A Simple Approach to Abductive Inference using Conceptual Graphs

Maurice Pagnucco

Knowledge Systems Group,
Basser Department of Computer Science,
Madsen Building F09,
University of Sydney,
NSW, 2006, Australia.
email: morri@cs.su.oz.au
fax : +61-2-692-3838

Abstract

Abductive reasoning (or simply abduction) is a form of logical inference that aims to derive plausible explanations for data. The term "abduction" was first coined by Charles Sanders Peirce who distinguished it as a fundamental form of logical inference alongside deduction and induction. Recently, abductive reasoning has begun to attract increasing interest within the field of Artificial Intelligence.

In this paper we attempt to present some preliminary ideas regarding the notion of abduction within the framework of conceptual graphs. We propose a definition of abduction in terms of conceptual graphs. A simple method for determining abductions of a restricted type of conceptual graph is suggested. We also present a process which may be useful in increasing the number of possible abductions determined by this method.

Keywords: Abductive Reasoning, Conceptual Graphs.

1 Introduction

The philosopher Charles Sanders Peirce was the first person to distinguish three fundamental types of logical inference: abduction, deduction and induction. In the field of Artificial Intelligence, deduction and induction have received a considerable amount of attention over the years but only recently has Peirce's theory of abduction[2] begun to attract interest.

Abductive reasoning (or simply abduction) is a form of logical inference that attempts to derive plausible "explanations" for some given data. In fact, abduction is sometimes referred to as "inference to the best explanation"[3, 4]. Essentially, an abductive inference proceeds by generating hypotheses which, when considered together with certain domain information, would account for or "explain" the given data.

For example, suppose you know that everyone who twists their ankle experiences pain in that ankle. If someone comes to you with a painful ankle, you might hypothesise that they have twisted their ankle for that would explain your observation. There may of course be other plausible explanations; the person may be suffering from arthritis in that ankle for instance.

The aim of this paper is to present some preliminary ideas regarding the notion of abduction for conceptual graphs. We shall discuss a reasonably simple method for performing abduction using conceptual graphs. We also suggest a process that may lead to an increase in the number of interesting abductions determined by this method.

In the following section we introduce some preliminary notions together with a more detailed definition of the notion of abduction. Section 3 rephrases this definition in terms of conceptual graphs and presents one method for determining abductions using conceptual graphs. In Section 4 we suggest a process that may allow us to generate more abductions. We present our conclusions and suggestions for future work in Section 5.

2 Preliminaries

We adopt a definition of abduction similar to those commonly found in the literature [5, 8].

Definition 2.1 *An abduction of a formula O with respect to a domain theory Γ is a set of formulae E such that:*

(i) $\Gamma \cup E \vdash O$*;*

(ii) $\Gamma \cup E$ *is consistent (i.e.,* $\Gamma \cup E \nvdash \perp$*).*

Some presentations consider abductions of a set of formulae Δ [5] instead of a single formula O. We adopt this definition in order to simplify the presentation.

Often, the formulae in question are subject to syntactic restrictions. Commonly, this takes the form of requiring them to be in clausal form. In order to mimic this form of abduction we shall place a similar restriction on the graphs used here. We consider graphs representing formulae in clausal form. That is, graphs mapping, under the ϕ-operator, to clauses.

A clause represented as a conceptual graph assumes the following form: $\neg[\neg[u_1] \ldots \neg[u_m]\ v_1 \ldots v_n]$ where each u_i is either a single concept or a relation connected to concepts which are in turn linked by lines of coreference to concepts in the outer context. We shall refer to such graphs as *clausal conceptual graphs*[1]. Although conceptual graphs do not always appear in this form, it may be possible to apply a procedure, such as that proposed by Emond [1], to perform the necessary conversion.

The graphs in Figure 1 present some examples of clausal conceptual graphs. The first graph expresses the fact that if a person twists their ankle, they will experience pain. The second says that a person suffering arthritis in their ankle experience pain in that ankle. The third graph expresses that an old person with a weak ankle will suffer from arthritis in that ankle.

