy .	uuu rdf:type xxx .
rdfs3	aaa rdfs:range xxx . uuu aaa vvv .	vvv rdf:type xxx .
rdfs4a	uuu aaa xxx .	uuu rdf:type rdfs:Resource .
rdfs4b	uuu aaa vvv.	vvv rdf:type rdfs:Resource .
rdfs5	uuu rdfs:subPropertyOf vvv . vvv rdfs:subPropertyOf xxx .	uuu rdfs:subPropertyOf xxx .
rdfs6	uuu rdf:type rdf:Property .	uuu rdfs:subPropertyOf uuu .
rdfs7	aaa rdfs:subPropertyOf bbb . uuu aaa yyy .	uuu bbb yyy .
rdfs8	uuu rdf:type rdfs:Class .	uuu rdfs:subClassOf rdfs:Resource .
rdfs9	uuu rdfs:subClassOf xxx . vvv rdf:type uuu .	vvv rdf:type xxx .
rdfs10	uuu rdf:type rdfs:Class .	uuu rdfs:subClassOf uuu .
rdfs11	uuu rdfs:subClassOf vvv . vvv rdfs:subClassOf xxx .	uuu rdfs:subClassOf xxx .
rdfs12	uuu rdf:type rdfs:ContainerMembershipProperty .	uuu rdfs:subPropertyOf rdfs:member .
rdfs13	uuu rdf:type rdfs:Datatype .	uuu rdfs:subClassOf rdfs:Literal .

Fig. 1. RDFS rules

2.3 Conceptual Graphs (CGs)

Conceptual Graphs were introduced by Sowa (cf. [6, 7]) as a diagrammatic system of logic with the purpose "to express meaning in a form that is logically precise, humanly readable, and computationally tractable". In this paper we use the term "Conceptual Graphs" to denote the *family of formalisms* rooted in Sowa's work and then enriched and further developed with a graph-based approach in [8].

Conceptual Graphs encoded knowledge as graphs and thus can be visualized in a natural way:

- The vocabulary, which can be seen as a basic ontology, is composed of hierarchies of concepts and relations. These hierarchies can be visualized by their Hasse diagram, the usual way of drawing a partial order.
- All other kinds of knowledge are based on the representation of entities and their relationships. This representation is encoded by a labeled graph, with two kinds of nodes, respectively corresponding to entities and relations. Edges link an entity node to a relation node. These nodes are labeled by elements of the vocabulary.

The **vocabulary** is composed of two partially ordered sets: a set of concepts and a set of relations of any arity (the arity is the number of arguments of the relation). The partial order represents a specialization relation: $t' \leq t$ is read as "t' is a specialization of t". If t and t' are concepts, $t' \leq t$ means that "every instance of the concept t' is also an instance of the concept t". If t and t' are relations, then these relations have the same arity, say k, and $t' \leq t$ means that "if t' holds between k entities, then t also holds between these k entities").

A **basic graph** (BG) is a bipartite graph: one class of nodes, called *concept* nodes, represents entities and the other, called *relation* nodes represents relationships between these entities or properties of them. A concept node is labeled by a couple $t : m$ where t is a concept (and more generally, a list of concepts) and m is called the marker of this node: this marker is either the generic marker, denoted by $*$, if the node refers to an unspecified entity, otherwise this marker is a specific individual name. BGs are used to represent assertions called *facts*. They are also building blocks for more complex kinds of knowledge (such as rules, or nested graphs). In this paper we only detail rules as they are of direct interest to the framework we are proposing.

A **rule** expresses implicit knowledge of form "if *hypothesis* then *conclusion*", where hypothesis and conclusion are both basic graphs. Using such a rule consists of adding the conclusion graph (to some fact) when the hypothesis graph is present (in this fact). There is a one to one correspondence between some concept nodes of the hypothesis with concept nodes of the conclusion. Two nodes in correspondence refer to the same entity. These nodes are said to be *connection nodes*. The knowledge encoded in rules can be made explicit by applying the rules to specific facts.

These graphical objects are provided with a **semantics in first-order-logic**, defined by a mapping classically denoted by Φ in conceptual graphs [7]. First, a FOL language corresponding to the elements of a vocabulary $\mathcal{V}$ is defined: concepts are translated into unary predicates and relations of arity k into predicates of arity k. Individual names become constants. Then, a set of formulas $\Phi(\mathcal{V})$ is assigned to the vocabulary. These formulas translate the partial orders on concepts and relations: if t and t' are concepts,

with $t' < t$, one has the formula $\forall x(t'(x) \rightarrow t(x))$; similarly, if r and r' are k-ary relations, with $r' < r$, one has the formula $\forall x_1 \ldots x_k(r'(x_1 \ldots x_k) \rightarrow r(x_1 \ldots x_k))$. A fact G is naturally translated into a positive, conjunctive and existentially closed formula $\Phi(G)$, with each concept node being translated into a variable or a constant: a new variable if it is a generic node, and otherwise the constant assigned to its individual marker. The logical formula assigned to a rule R is of form $\Phi(R) = \forall x_1 \ldots x_p\ ((hyp) \rightarrow \exists y_1 \ldots y_q\ (conc))$, where: hyp et $conc$ are conjunctions of atoms respectively translating the hypothesis and the conclusion, with the same variable being assigned to corresponding connection nodes; $x_1 \ldots x_p$ are the variables assigned to the concept nodes of the hypothesis; $y_1 \ldots y_q$ are the variables assigned to the concept nodes of the conclusion except for the connection nodes.

More importantly, first order logic subsumption can also be translated in a graphical operation: homomorphism. A homomorphism from G to H is a mapping between the node sets of G to the node sets of H, which preserves the adjacency between nodes of G and can decrease the node labels. If there is a homomorphism (say π) from G to H, we say that G *maps* to H (by π).

3 The sound and complete translation

This translation will simply translate each triplet RDF in a ternary relation where each of the concept nodes of the relation will represent the RDF triplet elements. See Figure 2 below where on top of the image the CG translation is shown for the RDF triplets shown at the bottom of the picture:

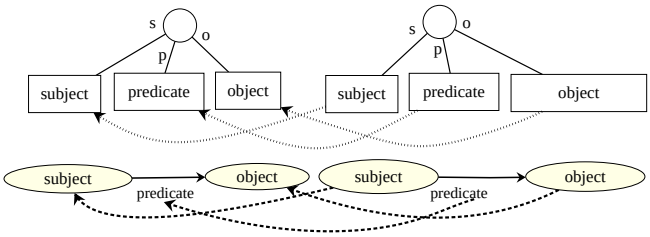


Fig. 2. RDF to CG translation example and the according homomorphism

This translation is straightforward and will ensure soundness and completeness of homomorphism (as opposed to the first translation where we are sound and complete but only with respect to a subset of RDF(S)). However, this translation is not visual or in the spirit of Conceptual Graphs as such (the support is totally flat).

3.1 Translating RDFS rules

The RDFS rules are translated in rules over graphs obtained from triplets as depicted in the following Figure 3. Darker nodes represent the conclusion of the rule.

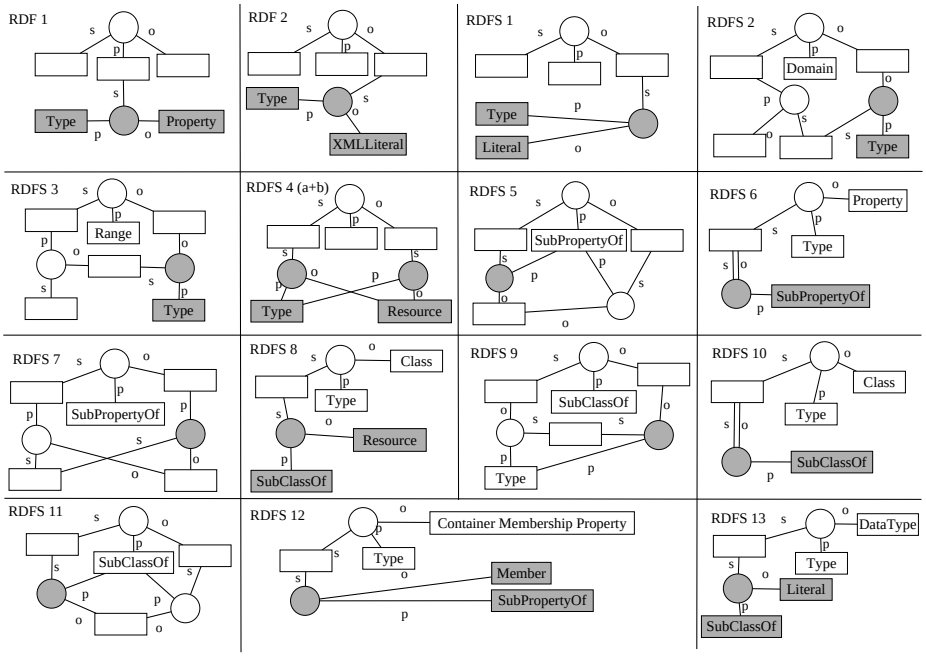


Fig. 3. Graph based RDFS rules

However, this translation is not in the spirit of Conceptual Graphs (as already disscussed in the introduction) given the (1) flatness of the support and (2) the visual properties of the representation. Indeed, for 2 nodes and one edge in the original graph, this representation will replace it with 4 nodes and 3 edges. In Figure 4 such a graph is depicted.

4 The CG spirit translation

In this section we present a more intuitive translation from RDF to CGs. The main idea behind it is to try to exploit as much as possible the separation between background knowledge and factual knowledge. However, RDF represents information at a different meta level than the one followed by the Conceptual Graphs optic. A direct consequence is that the same object can act as a individual marker, a relation or a concept type in RDF. For the work presented in this paper we will only focus on the RDF subset in which the three above mentioned sets are disjoint. Current and future work is looking at each case where the overlap might occur and how to address each case separately. These results are out of the scope of this paper. Also, the order in which we process the triplets will count. But for now we will only give the translation of each structural element in RDF versus CGs with a more in depth discussion of the problems that arise

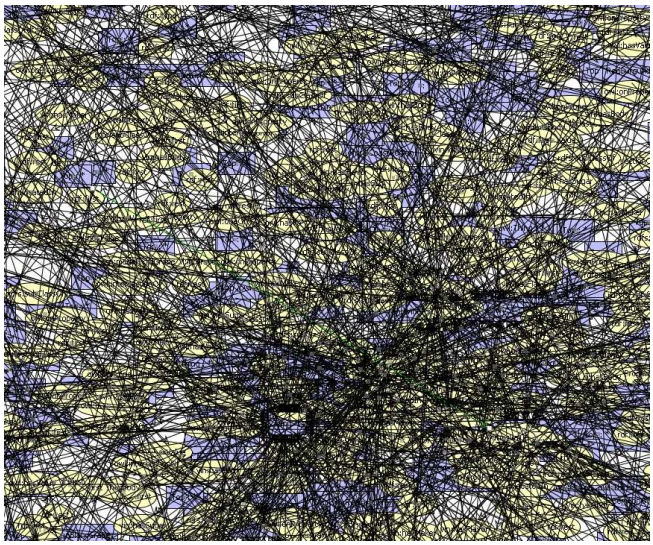


Fig. 4. Unintuitive visual representation of a RDF graph using CGs

as a consequence of the triplet ordering to be detailed in a different paper. A list of these constructs and their according translation in Conceptual Graphs is enumerated below.

