Memorial
University of Newfoundland

Technical Report #2008-03
Department of Computer Science
Memorial University of Newfoundland
St. John's, NL, Canada

ON THE COMPUTATIONAL COMPLEXITY OF ANALOGY DERIVATION IN STRUCTURE-MAPPING THEORY

by

Patricia Evans[1], Jason Gedge[2], Moritz Müller[3], Iris van Rooij[4], Todd Wareham[5]

[1] Faculty of Computer Science, University of New Brunswick - - Fredericton, Fredericton, NB, Canada

[2] Department of Computer Science, Memorial University of Newfoundland, St. John's, NF, Canada

[3]Abteilung für Mathematishe Logik, Albert-Ludwigs-Universität Freiburg, Freiburg, Germany

[4]Nijmegen Institute for Cognition and Information, Radboud University Nijmegen, Nijmegen, The Netherlands

[5]Department of Computer Science, Memorial University of Newfoundland, St. John's, NL, Canada

Department of Computer Science
Memorial University of Newfoundland
St. John's, NF, Canada A1B 3X5

January 2008

Technical Report #2008-03
Department of Computer Science
Memorial University of Newfoundland
St. John's, NL, Canada

On the Computational Complexity of Analogy Derivation in the Structure-Mapping Framework

Patricia Evans[1], Jason Gedge[2], Moritz Müller[3], Iris van Rooij[4], and Todd Wareham[2]

[1]School of Computer Science, University of New Brunswick
Fredericton, NB, Canada
[2]Department of Computer Science, Memorial University of Newfoundland
St. John's, NL, Canada
[3] Abteilung für Mathematische Logik, Albert-Ludwigs-Universität Freiburg
Freiburg, Germany
[4] Nijmegen Institute for Cognition and Information (NICI), Radboud University Nijmegen
Nijmegen, The Netherlands

Zusammenfassung

Gentner's Structure-Mapping Theory of analogy derivation (SMT) and its implementation in the Structure Mapping Engine (SME) have been very influential in Cognitive Science and Artificial Intelligence. Motivated by both the need for SME to handle realistic input sizes and claims that SMT is computationally intractable, the original exhaustive-search strategy in SME was replaced in later versions by a heuristic. However, computing optimal solutions in SMT (and hence SME) may yet be tractable for certain classes of inputs that occur in practice, making heuristics unnecessary in those situations. In this technical report, we give the first formal statement of the optimization problem of deriving optimal structure-mappings in SMT, and use parameterized complexity to investigate the polynomial-time algorithmic possibilities for this problem relative to a wide variety of qualitative and quantitative restrictions on input and output structure.

Supplementary Material for:

Identifying Sources of Intractability in Cognitive Models: An Illustration using Analogical Structure Mapping

Iris van Rooij, Patricia Evans, Moritz Müller, Jason Gedge, Todd Wareham

1 Definitions

Digraphs

A *directed graph* or *digraph*, for short, G is a pair (V, A) of a nonempty set of *vertices* V and a set of *arcs* $A \subseteq V^2$ of ordered pairs (v, w) of two distinct vertices $v, w \in V$. Such a pair is informally understood as an arc pointing from v to w. A digraph is *acyclic*, or a DAG, if and only if it does not contain directed cycles. A *subdag* of DAG G is a DAG $G' = (V', A')$ such that $A' \subseteq A$ and $V' \subseteq V'$. We call G' *induced (by V')* if and only if $A' = A \cap (V' \times V')$.

Let $G = (V, A)$ be a DAG. A *root* of G is a vertex with no incoming arcs. A *leaf* of G is a vertex with no outgoing arcs. A vertex which is not a leaf is *internal*. A vertex v is a *child* of a vertex w if and only if $(w, v) \in A$. A vertex v is a *descendent* of a vertex w if and only if there is a directed path from w to v. The *height* of G is the length (number of arcs) of a longest directed path in G. The graph *underlying* G is the (undirected) graph (V, E) with $E = \{\{v, w\} \mid (v, w) \in A\}$, i.e. the graph obtained when we forget the direction of the arcs in G. A *component* of G is a subdag induced by the vertices of a component of the graph underlying G.

A *directed tree* is a DAG such that for any two vertices $v, w \in V$ there is at most one directed path from v to w. A *poly-tree* is a DAG G such that the graph underlying G is a forest.

Concept graphs

A *concept graph* is a quadrupel $(G, \lambda_A, \lambda_B, \lambda_P)$ for a DAG $G = (V, A)$ and functions $\lambda_A, \lambda_B, \lambda_P$ called *labelings* such that

1. $\lambda_A : A \to \mathbb{N}$,
2. λ_B is injective and defined on the leafs of G,
3. λ_P is defined on the internal vertices of G,
4. If v is an internal vertex, then λ_A either enumerates the set of arcs leaving v or is constantly 0 on this set. In the first case v is *ordered*, in the second *unordered*.
5. for internal vertices u, v with $\lambda_P(u) = \lambda_P(v)$ the following holds:
   (a) either both u and v are ordered or both u and v are unordered,
   (b) u and v have the same number of children in G,

(c) $\{(v', \lambda_A(v, v')) \mid (v, v') \in A\} \neq \{(u', \lambda_A(u, u')) \mid (u, u') \in A\}$.

Usually we denote the range of λ_B by B and the range of λ_P by P. Elements of B are called *entities,* those of P *predicates* or *relations*. A predicate is *ordered* if and only if at least one vertex (equivalently: all vertices) labeled with it is ordered. A concept graph is *ordered* (*unordered*) if and only if all its vertices are ordered (unordered).