2.1 Resolution using Conceptual Graphs

One notion that will be useful is that of performing "resolution" using conceptual graphs. Sowa [10] notes that the following conceptual graph schema is analogous to resolution $(\neg[u\ v], \neg[\neg[u]\ w] \vdash_{\alpha\ or\ \beta\ rules} \neg[v\ w])$. A more general form of this schema may be represented in the following manner (a proof may be found in Appendix A).

$$\neg[\neg[u]\ w_1 ... w_n] \qquad \neg[u\ v_1 ... v_m]$$

$$\neg[v_1 ... v_m\ w_1 ... w_n]$$

Sowa [10] also notes that graph matching in conceptual graphs is analogous to unification in first-order resolution.

[1] See also [7] for remarks regarding Horn clauses and conceptual graphs.

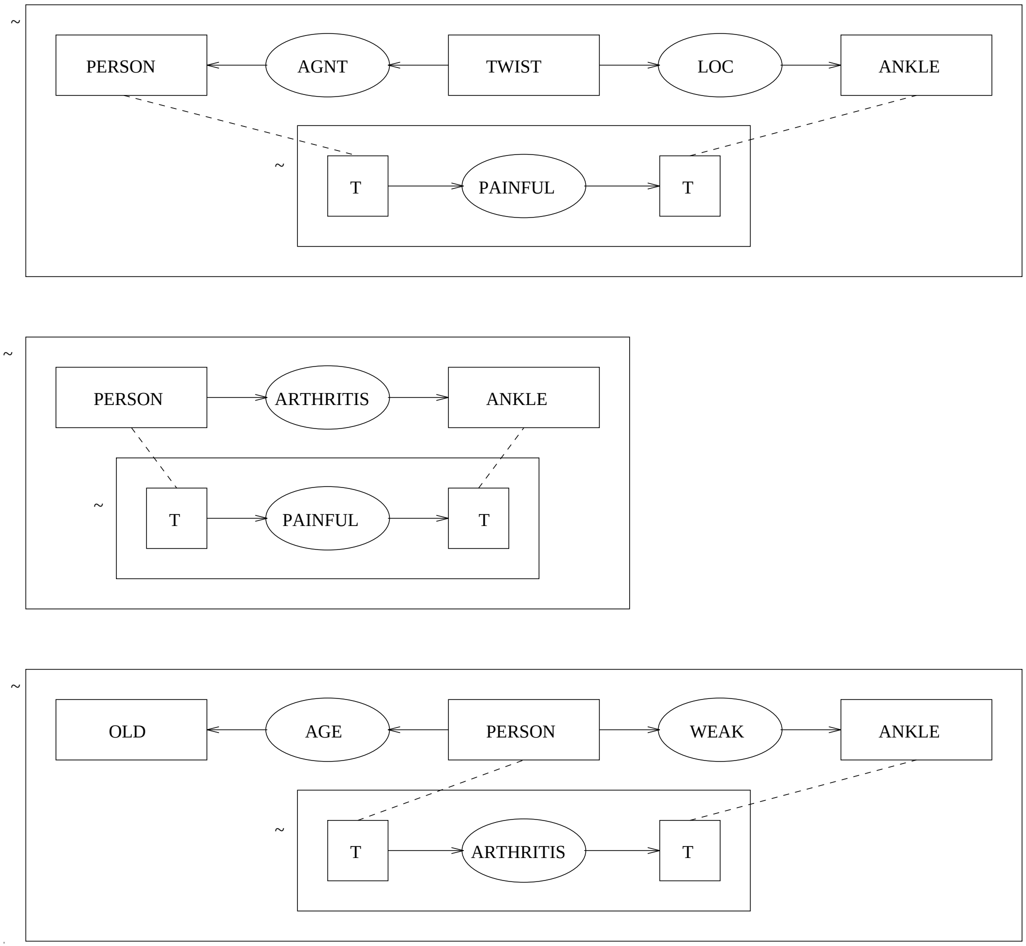

Figure 1: Examples of clausal conceptual graphs.

Bearing this in mind, we can define the restricted notion of a consensus operation (see [6] for instance) on conceptual graphs. First, however, we require the concept of a fundamental clausal conceptual graph.

Definition 2.2 *A fundamental clausal conceptual graph is a conceptual graph* $\neg[\neg[u_1] \ldots \neg[u_m]\ v_1 \ldots v_n]$ *in which no* $u_i = v_j$ *(for any* $1 \leq i \leq m$ *and* $1 \leq j \leq n$*).*

Such a graph would represent a contingent clause.