- Support: Concept Hierarchy
  - **rdfs:Resource** - top of the concept type hierarchy
  - **rdfs:Class** - given the particularity of the relation between Class and Resource, Class will be implicitly represented. Every time we find "class" we will create a new concept in the concept type hierarchy (just underneath top). The classes will be linked up with **rdfs:subClassOf**.
  - **rdfs:Literal** will become a concept right under top. This concept will be further refined into a set of concepts referring to datatypes. **rdf:XMLLiteral** will be another one of these concepts.
    rdfs:Datatype will represent the union of all these concepts.
  - **rdf:Container** with the descendants **rdf:Bag**, **rdf:Seq** and **rdf:Alt** as a subclass of top
  - **rdf:List** as a subclass of top
  - **rdf:Statement** as a subclass of top
- Support: Relation Hierarchy
  - There will be two relation hierarchies: one for binary relations and one for ternary relations. The top of the binary relations hierarchy is T(Resource, Resource). The top of the ternary relation hierarchy is T(Resource, Resource, Resource).
  - **rdf:Property** - treated the same as **rdfs:Class**: when we encounter it we will replace it by a new binary relation. The signature will be also managed by the

RDFS rules. The binary relations hierarchy obtained in this manner will be given by **rdfs:subPropertyOf**
    - **rdfs:member** is a subrelation of T(Resource, Resource, Resource). Its signature is (rdfs:Container, rdfs:Resource, rdfs:Literal).
      rdfs:ContainerMembershipProperty is a subrelation of this one with the same signature.
    - **rdf:first**, **rdf:rest** and **rdf:value** are subrelations of T(Resource, Resource). The signature of rdf:first is (Resource, List), the signature of rdf:rest is (List, List) and the signature of rdf:value is (Resource, Resource).
- Other manipulations
    - **rdfs:label** - treated by the multi language facilities in COGXML
    - **rdfs:comment** - also treated by the multi language facilities in COGXML
    - **rdfs:seeAlso** - also treated by the multi language facilities in COGXML
    - **rdfs:range** and **rdfs:domain** will give the signature of the relation along with rdfs rules.
    - **rdfs:type** will create the individuals. The individuals are crated as we parse the document except for **rdf:nil** which is a predefined individual.
    - **rdf:statement** is treated as a concept type. For each statement we will do a different nesting, eventually with the concepts linked up by corefs. **rdf:subject**, **rdf:predicate** and **rdf:object** will give the graph inside the nesting.

4.1 The first translation and RDFS rules

In this section we will present how this translation deals with the semantics imposed by the RDFS rules presented in the preliminary notions Section. Please note that in the list below the item numbers of each rule correspond to the numbers of RDFS rules as found in the W3C document available at: `http://www.w3.org/TR/rdf-mt/`

- **SE1**: A blank node is treated as a generic concept node (as a consequence we will not have the renaming isses the semantic web community have with the new set of blank nodes.) For the rule SE1 given the fact that projection will take care of the matching individual / generic concept we do not have any extension to do.
- **SE2**: For the rule SE2 given the fact that projection will take care of the matching individual / generic concept we do not have any extension to do.
- **lg**: Just as above, the rule will provide the generalisation mechanism for literals (since literals cannot be a subject or a predicate of a statement).
- **gl**: This rule will only work for Datatypes (as in rule RDFS 13 and RDFS 1). The intuition here is since a literal can only be an object then one needs to create a blank node to act as a potential subject. The RDFS 13 rule does not apply in our case since the datatype in the concept hierarchy is the reunion of all datatypes. This means that if x is of type literal then we put x as a subclass of Literal. We will also write a constraint that certifies that Literal can only be a second neighbor.
- **RDFS 2**: This rule will mean that we need to update the signature of the relation type in the support with the according domain
- **RDFS 3**: This rule will mean that we need to update the signature of the relation type in the support with the according range

– **RDFS 4**: Is already expressed (Resource is Top) in the support
– **RDFS 5**: Already expressed in the support
– **RDFS 6**: Taken care by the subsumption relation of the support
– **RDFS 7**: Already in the support of relations
– **RDFS 8**: Already in the support
– **RDFS 9**: Already in the support of concepts
– **RDFS 10**: Taken care by the subsumption relation of the support
– **RDFS 11**: Already in the support
– **RDFS 12**: Taken care by the decision to put $member$ as a superclass of $containerMembership$
– **XMLClash**: We will add a $ILLTypedLiteral$ concept in the hierarchy and parse the nodes accordingly.
– **ext1** Already taken care by the projection mechanism
– **ext2** Already taken care by the projection mechanism
– **ext3** Will be done by adding conjunctive types for signatures
– **ext4** Will be done by adding conjunctive types for signatures
– **ext5 - ext 9**: We cannot do with CGs.

A visual depiction of such translation is given in Figures 5 and 6 where we show the facts, and respectively hierarchy of relations of a subset of the WINE ontology available at: `http://www.schemaweb.info/webservices/rest/GetRDFByID.aspx?id=62`.

5 Current and future work

This paper has presented the initial work carried out towards the analysis of the translation between RDF and CGs. While a lot of problems are still currently addressed (hence not detailed in this paper) the direction of work is clearly of benefit for both CG and Semantic Web community. An in depth analysis of the semantic tradeoffs of the translation at hand, where each particular case is separately addressed, and where soundness and completeness results are clearly provided, is of great benefit in an era where the development of effective techniques for knowledge representation and reasoning (KRR) is crucial for successful intelligent systems.

References

1. Berners-Lee, T.: Conceptual graphs and the semantic web - reflections on web architecture (2001) `http://www.w3.org/DesignIssues/CG.html`, last accessed 25 June 2009.
2. Corby, O., Dieng, R., Hbert, C.: A conceptual graph model for w3c resource description framework. In: In In Proceedings of ICCS-2000, Springer (2000) 468–482
3. Martin, P., Eklund, P.W.: Knowledge retrieval and the world wide web. IEEE Intelligent Systems **15**(3) (2000) 18–25
4. Yao, H., Etzkorn, L.H.: Automated conversion between different knowledge representation formats. Knowl.-Based Syst. **19**(6) (2006) 404–412
5. Baget, J.F.: RDF entailment as a graph homomorphism. In: Proc. of the International Semantic Web Conference. (2005) 82–96

6. Sowa, J.F.: Conceptual graphs for a database interface. IBM Journal of Research and Development **20**(4) (1976) 336–357
7. Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley (1984)
8. Chein, M., Mugnier, M.L.: Graph-based Knowledge Representation. Springer (2009)

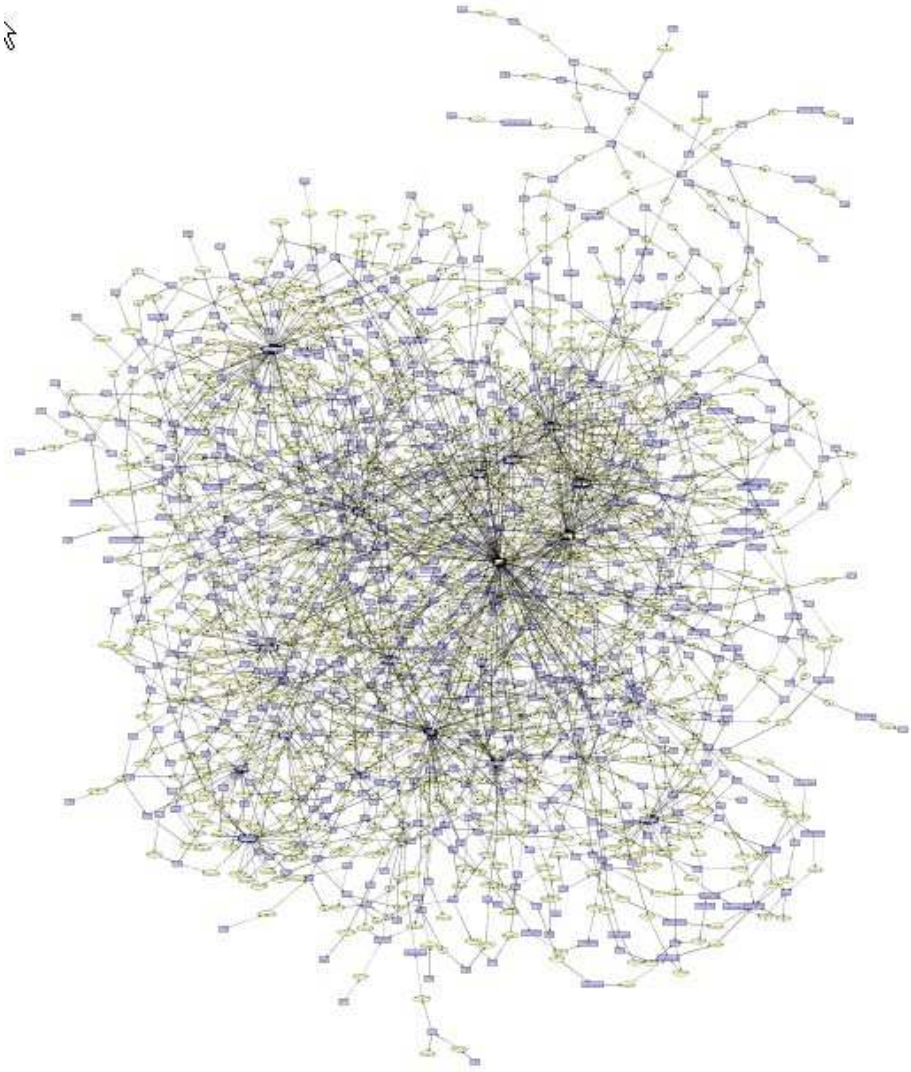


Fig. 5. RDF to CG translation for the WINE ontology (facts)

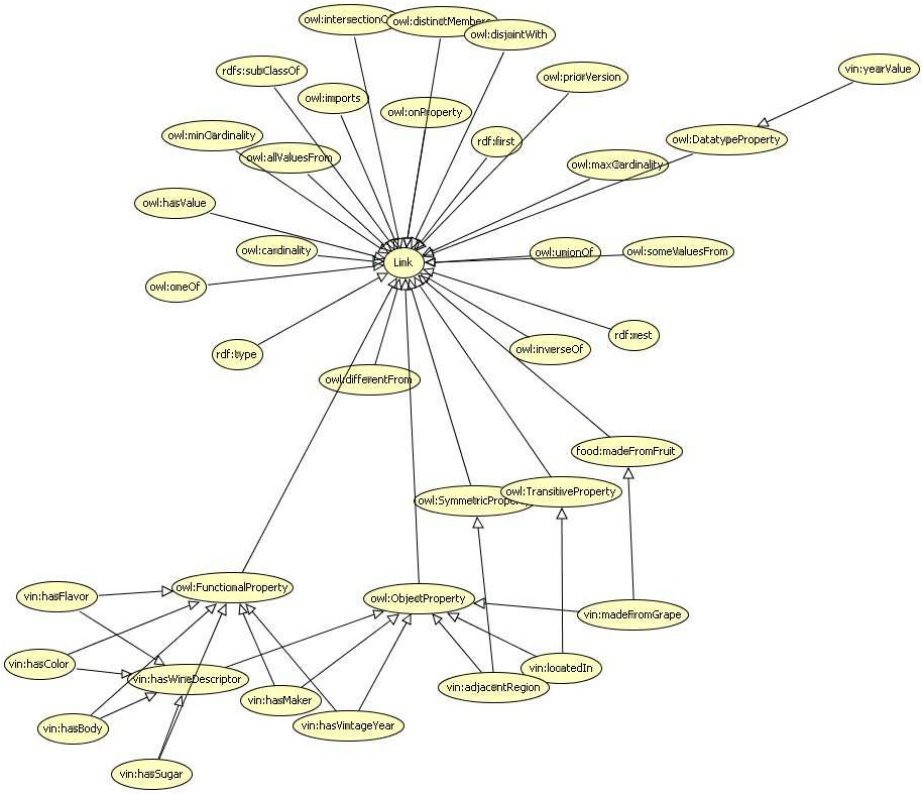


Fig. 6. RDF to CG translation for the WINE ontology (relation types)

Integrated research environment "Graph Model Workshop"

Victor Kokhov, Alexey Neznanov, Sergey Tkachenko

State University - Higher School of Economics, Myasnitskaya 20,
101000, Moscow, Russian Federation

vkokhov@hse.ru, aneznanov@hse.ru, svt@graphmodel.com

Abstract. Structured knowledge representation (e.g. conceptual graphs, semantic networks, etc.) needs to be supported by modern software. This is especially true for research fields where objects are modeled with graphs. We introduce an integrated research environment, "Graph Model Workshop", to support structural system analysis in such fields. This is a general-purpose software tool integrating a visual graph editor, an extensible set of graph problem solvers, a database, and modules for various kinds of analysis.

Keywords: graph model, conceptual graph, structural analysis, software.

1 Introduction

There are not many tools with rich functionality in the field of knowledge representation and manipulation. Such systems must be built on a strong mathematical (logical, graph-theoretical, etc.) basis, modern human computer interaction principles and technological achievements. A new generation of tools must adopt modern standards and programming environments.

Most of publicly available tools either are written in an interpretive language (Java in Amine toolkit [1]) or are designed only for editing and checking the conceptual graphs (CharGer CG drawer [2], ICom [3]) or may solve only specific problems or cannot are used by end users (Cogitant [4]).

We introduce the integrated research environment "Graph Model Workshop"[1] (GMW). It is a universal solution for computer-aided studies related to structural analysis of systems. Our principles: no Java, Python or other interpretive language in the *core* of system to improve the efficiency, a universal kernel of efficient graph functions and rich low-level DLL-API for functional extensibility, a maximal level of integration, a friendly user interface. The environment has been developed since 1998. At this time we are working on the fourth version of this application [5].

[1] Registered in ROSPATENT (CERTIFICATE № 2005612847, issued in 2005)

2 The Main Features of "Graph Model Workshop"

Consider the following features of the system:

- Tight integration (fig. 1) of interactive and batch tools in a flexible unified modern multi-document user interface (fig. 2):
  - visual graph editor;
  - text processor;
  - database management systems for structural and relational data;
  - solvers for structural analysis problems;
  - components for experimentation and analysis of results.

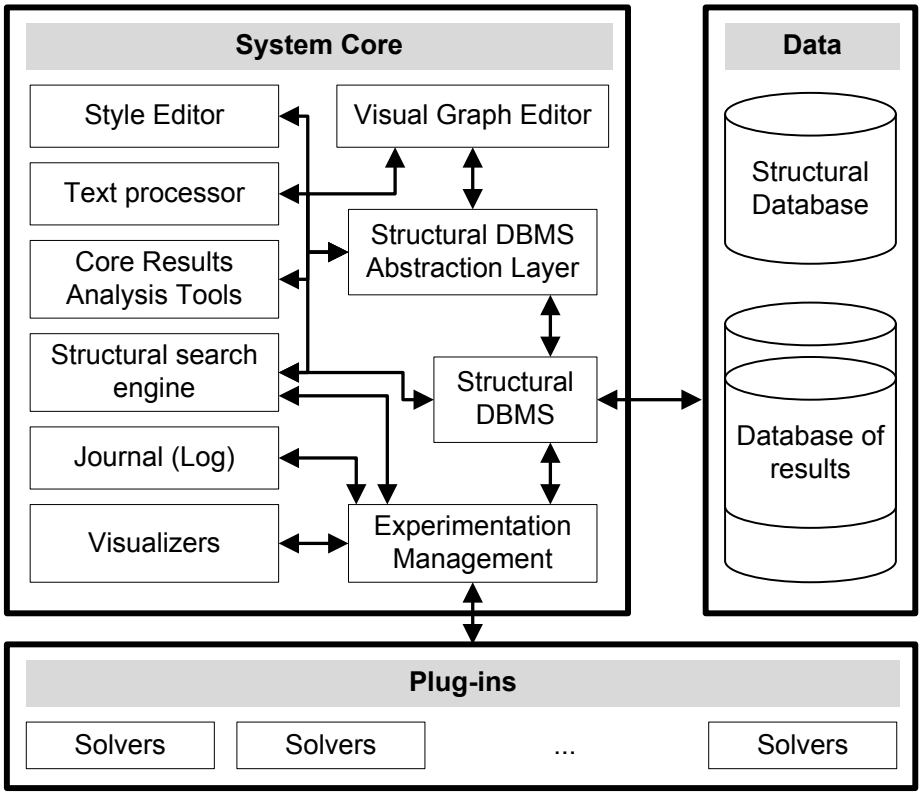


Fig. 1. GMW architecture

- Working with structures (graph models of systems), including digraphs with multiple weights (integer, float and string labels) on vertices and edges.
- The built-in editor of graph element styles designed to customize a visual appearance of structures for different theoretical and practical problem fields (for example, for formal concept analysis, group theory, organic chemistry, logistics or telecommunication).

- Automated multistage experimentation on structural databases with accumulation of results. Comparison, classification, ordering, clustering and visualization of results.
- Extendable structural search in databases, filtering of databases.
- Animated visualization of basic graph morphisms (automorphism, isomorphism, isomorphic embedding, etc), visualization of similarity graphs and graph fragments, visualization of graphs ordering and clustering.
- Import/export of graph databases in multiple formats. Ability to add new formats.
- The "PROVING GROUND" subsystem for performance assessment, testing and comparison of problem solvers.
- The open low-level and high-level application programming interfaces for adding new functions. Currently more than 370 GMW plug-ins have been developed.
- Learning and gaming components in educational versions of GMW ("STRIN", "POLYGON" and others).

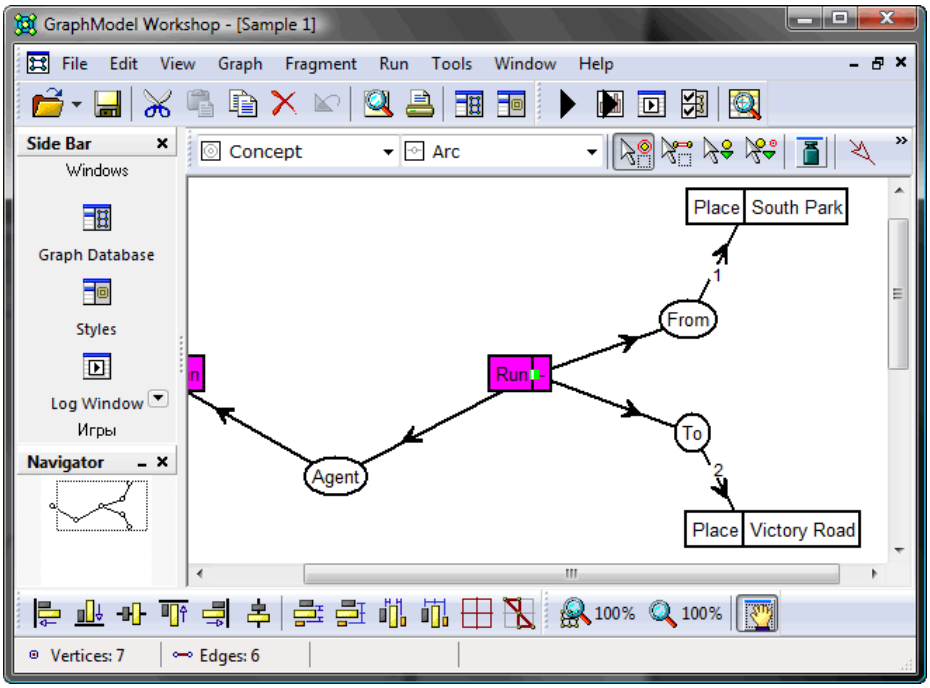


Fig. 2. The GMW main window with a visual graph editor of the conceptual graph "Sample 1"

2.1 Visual Editor of Structural Information

GMW supports interactive editing of structures from various problem fields (for example: conceptual graphs, semantic networks, maps, molecular graphs). The editor provides all the standard functions: labeling, complex selection, fragment rotation,

alignment and distribution of graph elements; unlimited undo, copy-paste capabilities, zooming. The editor is tightly integrated with manual and automated problem solvers.

Customization of graph elements labels/weights is achieved by editing *styles*. A style of a vertex or an edge defines its visual appearance depending on its weights.

2.2 Problem Solvers, Visualization and Animation

GMW provides a plug-in system. There are open APIs for adding new plug-ins: low-level API for plug-ins in dynamic link libraries and several high-level APIs for exe-files. All solvers are implemented as plug-ins and can be easily managed and applied to the selected (interactive mode) or all (batch mode) structures in the open database.

The core solvers include: identification and distinction analysis of structures (isomorphism, isomorphic embedding), maximal common fragment searching, transformation and decomposition of structures, ordering and classifying of structures by their complexity, similarity analysis of structures and their fragments, symmetry analysis.

GMW has built-in visualization capabilities for results of several important problems. They include animated demonstration of graph layout building and graph morphisms. Results of graph renumbering, drawing, transformation, ordering and statistical analysis can be shown. These capabilities can be used by all the core components and plug-ins.

2.3 Databases of Structural Information

GMW works with structures organized in databases. Databases contain both structures and results of computational experiments. GMW offers searching, comparison and visualization of structures, as well as advanced tools for processing of results: statistical analysis, check for equality, comparative check, clustering, ordering and so on (fig. 3).

2.4 Research Automation

The user can define schemes of experiments containing a list of problem solvers to be performed repeatedly in batch mode for whole database or filter (fig. 4). Parameters for these schemes are either defined up front or entered at the runtime. The user also can conduct multistage experiment, when the results of the previous stage are used as input data for the next stage.

2.5 “PROVING GROUND”

GMW contains the original subsystem for the efficiency analysis of graph algorithms. It performs computational complexity estimation, comparison of efficiency and correctness of different problem solvers.