Definition 2.3 *Given two conceptual graphs* $G_1 = \neg[\neg[t]\ v_1 \ldots v_n]$ *and* $G_2 = \neg[t\ w_1 \ldots w_m]$*, the consensus of* G_1 *and* G_2 *with respect to* t*, is defined as* $CS(G_1, G_2, t) = \neg[v_1 \ldots v_n\ w_1 \ldots w_m]$ *provided* $\neg[v_1 \ldots v_n\ w_1 \ldots w_m]$ *is a fundamental clausal conceptual graph and undefined otherwise.*

In other words, a consensus operation is a restricted form of resolution which is defined when the result is a fundamental clausal conceptual graph.

3 Abduction using Conceptual Graphs

We begin by defining a notion of abduction in terms of conceptual graphs. In accord with the remarks of the previous section, an abduction (in terms of conceptual graphs) is a set of graphs which, when added to a set of graphs denoting our domain knowledge, will allow us to "account" for the given data (a graph) and are also consistent with the domain knowledge graphs.

In this exposition, we consider the sheet of assertion — in particular, the graphs that belong to it — to represent the domain knowledge. So, an abductive inference can be viewed as the process of determining a set of graphs to add to the sheet of assertion in order to prove the given data (using alpha and beta rules for instance) and which are also consistent with the graphs on the sheet of assertion.

Morever, we shall assume that all conceptual graphs are in clausal conceptual graph form (or have been converted into this form). To simplify matters further, we shall only consider abductions of single clausal conceptual graphs here.

Definition 3.1 *An abduction of a conceptual graph* O_G *with respect to the sheet of assertion* SA *is a set of graphs* E_G *such that:*

(i) $SA \cup E_G \vdash_{\alpha,\beta-rules} O_G$

(II) $SA \cup E_G \not\vdash_{\alpha,\beta-rules} \neg[]$

In clausal form logic, if $\Gamma \cup E \vdash O$ for some clause O, then $\Gamma \vdash \neg E \vee O$. So, if E is a set of literals (which is commonly the case — see [9] for instance) then $\neg E \vee O$ is a clause. So, all we need do is look for clauses that are consequences of the domain information and that are subsumed by O and then negate the non-subsumed part of the clause to obtain an abduction.

We can, in fact, apply the same procedure for conceptual graphs. Given some new data in the form of a clausal conceptual graph, $O_G = \neg[\neg[u_1] \ldots \neg[u_m]\ v_1 \ldots v_n]$, if we can find a conceptual graph on the sheet of assertion (or a consequence of graphs on the sheet of assertion) subsumed by O_G — that is, a graph of the form $\neg[\neg[u_1] \ldots \neg[u_m]\ \neg[x_1] \ldots \neg[x_k]\ v_1 \ldots v_n\ w_1 \ldots w_l] \in SA$, then an abduction will consist of the following set of graphs $E_G = \{\neg[x_1], \ldots, \neg[x_k], \neg[\neg[w_1]], \ldots, \neg[\neg[w_l]]\}$ (or, equivalently $E'_G = \{\neg[x_1], \ldots, \neg[x_k], w_1, \ldots, w_l\}$ after repeated application of the double negation alpha rule). At this point, a consistency check needs to be performed to ensure that the proposed abduction is consistent with the graphs on the sheet of assertion.

To see that this is an abduction according to our definition is straightforward. If we resolve each conceptual graph in E_G with the graph from SA above then we will obtain the conceptual graph O_G as desired. The consistency check will ensure that any possible abductions are consistent with the sheet of assertion.

For example, an observation that there is someone suffering from a painful ankle may be expressed by the graph `[PERSON]->(PAINFUL)->[ANKLE]`. However, this graph is not in clausal conceptual graph form but an equivalent graph is provided in Figure 2 (after a double negation has been applied to the centre graph). Using this graph and supposing the sheet of assertion contains the graphs in Figure 1, we can apply our abductive procedure to obtain the three possible abductions in Figure 3. These graphs have had double negations removed, by an application of the double negation rule to each, in order to enhance readability. The last of these may not in fact be generated using the current method but will be possible after using the consensus algorithm to be presented in Section 4. It can be generated when the

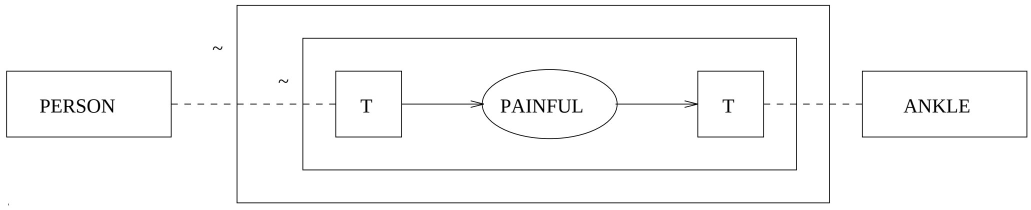

Figure 2: Observation in equivalent clausal conceptual graph form.