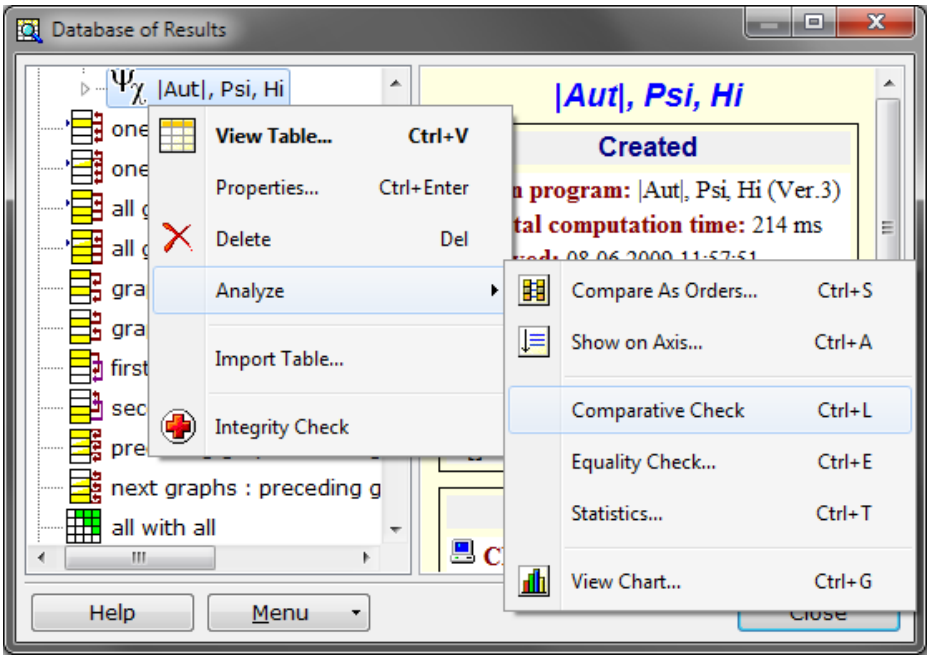


Fig. 3. Stored results of computations and standard tools for their analysis

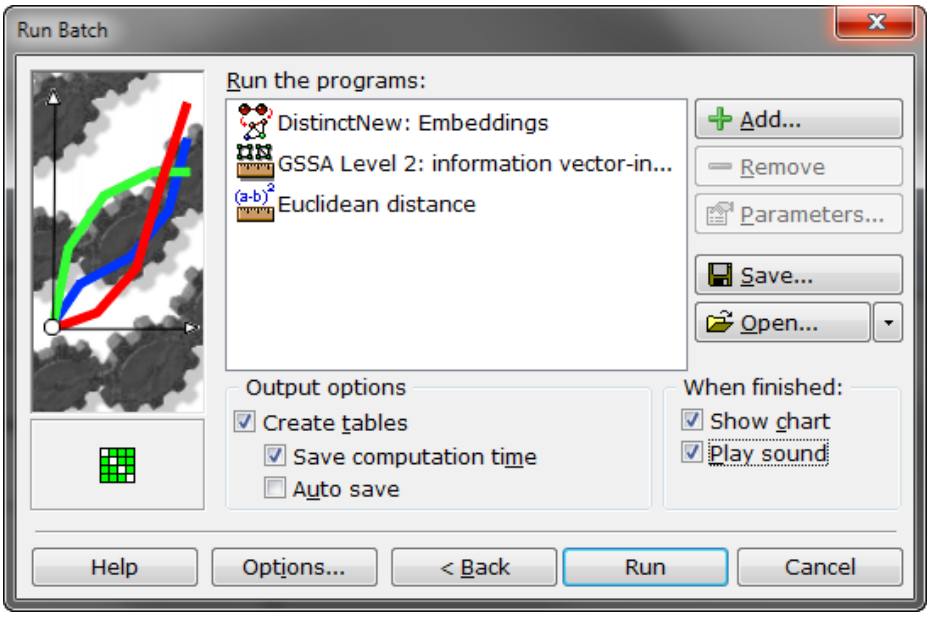


Fig. 4. Last screen of master for running experimentation in batch mode

2.6 GMW system requirements

For using last version of GMW computer must meet following requirements:

- OS Windows 98/Me/NT/2000/XP/2003/Vista (Windows 2000 or higher is recommended).
- Intel Pentium class processor (Intel Pentium III / AMD Athlon 500MHz or faster are recommended).
- 64 MB of free memory (256 MB are recommended for real experiments).
- 14-60 MB on HDD (without bases of structural information).

3 Conclusion

GMW helps solving the following problems:

- Identification and finding distinctions of structures (canonization, isomorphism, homomorphism, isomorphic embedding, isomorphic intersection).
- Decomposition of structures into all labeled or nonisomorphic fragments with the specified constraints.
- Constrained generation and constructive enumeration of structures.
- Symmetry analysis (automorphism group processing).
- Finding the distinction of fragments placement in the structure topology.
- Ordering and classifying structures according to their complexity.
- Structural similarity analysis taking a placement of fragments into account.
- Exact or fuzzy searching in databases of structural information, graph mining.
- Automatic diagram drawing, finding optimal graph layouts.

We see the further development of GMW in continuous expansion of the set of solvers, supported data formats (e.g. a full support of Conceptual Graph Interchange Format (CGIF)), and application fields without a loss of generality. The system core must be independent and as fast, isolated and robust as possible. As a result, an extensible toolkit with balanced characteristics will be built and successfully used in applied graph theory, group theory, formal concept analysis, structural linguistics, natural language processing.

References

1. Amine. A Java Open Source Platform for the Development of Intelligent Systems & Multi-Agent Systems, http://amine-platform.sourceforge.net/kora.htm
2. CharGer. A Conceptual Graph Editor by Harry Delugach, http://charger.sourceforge.net
3. Fillottrani, P. R., Franconi, E, Tessaris, S: The new ICOM Ontology Editor. Proceedings of the 2006 Int. Workshop on Description Logics, Lake District, UK, (2006)
4. Cogitant – A Conceptual Graph Library, http://cogitant.sourceforge.net
5. Kokhov, V.A., Neznanov, A.A., Tkachenko, S.V.: Progress of IRE “Graph Model Workshop” for research in field of structural analysis. Proceedings of Int. Conf. “Information tools and technologies”, pp. 45–49. Moscow, (2008)

The FcaFlint Software for Relation Algebra Operations on FCA Contexts

Uta Priss

Edinburgh Napier University, School of Computing,
www.upriss.org.uk

Abstract. FcaFlint is a tool that implements context operations for FCA formal contexts. It is part of the FcaStone package. This paper provides an introduction to Relation Algebra operations as used by FcaFlint and discusses the command-line interface and implementation details of FcaFlint.

1 Introduction

FcaFlint[1] is a tool that implements context operations for FCA formal contexts. It can be used as a stand-alone tool (in which case the context must be formatted in the ".cxt" or "Burmeister" format), or it can be used in combination with FcaStone[2] (using any context format supported by FcaStone). The context operations are modelled using Relation Algebra (RA), which is an algebra that was invented by Peirce, Tarski and others and should not be confused with Codd's Relational Algebra (RLA). Both RA and RLA provide a foundation for query languages. While RLA is normally used for many-valued tables in relational databases (using SQL), RA is suitable for binary matrices as used in Formal Concept Analysis (FCA). RLA is more expressive than RA, but RA has some interesting features for the use with formal contexts. Although some context operations (such as calculating dual contexts) are already provided by other FCA tools, as far as we know, none of the available tools implement a larger set of operations.

This paper provides a brief introduction to RA (Sections 2 and 3) and discusses FcaFlint's command-line options and the implementation of RA operations in FcaFlint (Section 4). The RA operations are described at a very basic level which assumes no prerequisite knowledge about matrix operations. The use of RA for FCA, in particular for linguistic applications has been described by Priss (2005), Priss (2006) and Priss & Old (2006). Those papers also provide more background and references which are omitted in this paper.

There are many examples in the FCA literature where contexts are constructed from other contexts in a systematic manner, often involving RA operations (although RA may not be explicitly mentioned). Without having RA software, each application that uses RA operations requires special purpose code written for the application or manual editing of the formal contexts. With FcaFlint, RA context constructions (and also some additional non-RA constructions) can be much more easily derived. Some examples

[1] FcaFlint will be bundled with the next edition of FcaStone.

[2] http://fcastone.sourceforge.net/

of using FcaFlint for context constructions are shown by Priss (2009). A very brief example of using RA as a query language is given at the end of Section 3 below. A more extended introduction to RA (which contains Sections 2 and 3 below, but has more details) can be found on-line[3].

2 Introduction to RA: Basic operations

RA can be defined in a purely axiomatic fashion and can be used with many different applications. For this paper, only the application to Boolean matrices is of interest. Thus, the operations are defined with respect to Boolean (or binary) matrices, which are matrices that only contain 0s and 1s.

Figure 1 shows some of the basic operations. Union $I \cup J$, intersection $I \cap J$, complement $\overline{I}$ and dual I^d are applied coordinate-wise. For example, for union, a coordinate of the resulting matrix is 1, if a coordinate in the same position of any of the original matrices is 1. For intersection, the resulting matrix has a 1 in positions where both intersected matrices have a 1. Complementation converts 0s into 1s and 1s into 0s. The dual of a matrix is a mirror image of the original matrix, mirrored along the diagonal. There are three special matrices: the matrix one contains just 1s; nul contains just 0s; and dia, the identity matrix, contains 1s along the diagonal, 0s otherwise.

Figure 2 shows the composition operation $I \circ J = K$, which is a form of relational composition or Boolean matrix multiplication. If one conducts this operation by hand, it is a good idea to write the matrices in a schema as shown in the middle of Figure 2. The example on the right shows how an individual coordinate is calculated. The coordinate in the ith row and jth column is calculated by using the ith row of the left matrix and jth column of the right matrix. The individual coordinates are multiplied (using Boolean AND: $1 \times 1 = 1$; $1 \times 0 = 0 \times 1 = 0 \times 0 = 0$) and then added (using Boolean OR: $1 + 1 = 1 + 0 = 0 + 1 = 1$; $0 + 0 = 0$).

The bottom part of Figure 2 shows that non-square matrices can also be composed. But non-square matrices are not part of the usual RA definition. There are many ways to define RAs, abstractly or with respect to specific applications. The FCA-oriented RA definitions used in this paper are adapted from Priss (2006).

Definition 1. *A matrix-RA is an algebra* $(R, \cup, ^-, one, \circ, ^d, dia)$ *where* R *is a set of square Boolean matrices of the same size;* one *is a matrix containing all 1s;* dia *is a matrix, which has 1s on the diagonal and 0s otherwise;* $\cup, ^-, \circ, ^d$ *are the usual Boolean matrix operations.* $\cap$ *and* nul *are defined as* $I \cap J := \overline{\overline{I} \cup \overline{J}}$ *and* $nul := \overline{one}$.

Non-square matrices do not form an RA, because these require:

Special rules for non-square matrices:

- For $\cup$ and $\cap$: the dimensions of the right matrix must be the same as the dimensions of the left matrix.

[3] An "Introduction to using Relation Algebra with FCA" can be downloaded from: http://www.upriss.org.uk/fca/relalg.html

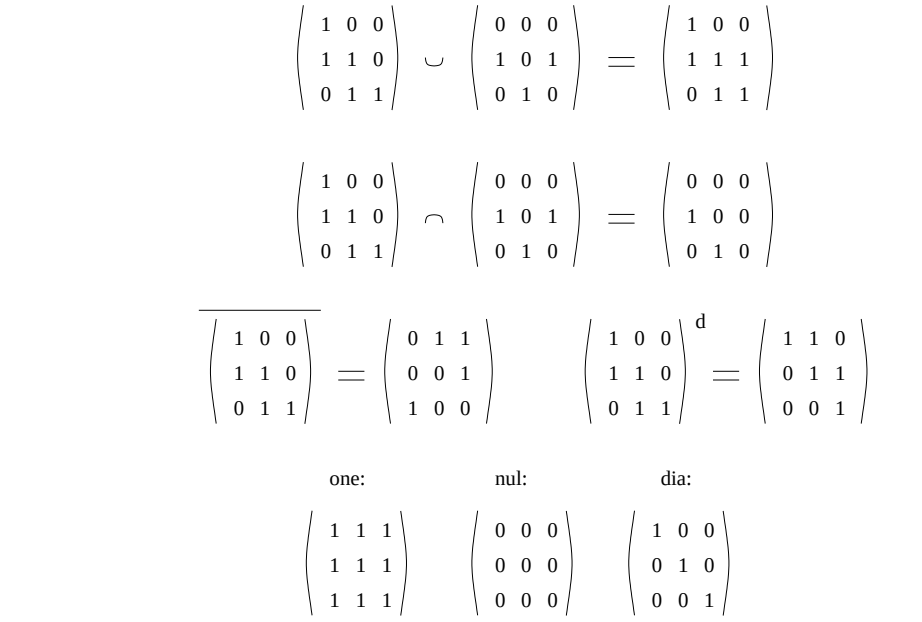


Fig. 1. Union, intersection, complement, dual matrix; the one, nul and dia matrices

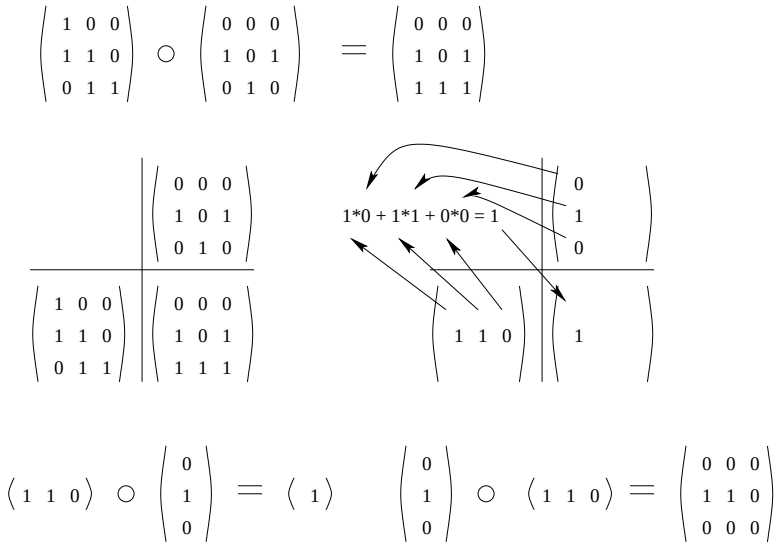


Fig. 2. Relational composition

– For $\circ$: the number of columns in the left matrix must equal the number of rows in the right matrix.
– dia, one and nul refer to sets of matrices whose actual dimensions depend on the matrices and operations with which they are used.