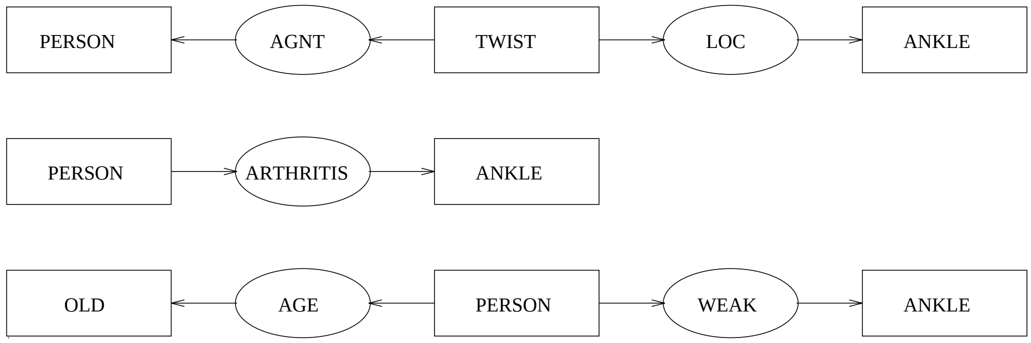

Figure 3: Possible abductions (double negations removed).

clausal conceptual graph in Figure 4 is added to the sheet of assertion and the graph in Figure 2 is used as the new data

4 Prime Implicates and Conceptual Graphs

If the sheet of assertion is not deductively closed, then it is possible that there are some interesting abductive inferences that will not be determined by this process. In clausal form logic, prime implicates are calculated in order to allow calculation of a class of "minimal" abductions. One method for determining prime implicates is known as Tison's method [11]. We present a variant of it here[2].

We let $\mathcal{A}$ represent the graphs from the sheet of assertion SA converted into clausal conceptual graph from.

[2]We do not claim, however, that it will produce any sort of "minimal" abductions for conceptual graphs but it will produce extra graphs which may lead to the generation of further abductions which may be argued to be of interest.

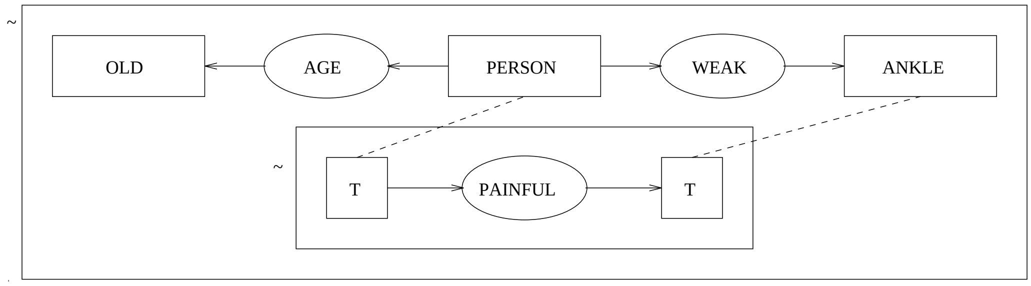

Figure 4: Example of a graph generated by the consensus method.

Algorithm:

> `For each subgraph` v_i `in some graph` $G \in \mathcal{A}$,
> $G = \neg[v_1 \ldots v_m \ \neg[u_1] \ldots \neg[u_n]], (1 \leq i \leq m)$ **do begin**.
>
> `For every pair of graphs` $G_1, G_2 \in \mathcal{A}$, `if` $CS(G_j, G_k, v_i)$ `is defined, add it to` $\mathcal{A}$
>
> `Delete any` $G \in \mathcal{A}$ `such that there is a` $G' \in \mathcal{A}$ `where` G' `subsumes` G `(i.e.,` $G' \vdash_{\alpha,\beta-rules} G$`).`
>
> **end**

An example of a graph generated by this consensus method, using the latter two graphs in Figure 1, is presented in Figure 4.

5 Conclusions and Future Work

In this paper we have suggested a definition of abduction involving conceptual graphs. A simple method for determining abductions of conceptual graphs based on graph matching was provided. A consensus-based algorithm (a variant of Tison's algorithm) was also presented and allows the generation of other graphs which may lead to the determination of further abductions.

A number of problems with the current approach still exist. There is no criteria for selecting the "best" abduction from among those derived. We are also restricted to using clausal conceptual graphs by the procedure given here. Although procedures to convert conceptual graphs into this form may exist, it would still be advantageous to have a general account of abduction for conceptual graphs without relying on syntactic restrictions on the graphs. Such an account might be closer in spirit to the alpha and beta rules rather than the graph matching approach presented here. The syntactic transformation into clausal conceptual graph form may in fact result in the loss of contextual information. This restriction may also lead to non-intuitive procedures such as the transformation of the "observation graph" (Figure 2 in our example into clausal conceptual graph form in order to perform abduction. Another problem is whether a sufficient number of new graphs are introduced by the variant of Tison's method presented, to allow a rich enough set of abductions to be inferred using the procedure outlined here.

So, there is still a lot of work remaining to be done. Notions of minimality, triviality and specificity as well as some form of selection criteria would all be useful. These notions would help in classifying the classes of abductions determined — for instance, are the class of abductions determined by the procedure suggested here minimal in some way? (It is unlikley that this will be the case without some form of subsumption checking.) Another area of investigation is whether subgraph matching would yield interesting results. Considerations of equivalent logical forms tends to suggest that this may well be significant.

Acknowledgements

The author would especially like to thank Norman Foo and Abhaya Nayak for their invaluable comments regarding the contents and presentation of this paper. The author is supported by an Australian Postgraduate Research Award.

References

[1] B. Emond. Operations on conceptual structures and Peirce's system of existential graphs. In *Proceedings of the First International Conference on Conceptual Structures*, pages 238–253, Quebec, 1993.

[2] K. T. Fann. *Peirce's Theory of Abduction*. Martinus Nijhoff, The Hague, Holland, 1970.

[3] G. H. Harman. Inference to the best explanation. *Philosophical Review*, 74:88–95, January 1965.

[4] G. H. Harman. Enumerative induction as inference to the best explanation. *The Journal of Philosophy*, 65(18):529–533, September 1968.

[5] P. Jackson. Propositional abductive logic. In *Proceedings of the Seventh Conference of the Society for the Study of Artificial Intelligence and Simulation of Behaviour*, pages 89–94, 1989.

[6] A. Kean. A formal characterisation of a domain independent abductive reasoning system. Technical Report HKUST-CS93-4, Department of Computer Science, The Hong Kong University of Science and Technology, March, 1993.

[7] M. Pagnucco and N. Y. Foo. Inverting resolution with conceptual graphs states. In *Proceedings of the First International Conference on Conceptual Structures*, pages 238–253, Quebec, 1993.

[8] D. Poole. A logical framework for default reasoning. *Artificial Intelligence*, 36:27–47, 1988.

[9] R. Reiter and J. de Kleer. Foundations of assumption-based truth maintence systems: Preliminary report. In *Proceedings of the National Conference in Artificial Intelligence*, pages 183–188, 1987.

[10] J. F. Sowa. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, Reading, MA, 1984.

[11] P. Tison. Generalization of consensus theory and application to the minimization of boolean functions. *IEEE Transactions on Electronic Computers*, EC-16:100–105, 1967.

A Resolution Using Conceptual Graphs

In this section we present a proof of the following sequent:
$\neg[\neg[u]w_1 ... w_n], \neg[uv_1 ... v_m] \vdash_{\alpha rules} \neg[v_1 ... v_m w_1 ... w_n]$

1	$\neg[uv_1...v_m]$	HYP
2	$\neg[\neg[u]w_1...w_n]$	HYP
3	$\neg[\neg[u\neg[uv_1...v_m]]w_1...w_n]$	1,2 iteration
4	$\neg[\neg[u\neg[v_1...v_m]]w_1...w_n]$	3 deiteration (u)
5	$\neg[\neg[\neg[v_1...v_m]]w_1...w_n]$	4 erasure (u)
6	$\neg[v_1...v_m w_1...w_n]$	5 double negation