These special rules mainly refer to the theoretical definition of matrix-RAs. It is possible to extend the definition of union, intersection and composition to matrices of arbitrary dimensions (if one is not concerned about creating an RA). But because there is more than one way to extend these definitions, it needs to be carefully considered which operations are meaningful for a particular application. This is further discussed in the next section.

A further operation called "transitive closure" is sometimes defined. It should be noted that when the expressivity of RAs is discussed, transitive closure is not part of RA because it cannot be expressed by the basic RA operations. This is because although it only uses $\cup$ and $\circ$ in its definition, the dots (...) in its definition indicate some sort of infinity, which cannot be expressed by the other operations. (The proof for this is well-known and beyond this paper.)

Figure 3 shows the transitive closure of the composition operation, which is defined as $I^{trs} := I \cup I \circ I \cup I \circ I \circ I \cup \ldots$. If the matrix I has 1s on the diagonal, I^{trs} is calculated by composing I with itself until it does not change anymore (as shown in the top half of Figure 3). If the matrix I does not have all 1s on the diagonal, I is still composed with itself until it does not change anymore, but I and the results at each stage are unioned (as shown in the bottom half of Figure 3).

3 Using RA with formal contexts

Formal Concept Analysis (FCA) uses the notion of a formal context $(\mathtt{G}, \mathtt{M}, I)$ which consists of a set $\mathtt{G}$, a set $\mathtt{M}$ and a binary relation between $\mathtt{G}$ and $\mathtt{M}$ represented[4] by the Boolean matrix I. Figure 4 shows two formal contexts (K_I and K_J). The elements of $\mathtt{G}$ are called *(formal) objects*; the elements of $\mathtt{M}$ are called *(formal) attributes*. RA should be applicable to the matrices of formal contexts, but because the matrices need not be square, this is not completely straightforward. Furthermore, the rows and columns in the matrices have interpretations: each row corresponds to an object; each column corresponds to an attribute. RA operations on formal contexts are only meaningful if they take these interpretations into consideration.

In FCA, concept lattices are produced from the formal contexts. This is not relevant for this paper, but it should be pointed out that the RA operations on contexts in general do not translate into the same operations on lattices. For example, a union of contexts does not produce a union of lattices. Some RA operations may not have any useful applications for FCA.

Because the use of RA operations for formal contexts is intuitive, but the construction of an RA for FCA is not completely straightforward, Priss (2006) provides two

[4] Because sets can be encoded as matrices, typewriter font ($\mathtt{H}$) is used to denote sets, italics (H) is used for matrices (but not in the figures). The matrices of formal contexts are written with crosses instead of 1s.

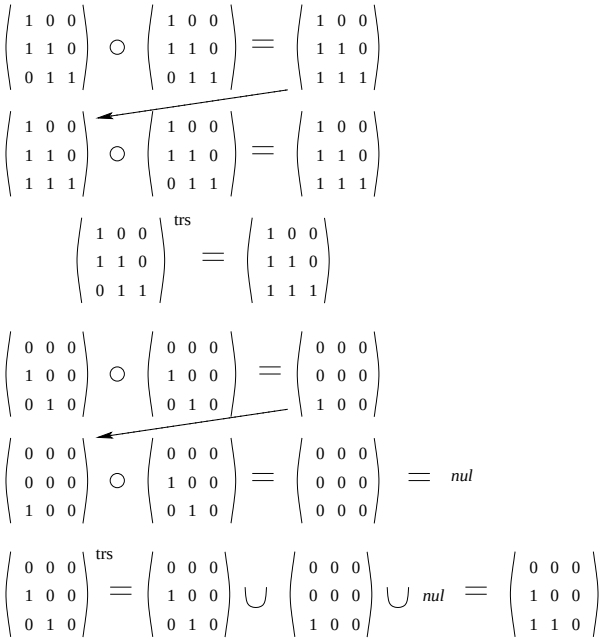


Fig. 3. Transitive closure

different suggestions to model this. The "unnamed perspective" is mostly of theoretical value because it makes it easy to form an RA, but is not practically useful. This perspective is called "unnamed" because objects and attributes are assigned to positions in a matrix, but their names are not used. Many FCA applications use not one, but many formal contexts which are often stored in a database. In the case of the unnamed perspective all objects and attributes of all of the contexts stored in a database are gathered into one linearly ordered set called an *active domain* $\mathcal{A}$. The matrices of the contexts are transformed into $|\mathcal{A}|$-dimensional matrices. It would not be practically useful to actually implement RA operations for FCA in this manner. Therefore this perspective is not further discussed in this paper[5].

The second suggestion is called the "named perspective" and describes a more practical approach that can be directly implemented in software, but which is more remote from the theoretical aspects of RAs. This perspective is discussed in the next section. The terms "named" and "unnamed" are chosen in analogy to their use in relational database theory.

[5] Further details on this perspective can be found in "Introduction to using Relation Algebra with FCA" available at: http://www.upriss.org.uk/fca/relalg.html

3.1 The named perspective

The "named perspective" uses names of rows and columns for all its matrices. In other words, the matrices used in this perspective all belong to formal contexts. Figure 4 shows the union of contexts according to the named perspective. In this perspective, operations can be defined in a set-theoretic manner or in a matrix-based manner, where each row and column corresponds to a "named" object or attribute. The ordering of rows and columns can be changed while the names are explicit (top half of Figure 4), but not while matrices are used (bottom half of Figure 4). Definition 2 shows the set-theoretic definitions of context operations (according to Priss (2006)). Apart from complement and dual, these operations are different from Ganter & Wille's (1999) definitions, although they are similar.

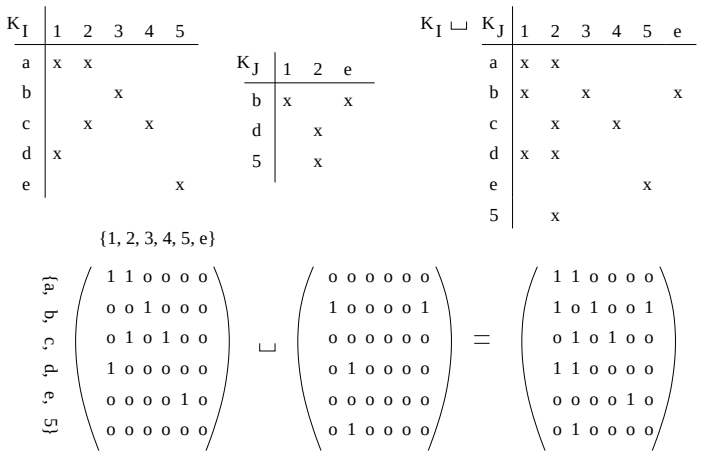


Fig. 4. Union in the named perspective

Definition 2. *For formal contexts* $\mathcal{K}_1 := (\mathtt{G}_1, \mathtt{M}_1, I)$ *and* $\mathcal{K}_2 := (\mathtt{G}_2, \mathtt{M}_2, J)$*, the following context operations are defined:*
$\mathcal{K}_1 \sqcup \mathcal{K}_2 := (\mathtt{G}_1 \cup \mathtt{G}_2, \mathtt{M}_1 \cup \mathtt{M}_2, I \sqcup J)$ *with* $\mathtt{g}I \sqcup J\mathtt{m} :\Longleftrightarrow \mathtt{g}I\mathtt{m}$ *or* $\mathtt{g}J\mathtt{m}$
$\mathcal{K}_1 \sqcap \mathcal{K}_2 := (\mathtt{G}_1 \cup \mathtt{G}_2, \mathtt{M}_1 \cup \mathtt{M}_2, I \sqcap J)$ *with* $\mathtt{g}I \sqcap J\mathtt{m} :\Longleftrightarrow \mathtt{g}I\mathtt{m}$ *and* $\mathtt{g}J\mathtt{m}$
$\mathcal{K}_1 \diamond \mathcal{K}_2 := (\mathtt{G}_1, \mathtt{M}_2, I \diamond J)$ *with* $\mathtt{g}I \diamond J\mathtt{m} :\Longleftrightarrow \exists_{\mathtt{n} \in (\mathtt{M}_1 \cap \mathtt{G}_2)} : \mathtt{g}I\mathtt{n}$ *and* $\mathtt{n}J\mathtt{m}$
$\overline{\mathcal{K}_1} := (\mathtt{G}_1, \mathtt{M}_1, \overline{I})$
$\mathcal{K}_1^d := (\mathtt{M}_1, \mathtt{G}_1, I^d)$

The operations in Definition 2 can be used with all formal contexts. This is in contrast to the operations in Definition 3, which can only be applied in cases where special conditions are met (such as, $\mathtt{G}_1 = \mathtt{G}_2, \mathtt{M}_1 = \mathtt{M}_2$).

Definition 3. *The following additional operations for formal contexts are defined for formal contexts* $\mathcal{K}_1 := (\mathtt{G}_1, \mathtt{M}_1, I)$ *and* $\mathcal{K}_2 := (\mathtt{G}_2, \mathtt{M}_2, J)$*:*

1. $\mathcal{K}_1 \cup \mathcal{K}_2 := \mathcal{K}_1 \sqcup \mathcal{K}_2$ *if* $\mathtt{G}_1 = \mathtt{G}_2, \mathtt{M}_1 = \mathtt{M}_2$
2. $\mathcal{K}_1 \cap \mathcal{K}_2 := \mathcal{K}_1 \sqcap \mathcal{K}_2$ *if* $\mathtt{G}_1 = \mathtt{G}_2, \mathtt{M}_1 = \mathtt{M}_2$
3. $\mathcal{K}_1 \circ \mathcal{K}_2 := \mathcal{K}_1 \diamond \mathcal{K}_2$ *if* $\mathtt{M}_1 = \mathtt{G}_2$

Using Definition 3, $\sqcup, \sqcap, \diamond$ can be translated into matrix-based operations as shown below and in the bottom half of Figure 4 because Definition 3 fulfills the "Special rules for non-square matrices". In contrast to the unnamed perspective, the matrices used here are of minimal dimensions (according to $\mathcal{K}_1^*$ and $\mathcal{K}_2^*$ below). The active domain $\mathcal{A}$ is used in this perspective as well but only in order to determine the order of the rows and columns: the objects and attributes corresponding to each matrix must be ordered in the same order as they appear in $\mathcal{A}$.

- $\mathcal{K}_1 \sqcup \mathcal{K}_2 = \mathcal{K}_1^* \cup \mathcal{K}_2^*$ with $\mathcal{K}_1^* = (\mathtt{G}_1 \cup \mathtt{G}_2, \mathtt{M}_1 \cup \mathtt{M}_2, I); \mathcal{K}_2^* = (\mathtt{G}_1 \cup \mathtt{G}_2, \mathtt{M}_1 \cup \mathtt{M}_2, J)$.
- $\mathcal{K}_1 \sqcap \mathcal{K}_2 = \mathcal{K}_1^* \cap \mathcal{K}_2^*$ with $\mathcal{K}_1^* = (\mathtt{G}_1 \cup \mathtt{G}_2, \mathtt{M}_1 \cup \mathtt{M}_2, I); \mathcal{K}_2^* = (\mathtt{G}_1 \cup \mathtt{G}_2, \mathtt{M}_1 \cup \mathtt{M}_2, J)$.
- $\mathcal{K}_1 \diamond \mathcal{K}_2 = \mathcal{K}_1^* \circ \mathcal{K}_2^*$ with $\mathcal{K}_1^* = (\mathtt{G}_1, \mathtt{M}_1 \cup \mathtt{G}_2, I); \mathcal{K}_2^* = (\mathtt{M}_1 \cup \mathtt{G}_2, \mathtt{M}_2, J)$.

Figure 5 shows how basic FCA operations can be formed in the named perspective, calculating $\mathtt{c}' = \{2, 4\}$ and $\mathtt{H}' = \{2\}$ for $\mathtt{H} = \{a, c\}$. The resulting algebraic structure is described in the next definition.

Definition 4. *A context algebraic structure (CAS) based on $\mathcal{A}$ is an algebra that implements the context operations from Definition 3. (See Priss (2006) for the complete definition.)*

A CAS is not an RA. There are no unique dia, one and nul matrices because these matrices need to change their dimensions and their sets of objects and attributes depending on what other matrices and operations they are used with. Furthermore, if negation is used in combination with composition, the results can be different from the ones in the unnamed perspective, which is a problem because the unnamed perspective does form an RA. There are ways to modify the CAS operations so that they yield an RA which is equivalent to the unnamed perspective, but this is complicated. Basically, CAS operations enlarge contexts as needed, by adding rows and columns. These rows and columns are usually filled with 0s, but if a context was previously negated once, they should be filled with 1s.

It is possible to solve this by distinguishing the *inside* of a formal context (which consists of the relation between objects and attributes that is currently defined for the context) and the *outside* of the context (which collects conditions about potential elements from the active domain that might be added to the context at a later stage). Only the inside is stored as a matrix. The outside is stored as a set of conditions (e.g., "all 0", "all 1") without having a complete list of which elements belong to the outside.

All contexts start out with their outside containing all 0s. Negation changes the outside to "all 1s". Union and intersection may enlarge the inside, but the outside is still either "all 0s" or "all 1s". Unfortunately, composition can change the outside of a context into conditions which are more complicated than "all 1" or "all 0". Thus, it is still not easy to create an RA in this manner. But the complexity of these conditions increases slowly. For many applications, the outside conditions of the contexts will be simple or irrelevant.

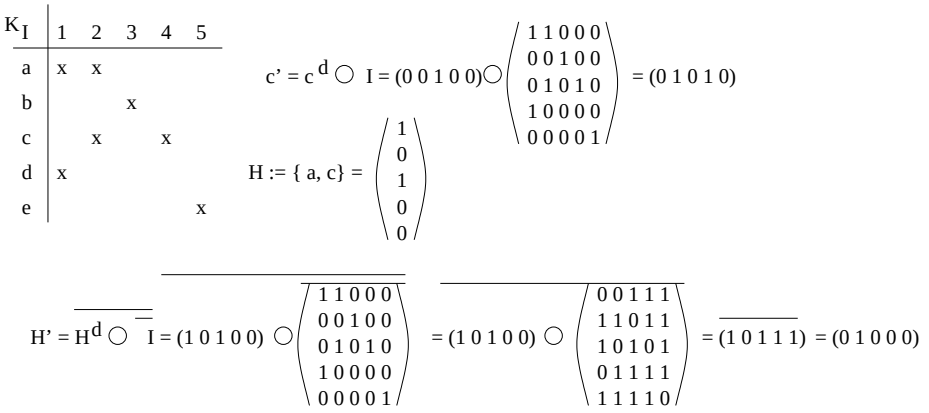


Fig. 5. Basic FCA operations in the named perspective

While it is beyond this paper to discuss applications in detail, Figure 6 provides a glimpse of how CAS can be used to model databases (Priss, 2005). In this example, a table with Employee data is translated into a relational schema, a many-valued context C_{Emp} and value scales V_{ename} and V_{eaddr}. A corresponding binary matrix I_{Emp} contains a 1 for every non-null value of C_{Emp} and a 0 for every null value (in this case a one matrix). The bottom half of Figure 6 calculates the RA equivalent of the RLA query "`select ename,eaddr from Emp where ename = 'mary' and eaddr = 'UK'`". In this modelling, $\underline{mary}$ is a row matrix indicating the position of "mary" in V_{ename}; similarly $\underline{UK}$ for $\overline{V_{eaddr}}$ and $\underline{ename}, \underline{eaddr}$ for I_{Emp}. The result is a matrix that has 1s in the positions of "mary" and "UK". The $mv()$ function maps this onto a submatrix of C_{Emp} with the values "mary" and "UK". This example shows how RA can be used as a query language, although whether this is practically useful still needs to be determined. Other, definitely useful applications are discussed in Priss (2009).

4 The implementation of RA operations in FcaFlint

The FcaFlint software implements all of the RA operations discussed in this paper. In the first version, the operations are implemented mostly as matrix operations as described in Section 2. The only checks that are implemented refer to the "Special rules for non-square matrices". (For example, the dimensions of $one, dia, nul, \overline{dia}$ are automatically determined where possible.) Furthermore, computing a dual matrix switches the object and attribute set and the result of composition selects the set of objects from the first matrix and the set of attributes from the second matrix. Otherwise, it is up to the user to make sure that the operations are meaningful with respect to formal contexts, i.e. that objects and attributes are ordered correspondingly and so on.

FcaFlint also provides the non-RA operations apposition, subposition, equality and transitive closure of composition. The $one, dia, nul, \overline{dia}$ matrices can be used for ap-

Employee table

key	ename	eaddr
e1	paul	UK
e2	mary	UK
e3	carl	USA
e4	sue	USA

a relational schema:

	Emp	eID	ename	eaddr
tEmp	1	1	1	1
e1	1	1	paul	UK
e2	1	2	mary	UK
e3	1	3	carl	USA
e4	1	4	sue	USA

C_{Emp}	eID	ename	eaddr
e1	1	paul	UK
e2	2	mary	UK
e3	3	carl	USA
e4	4	sue	USA

V_{ename}	paul	mary	carl	sue
e1	1			
e2		1		
e3			1	
e4				1

V_{eaddr}	UK	USA
e1	1	
e2	1	
e3		1
e4		1

$$\mathrm{dia}\left(\left\langle \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}\right\rangle \bigcirc \left\langle \begin{pmatrix}0\\1\\0\\0\end{pmatrix}\right\rangle \cap \left\langle \begin{pmatrix}1&0\\1&0\\0&1\\0&1\end{pmatrix}\bigcirc \left\langle \begin{pmatrix}1\\0\end{pmatrix}\right\rangle \right\rangle = \mathrm{dia}\left(\left\langle \begin{pmatrix}0\\1\\0\\0\end{pmatrix}\right\rangle \cap \left\langle \begin{pmatrix}1\\1\\0\\0\end{pmatrix}\right\rangle\right) = \left\langle \begin{pmatrix}0&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\right\rangle$$

$$\left\langle \begin{pmatrix}0&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\right\rangle \bigcirc \left\langle \begin{pmatrix}1&1&1\\1&1&1\\1&1&1\\1&1&1\end{pmatrix}\right\rangle \bigcirc \left\langle \begin{pmatrix}0&0&0\\0&1&0\\0&0&1\end{pmatrix}\right\rangle = \left\langle \begin{pmatrix}0&0&0\\0&1&1\\0&0&0\\0&0&0\end{pmatrix}\right\rangle \qquad \mathrm{mv}\left(\left\langle \begin{pmatrix}0&0&0\\0&1&1\\0&0&0\\0&0&0\end{pmatrix}\right\rangle\right) = \left\langle \begin{pmatrix}0&0&0&0\\0&\mathrm{mary}&\mathrm{UK}\\0&0&0&0\\0&0&0&0\end{pmatrix}\right\rangle$$

I_{Emp} dia(ename,eaddr)

Fig. 6. Calculating $dia((V_{ename} \circ \underline{mary}^d) \cap (V_{eaddr} \circ \underline{UK}^d)) \circ I_{Emp} \circ dia(\underline{ename}^d \cup \underline{eaddr}^d)$
or: `select ename,eaddr from Emp where ename='mary' and eaddr='UK'`

position and composition with other matrices, but not in combination with each other. This is because in that case, the dimensions of the matrices would be unknown. The composition function also implements the (non-RA) operations of requiring at least n values to be shared in the composition.

The operations are applied to a context stored in an input file (e.g. ex1.cxt) and the result is saved in a new file (e.g. output.cxt). The default format of the contexts is the Burmeister format, but in combination with FcaStone, any context format can be used that is supported by FcaStone. The RA operations are entered as functions as shown in Table 1. Table 2 shows examples of command-line usage of FcaFlint.

FcaFlint function	Meaning
dual(ex1.cxt)	Dual context
invers(ex1.cxt)	Invers context
union(ex1.cxt, ex2.cxt)	Union
inters(ex1.cxt, ex2.cxt)	Intersection
compos(ex1.cxt, ex2.cxt)	Relational composition
appos(ex1.cxt, ex2.cxt)	Apposition: ex1 on the left of ex2
subpos(ex1.cxt, ex2.cxt)	Subposition: ex2 underneath of ex1
equal(ex1.cxt, ex2.cxt)	Prints "Matrices are equal" or "Matrices are not equal"
trans(ex1.cxt)	Transitive closure of composition
<ONE>	one
<NUL>	nul
<DIA>	dia
<AID>	$\overline{dia}$

Table 1. RA operations in FcaFlint

FcaFlint command-line	Meaning or result
fcaflint "inters(ex1.cxt,invers((ex1.cxt))" output.cxt	$I \cap \overline{I}$
fcaflint "inters(ex1.cxt,compos(ex1.cxt,ex2.cxt))" output.cxt	$I \cap (I \circ J)$
fcaflint "equal(ex1.cxt,(dual(dual(ex1.cxt))))" output.cxt	$I = (I^d)^d$, prints: "Matrices are equal"
fcaflint "invers(<ONE>)" output.cxt	prints: "Result is NUL matrix"
fcaflint file.bat output.cxt	Reads the operations from a batch file "file.bat".

Table 2. Examples of FcaFlint command-lines

FcaFlint has been tested on matrices of sizes of up to 50×400. It returns reasonably fast results, with the exception of the transitive closure function, which should only be used for smaller matrices. It should be stressed that FcaFlint is aimed at expert users because using RA requires some expertise.

The second version of FcaFlint intends to support RA operations according to the named perspective described in Section 3.1. In particular, it is fairly easy to implement checks for objects and attributes ensuring that they are compatible. The implementation of "inside" and "outside" conditions is slightly more complicated. The approach that is currently envisioned is to store simple conditions and to stop the program with a warning if the conditions are getting too complex. A warning would tell users that they need to manually check the sets of objects and attributes of their formal contexts and to verify whether the CAS operations that they are attempting to use are actually meaningful.

Websites

1. RA resources: `http://www.upriss.org.uk/fca/relalg.html`
2. FCA website: `http://www.upriss.org.uk/fca/fca.html`

References

1. Ganter, Bernhard, & Wille, Rudolf (1999). *Formal Concept Analysis. Mathematical Foundations.* Berlin-Heidelberg-New York: Springer.
2. Priss, U. (2005). *Establishing connections between Formal Concept Analysis and Relational Databases.* In: Dau; Mugnier; Stumme (eds.), Common Semantics for Sharing Knowledge: Contributions to ICCS 2005, p. 132-145.
3. Priss, Uta (2006). *An FCA interpretation of Relation Algebra.* In: Missaoui; Schmidt (eds.), Formal Concept Analysis: 4th International Conference, ICFCA 2006, Springer Verlag, LNCS 3874, p. 248-263.
4. Priss, Uta; Old, L. John (2006). *An application of relation algebra to lexical databases.* In: Schaerfe, Hitzler, Ohrstrom (eds.), Conceptual Structures: Inspiration and Application, Proceedings of the 14th International Conference on Conceptual Structures, ICCS'06, Springer Verlag, LNAI 4068, p. 388-400.
5. Priss, Uta (2009). *Relation Algebra Operations on Formal Contexts.* In: Proceedings of the 17th International Conference on Conceptual Structures, ICCS'09, Springer Verlag.

A Commentary on Standardization in the Semantic Web, Common Logic and MultiAgent Systems

Doreen Mizzi, Wolfgang Browa, Jeremy Loke and Simon Polovina

Department of Computing /Cultural, Communication & Computing Research Centre
Sheffield Hallam University, Sheffield, United Kingdom
{dmizzi, wbrowa}@hera.shu.ac.uk,{j.loke, s.polovina}@shu.ac.uk

Abstract. Given the ubiquity of the Web, the Semantic Web (SW) offers MultiAgent Systems (MAS) a most wide-ranging platform by which they could intercommunicate. It can be argued however that MAS require levels of logic that the current Semantic Web has yet to provide. As ISO Common Logic (CL) ISO/IEC IS 24707:2007 provides a first-order logic capability for MAS in an interoperable way, it seems natural to investigate how CL may itself integrate with the SW thus providing a more expressive means by which MAS can interoperate effectively across the SW. A commentary is accordingly presented on how this may be achieved. Whilst it notes that certain limitations remain to be addressed, the commentary proposes that standardising the SW with CL provides the vehicle by which MAS can achieve their potential.

1 Introduction

Since Tim Berners-Lee [1] publicly stated his vision of the Semantic Web (SW), there has been much work on devising Web technologies for explicitly annotating the existing data on the Web and for performing intelligent reasoning upon it. One potentially powerful area worthy of investigation is a SW in which software agents will be able to perform tasks and take decisions on behalf of humans. Given the ubiquity of the Web itself and developments in MultiAgent Systems (MAS) [34], the SW offers a global platform that MAS could be brought to bear and allow software agents to intercommunicate across this most wide-ranging arena.

2 Common Logic

The SW includes languages for ontologies and reasoning such as Resource Description Framework (RDF, www.w3.org/RDF), Web Ontology Language (OWL, www.w3.org/TR/owl-features), and TRIPLE (triple.semanticweb.org) [18, 2].

Common Logic (CL) arose from two separate projects that were aimed at developing parallel ANSI standards for conceptual graphs and the Knowledge Interchange Format (KIF) [11]. These two projects eventually merged into a single

ISO project, in order to develop a common abstract syntax for a family of logic-based languages. In fact, CL is now an established International Standard that provides a logic framework for the exchange of data and information amongst distinct computers on an open communication network. ISO/IEC 24707:2007 [19] is the International Standard that describes a selection of logic languages that are directed towards this aim. This international standard ensures that the logic languages that it specifies all share their declarative semantics. Also, every logic language in the CL must be compliant with the generation of first-order logical sentences. All of this must be centred on sharing of data and information between incongruous computer systems. Hayes and Menzel [16] defined a very general model theory for CL, which Hayes and McBride [15] used to define the semantics for the languages Resource Description Framework (RDF) and Web Ontology Language (OWL). In fact, CL is not a single language that specifies a unique set of syntactic forms and hence it does not exclude any other forms of languages. CL allows for a number of so-called dialects to interoperate with the CL common abstract syntax. These dialects can have their own syntax, but must share uniform semantics, so that they can then be expressed using the CL abstract syntax.

Three concrete syntaxes are specified in the standard:

- CGIF - Conceptual Graph Interchange Format which is a text version of conceptual graphs.
- CLIF - Common Logic Interchange Format whose syntax is based on KIF.
- XCL - XML-based notation for Common Logic. This is intended to be a generic dialect for distributing CL content on any network that supports XML [19].

As stated by Sowa[31], CL can also be utilized to map to and from controlled natural languages such as using CLCE (Common Logic Controlled English).

3 CL, Ontologies and the SW

An ever-increasing number of applications are benefiting from the use of ontologies as a way to achieve semantic interoperability among heterogeneous, distributed systems [32]. However, different interpretations may occur across different ontologies that lead to mismatches being made by MAS if they are to interoperate across them. One way forward would be to map the ontologies as suggested by Ding et al. [8]. However, this would result in the numerous combinations of mappings between each of the different ontologies.

A more effective approach would be to map each ontology with CL, as it covers a larger set of first-order logic. As such it provides a general basis for dialects, applications or even networks to determine/establish conformance to CL syntax, hence a basis for meaning-preserving translation [24]. It also makes CLIF a genuine network logic, because CLIF texts retain their full meaning when transported across a network, archived, or freely combined with other CLIF texts (unlike the SW's OWL-DL and GOFOL) [17]. The challenge arises when not

all of the knowledge available in the ontology can be transferred back into CL as first-order logic and therefore this would filter out and reduce some of the information that can be transferred on to other mapped ontologies. The significance of this will depend on the importance of the non-transferable knowledge. It may also be possible to retain this knowledge as, say, comments; it would also explicate these differences and focus research energy onto how they may be in turn be interoperated. Success in this approach has been demonstrated with interoperating ontologies in the Amine conceptual graphs-based software (amine-platform.sourceforge.net) with the SW's Protégé OWL (Web Ontology Language) software (protege.stanford.edu) [27].

4 MAS Interoperability

Central to the development of the SW is the concept of machine-readable transfers of information, marking a distinct move away from the current Web that was, as pointed out by MciLraith et al [23] 'designed primarily for human interpretation and use'. We can of course note that Web Services are designed to provide interoperability between proprietary IT solutions, and have successfully contributed to the growth of electronic business and e-commerce solutions. However, MAS are designed to exist in a far more dynamic way. They need to develop their own interconnections and exist as self regulating open and decentralized applications. In this way, they would avoid the problems of traditional systems including 'software reuse and discovery' [28] while supporting interoperability by implementing semantic structure in a rigorous and standardized way. Fensel et al [9] state that the implementation of Web Services in an interoperable and machine-to-machine understandable manner would have a further impact on e-commerce and enterprise applications integrations. They identify this development as 'a significant, if not the most important, factor in IT'. Web Services are being seen as the technological platform to deliver services to requesters. Agents on the other hand are intelligent systems that are trying to reason within the world they know. They can offer services but also make use of services provided elsewhere in order to reach their goals. Furthermore they could assemble, or 'orchestrate', a series of Web Services on the fly according to their autonomously decided aims at the time. An agent, for example, may assemble individual Web Services into the requisite business processes so that it can successfully negotiate a high-level business deal or transaction. Presently the orchestration (business or otherwise) remains a human endeavour [33].

Meanwhile Leuf [21] draws attention to the fact that even 'the lack of formal compatibility on syntactic as well as semantic levels greatly hinders the ability to accommodate domain interoperability' [21]. Although the ability to rigorously share ontologies is key to 'seamless integration' [20], 'fully automatic derivation of mappings is considered impossible as yet' [5]. In the MAS environment, agents often may not be able to execute their programs properly as expected due to such compatibility issues. Correct interpretation of messages between different agents is essential for successful intercommunication between MAS environments. Gold-

man et al [12] recommends that 'multi-agent systems can be made more robust if they can autonomously overcome problems of miscoordination'. The same viewpoint is emphasized by Leuf [21], who stresses that 'a major goal is to enable agent-based systems to interoperate with a high degree of autonomy, flexibility, and agility - without misunderstanding.' As a result, this gives rise to the need for a more effective representation to describe and reason over the agent's knowledge domain. As noted by Delugach [6] CL is the first standardized effort at developing techniques for interoperability between systems using various representations. Similarly, CL is supported by various existing commercial and open source software tools that can build and deploy ontologies. Such tools were designed to build ontologies compliant with the industry standards prior to the final CL standard [7].

Also, Flores-Mendez [10] describe the work of standards in MAS. The Foundation for Intelligent Physical Agents (FIPA) have developed a series of MAS specifications like Agent Management and Agent Communication Language. FIPA targets to use a minimal framework to manage agents in an open environment. The Object Management Group (OMG) have proposed a model for agent technologies for the collaboration of agent components and agencies using interaction policies. These agencies are then in turn capable to support agent mobility, security and concurrent execution. The Knowledgeable Agent-oriented System (KAoS) approach addresses the standardization of open distributed architecture for software agents to facilitate agent implementations and messaging communication. Bearing commercial background, General Magic focuses on mobile agent technology in e-commerce allowing agents to move, connect, approve and interact in particular instances. Regardless of all these contributions, Flores-Mendez is of the view that practically not much effort has been realized in establishing a standard and acceptable definition of MAS architectures and frameworks. He states that this is possibly due to the belief that both the architectures and frameworks are required to match individual project requirements for the benefits of project efficiency. To allow for forward development of the SW with MAS, there should thus be stronger support for the integration and interoperability of MAS in this direction.

There are also other critical technical issues in enabling software agents to deal with ontologies, such as the ability to achieve a seamless integration with existing Web browsers or user interface tools. To get around these problems, Blois et al [4] has recommended the SemantiCore framework. Another issue highlighted was that agents in MAS execute in a container separated from the containers of other MAS applications. Whilst work is in place to overcome this, such as Jade - a framework implemented in JAVA for MAS in providing JAVA classes for ontology manipulation), Blois et al [4] would still comment that no agents are specifically designed in support of the deployment of MAS to the SW. Like other agent technology developments, the Semantics MAS development also identifies security issues are remain to be resolved. This arises because security aspects are not taken into active consideration during the development cycle. As a result, security vulnerabilities are often exploited or detected only in the stage

of program execution. The FAME Agent-oriented Modelling Language meta-model suggested by Beydoun et al [3], and other previously defined security-aware meta-models, help developers to take a conceptual approach to tackle security requirements of MAS as early as need be during the design stage of the development process. Another advantage would be the introduction of a common identification and validation process to register Agents and accord access rights, and the development and adoption of the WS-I standards in this area (www.ws-i.org).

Nonetheless first order logic based knowledge representation represents a powerful tool for machine-machine interaction. Claimed by Grimm et al [13] as 'the prevalent and single most important knowledge representation formalism.... First-order logic captures some of the essence of human reasoning by providing a notion of logical consequence'. A computer agent is able to deduce and infer data from knowledge statements provided in a logical structure, replicating at least in part human domain expertise. Furthermore, by providing an abstract syntax CL removes the syntactic limitations of traditional first order logic, thus consistently facilitating translation between one language and another as we have stated. Using CL or one of its dialects as a content language will help facilitate the communication between the different agents in the system. CL dialects have been appropriately set up for the purpose of sharing semantics rather than syntax. As raised by Blois et al [4], a compatible or well-suited infrastructure is mandatory for the forward development of semantic MAS. An in all therefore, CL offers a most promising vehicle to MAS interoperability.

5 Preciseness

Given the overall usefulness of CL for MAS on the SW as discussed so far, we can now consider the preciseness of logic in achieving this aim. Sheth et al [29] among others subcategorize types of semantics into distinguishable categories based on how they are constructed and represented. They argue that formal semantics (information or knowledge represented in a highly structured way and in which structures can be combined to represent the meanings of other complex structures), and which would also apply to CL, has had a disproportionately high level of attention even within the SW. Focus is also required on what they view as the two other kinds; implicit semantics (semantics that rely on the analysis of unstructured texts with a loosely defined and less formal structure) and powerful semantics (the representation of imprecise, uncertain, partially true and/or approximate information). In essence, while CL provides a robust level of functionality for MAS interoperability it may, for a significant subtypes of data, not offer sufficient functionality for intelligent machine interaction to replace human domain expertise. Some domain knowledge possessed by human experts is intrinsically complex and may require the more expressive representations of powerful semantics that seeks to allow the representation of degree of membership and degree of certainty. Sheth et al [29]:

'In our opinion, for this vision [the SW] to become a reality, it will be necessary to go beyond RDFS and OWL, and work towards standardized formalisms that support powerful semantics.'

Given that fully functional MAS must be able to process information that is not "precisely axiomatised", subject to degrees or relation and degrees of accuracy then it must be concluded that CL alone is insufficient to fully realize the vision for the SW. There are still as yet "unidentified issues" to be discussed [6]. Notwithstanding a suitable model for "soft computing" that may embody known and unknown issues, the current situation still offers a robust way forward for MAS to interoperate. If that can be achieved without excluding future developments in these relevant fields then the vision of truly interoperable MAS on the SW can still continue to be realized.

6 Further Considerations

We can also note other aspects of CL, such as that it 'also supports some features of higher order logics, such as quantifying over predicates specified in the domain of discourse' [26]. This is highlighted by CL partial support the higher-order logics of Semantics of Business Vocabulary and Rules (SBVR). Section 10.2 of the SBVR specification 'gives a partial mapping of SBVR concepts to OWL and to ISO Common Logic' [22]. In some cases, the mapping is one-to-one, but in many others, a single SBVR concept is represented as a composition of multiple OWL or CL Constructs [22].

Another dimension in the area of specialized digital libraries, where significant improvement has been made over the quality of the information as represented by the SW technologies versus the resources as retrieved by normal search engines [25]. This work also concluded that the SW can leverage the common data model and syntax to ensure the interoperability of resources which is also platform-independent. As a result there are added benefits for MAS to be had in the exchange and collaboration of digital libraries and corporate information repositories across the SW, and on which CL can be brought to bear.

Lastly, no discussion on standardization is complete without reference to approaches that allow unstructured data to be accommodated, hence reducing the need for standards and recognising the Web is built upon myriad formats with various (lack of) structures each serving some individual, ill-compatible purpose. Embedding learning capabilities (e.g. Neural Net) technology that allow intelligent systems to learn like humans from their divergent surroundings provide another approach. Commercial companies like Autonomy (www.autonomy.com) employ such approaches that allows organisations to extract value from the myriad of different systems and information formats within their systems. Autonomy's mixture of propriety Bayesian probability and Shannon theory, allows it to make some 'sense' of the information held in many different formats including voice, video and images. There are also open-source developments such as Unstructured Information Management applications (UIMA, incubator.apache.org/uima). Whilst we can envisage that unstructured data will be an

ongoing feature of the Web, the thrust of the SW augmented with CL for the benefit of MAS remains. Rather we can expect the two approaches (structured and unstructured) to be complimentary activities that can benefit from each other. Structuring and thus automating knowledge allows humans to articulate their thinking, capture and interact with it through the productivity of computers i.e. conceptual structures, identified in the "Knowledge Architectures" theme of ICCS 2007 (www.iccs2007.info). Given also that conceptual structures in the form of conceptual graphs are core to CL, the need for CL on the SW for MAS remains undiminished.

7 Concluding Remarks

Even though relatively new as a standard, CL will aid towards achieving the much needed interoperability between the multitude of existing knowledge representations. Assisted by CL, intercommunication between MAS in on the global, most wide-ranging arena that the SW offers will be a key step in leading to the original vision as proposed by Tim Berners-Lee that the SW will evolve human knowledge [1]. Our work has helped identify the need, the key issues and direction in helping to realise that vision. Given the evidence and discussion that we have provided, we would support the view that additionally standardising the SW with CL provides the vehicle by which MAS can achieve their potential.

Acknowledgements

This work is based upon extensive research conducted by graduate students at Sheffield Hallam University, and we would sincerely like to acknowledge their valuable contributions that have culminated in this paper. We are also indebted to the anonymous reviewers of our originally submitted paper in guiding our thinking for this paper.

References

1. Berners-Lee, T. et al. (2001) The Semantic Web [online] Scientific American Magazine. Last accessed on 20th February, 2009 at http://www.sciam.com/.
2. Berners-Lee, T. et al. (2008). N3Logic: A logical framework for the World Wide Web, Theory and Practise of Logic Programming, Vol. 8(3), p249-269; (AN 14185093).
3. Beydoun, G. et al. (2008). A security-aware metamodel for multi-agent systems (MAS). Information and Software Technology Journal, Vol. 51(5) p 832-845.
4. Blois, M. et al. (2008). Using Agents & Ontologies for Application Development on the Semantic Web. Computer Engineering Department, IME.
5. Conroy et al. (2008) Ontology Mapping Through tagging. International Conference on Complex, Intelligent and Software Intensive Systems. Last accessed on 27th April 2009 at http://ieeexplore.ieee.org.
6. Delugach, H.S., (2007). Towards Conceptual Structures Interoperability using Common Logic. [online] University of Alabama. Last accessed 27 February 2009 at:http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-352/paper3.pdf.

7. Denny, Michael (2004), Ontology Tools Survey, Revisited, [online]. Last accessed 27 February 2009 at: http://www.xml.com/pub/a/2004/07/14/onto.html.
8. Ding, Y. et al., (2001). Ontology management: survey, requirements and directions, IST Project IST-1999-10132 On-to-Knowledge [online]. Last accessed 18 March 2009 at http://www.ontoknowledge.org/downl/del4.pdf.
9. Fensel, Dieter et al., (2007). Enabling Semantic Web Services: The Web Service Modeling Ontology. Springer.
10. Flores-Mendez, Roberto A. (1999). Towards a Standardization of Multi-Agent System Frameworks. University of Calgary. [online]. Last accessed 27 April 2009 at: http://www.acm.org/crossroads/xrds5-4/multiagent.html.
11. Genesereth, M.R., and Fikes, R., (1992). Knowledge Interchange Format. Version 3.0 Reference Manual, Computer Science Department, Stanford University, Stanford, CA.
12. Goldman, C. et al., (2007). Learning to communicate in a decentralized environment. Autonomous Agents and Multi-Agent Systems, Vol. 15 (1), p47-90.
13. Grimm et al. (2007) Knowledge Representation and Ontologies. Logic, Ontologies and Semantic Web Languages in Studer et al. (2007) Semantic Web Services: Concepts, Technology and Applications, pp51-106. Springer, Berlin. Last accessed on 27th April 2009 at http://www.aifb.uni-karlsruhe.de/WBS/phi/pub/kr-onto-07.pdf.
14. Grosof, Benjamin, N., (2004) Representing e-commerce rules via situated courteous logic programs in RuleML. Electronic Commerce Research and Applications, Vol. 3(2004), p2-20.
15. Hayes, P. & McBride, B., (2003). RDF semantics, W3C Technical Report [online]. Last accessed 17 February 2009 at: http://www.w3.org/TR/rdf-mt.
16. Hayes, P. & Menzel, C., (2001). A semantics for the Knowledge Interchange Format. Proceeding of IJCAI 2001 Workshop on the IEEE Standard Upper Ontology, Seattle.
17. Hayes, P. (2008) Using Common Logic. Mapping other KR technologies to CL, Eliminating dongles, The zero case. Presentation delivered at Semtech 2008 workshop. [online]. Last accessed 27 April 2009 at: www.ihmc.us/users/phayes/SemTech2008slides.ppt.
18. Henze, N., et al., (2004) Reasoning and Ontologies for Personalized E-Learning in the Semantic Web. Educational Technology & Society, Vol. 7 (4), p82-97.
19. International Standards Office, (2007). ISO 24707/2007 Information Technology - Common Logic (CL): a framework for a family of logic-based languages. [online] Geneva: ISO. Last accessed 27 April 2009: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39175.
20. Lanzenberger, M. & Sampson, J., (2008). Making Ontologies Talk: Knowledge Interoperability in Semantic Web. IEEE intelligent systems journal, November/ December 2008, p72.
21. Leuf, B., (2006). The Semantic Web - Crafting Infrastructure for Agency. John Wiley & Sons, Ltd.
22. Linehan, Mark H., SBVR Use Cases, IBM T.J. Watson Research Center. [online]. Last accessed 27 April 2009 at: http://www.aaai.org/Papers/Symposia/Spring/2008/SS-08-01/SS08-01-010.pdf.
23. McIlraith, S. et al. (2001). Semantic Web Services. IEEE intelligent systems journal, March/ April 2001, p46.
24. Menzel, C. (2008). Common Logic - Motivations and Some Gentle Theory. Presentation delivered at Semtech 2008 workshop. [online]. Last accessed 27 April 2009 at: http://cl.tamu.edu/.

25. Morales-del-Castillo J.M. et al., (2009). A Semantic Model of Selective Dissemination of Information for Digital Library. Information Technology & Library, American Library Association.
26. Obitko, Marek. (2007).Ontologies and Semantic Web. [online]. Last accessed 2 April 2009 at: http://www.obitko.com/tutorials/ontologies-semantic-web/introduction.html.
27. Polovina, S. et al., (2009). A Practical Exploration of Ontology Interoperability. S. Rudolph, F. Dau, and S.O. Kuznetsov (Eds.): ICCS 2009, LNAI 5662, pp. 247256. Springer-Verlag, Berlin and Heidelberg.
28. Sabou, M. and Pan, J., (2006). Towards semantically enhanced web service repositories. Journal of Web Semantics. Vol 5 (2), June 2007, p142-150 [online]. Available http://www.sciencedirect.com/.
29. Sheth, A. et al., (2005), Semantics for the Semantic Web: The Implicit, the Formal and the Powerful. International Journal on Semantic Web and Information Systems. Idea Group Publishing.
30. Semantics of Business Vocabulary and Business Rules (SBVR), v1.0, OMG Available Specification, January 2008 [online]. Last accessed 2 April 2009 at http://www.omg.org/docs/formal/08-01-02.pdf.
31. Sowa, John F., (2008). VivoMind Intelligence, Inc. 4 March. [online]. Last accessed 2 April 2009 at http://www.jfsowa.com/talks/semtech1.pdf.
32. Tamma, V. et al., (2002). An ontology for automated negotiation. Proc. AAMAS'2002 International Workshop on Ontologies in Agent Systems, 16-20 July, Bologna, Italy.
33. Woods, D. (2006). Enterprise SOA. Beijing; Farnham: O'Reilly.
34. Wooldridge, M. (2009). An Introduction to MultiAgent Systems (Second Edition), Wiley.