Analogy morphisms

Let $G = (V, A)$ and $G' = (V', A')$ be two DAGs. A *subdag isomorphism from G to G'* is an isomorphism f of a subdag of G onto a subdag of G'. We write $G_f = (V_f, A_f)$ for the subdag on the domain of f and call it the *subdag associated with f*.

Let $\mathcal{G} := (G, \lambda_A, \lambda_P, \lambda_B)$ and $\mathcal{G}' = (G', \lambda_{A'}, \lambda_{P'}, \lambda_{B'})$ be two concept graphs. An *analogy morphism* of $\mathcal{G}$ and $\mathcal{G}'$ is a subdag isomorphism from G to G' satisfying the following three conditions:

1. for all $v \in V_f$ also all children of v in G are in V_f.
2. $\lambda_{P'}(f(v)) = \lambda_P(v)$ for all $v \in V_f$.
3. $\lambda_{A'}((f(v), f(w))) = \lambda_A((v, w))$ for all $(v, w) \in A_f$.

The value of analogy morphisms

Let a concept graphs $\mathcal{G} = (G, \lambda_A, \lambda_P, \lambda_B)$ be given. Relative to a function *pval* : $P \to \mathbb{N}$ and two naturals *lm*, *trd* $\in \mathbb{N}$ we associate a *valuation val* mapping vertices of G to a *value* in $\mathbb{N}$. This function is defined inductively over the height of the vertex in G: the value of a vertex v is

$$\mathit{match}(v) + \sum_{(w,v) \in A} \mathit{trd} \cdot \mathit{val}(w)$$

where $match(v)$ is $\mathit{pval}(\lambda_P(v))$ if v is an internal vertex and lm if v is a leaf. The value $\mathit{val}(G')$ of a subdag $G' = (V', A')$ of G is $\sum_{v \in V'} \mathit{val}(v)$. The value of an analogy morphism f of two concept graphs is $\mathit{val}(G_f)$.

It is easy to see that the value of a given analogy morphism between two concept graphs can be computed in time polynomial in the size of the concept graphs.

The problem

The NP-optimization problem is

> *Input:* two concept graphs $\mathcal{G}$ and $\mathcal{G}'$, a function *pval* : $P \to \mathbb{N}$, where P are the predicates in G, and naturals *lm, trd* $\in \mathbb{N}$.
>
> *Solutions:* all analogy morphisms between $\mathcal{G}$ and $\mathcal{G}'$
>
> *Cost:* the valuation *val* associated with *pval, lm, trd*.
>
> *Goal:* maximization.

The associated decision problem is

Input: two concept graphs $\mathcal{G}$ and $\mathcal{G}'$, a function *pval* : $P \to \mathbb{N}$, where P are the predicates in $\mathcal{G}$, naturals *lm, trd* $\in \mathbb{N}$ and a natural $k \in \mathbb{N}$

Question: is there an analogy morphism between $\mathcal{G}$ and $\mathcal{G}'$ of value at least k?

Here it is understood that "value" refers to the valuation associated with *pval*, *lm*, *trd*.

In our work we are concerned with the following slightly simplified version. Of course intractability of this simplified version immediately implies intractability of the non-simplified version.

SMAD	
Input:	two concept graphs $\mathcal{G}$ and $\mathcal{G}'$.
Problem:	is there an analogy morphism between $\mathcal{G}$ and $\mathcal{G}'$ of value at least k?

Here it shall be understood that value refers to the valuation associated with the function *pval* which is constantly one and the constants *lm* $=$ *trd* $= 1$. We denote this valuation by *val*.

2 Complexity

Classical complexity

SUBGRAPH ISOMORPHISM	
Input:	two graphs G and H.
Problem:	is H isomorphic to a subgraph of G?

SUBFOREST ISOMORPHISM is the restriction of SUBGRAPH ISOMORPHISM to instances where G is a tree and H is a forest.

Lemma 1 *There is a polynomial time reduction from* SUBGRAPH ISOMORPHISM *to* SMAD.

Proof: Let $G = (V, E)$ be a graph. We define the concept graph

$$\mathcal{C}(G) = (C(G), \lambda_A, \lambda_P, \lambda_B)$$

as follows. The DAG $C(G)$ has vertices $V \cup E$ and arcs

$$A := \{(e, v) \mid v \in e \in E\}.$$

λ_A is constantly 0, λ_P is constantly p (for some predicate p) on E and λ_B is the identity on V.

Let (G, H) be an instance of SUBGRAPH ISOMORPHISM. Then if f is an isomorphism from H onto a subgraph of G, then f' is an analogy morphism between $\mathcal{C}(H)$ and $\mathcal{C}(G)$, where f' is defined as follows: it maps all entities (vertices of H) as f

does and additionally maps an edge $\{h, h'\}$ of H (which is also a vertex in $C(H)$) to $\{f(h), f(h')\}$ (a vertex in $C(G)$). The value of f' is the value of $C(G)$.

Conversely, if f' is an analogy morphism between $\mathcal{C}(H)$ and $\mathcal{C}(G)$ with value $val(C(H))$, then its domain is the set of all vertices of $C(H)$. Hence its restriction to the vertices of H is an isomorphism to a subgraph of G - why? To see this let $\{h, h'\}$ be an edge of H. This is a vertex in $C(H)$. Since f' preserves predicates, this vertex is mapped to a vertex in $C(G)$ which is an edge $\{g, g'\}$ of G. By defnition of analogy morphisms, then $\{f(h), f(h')\}$ equals $\{g, g'\}$, and so is an edge of G.

It follows that $(G, H) \mapsto (\mathcal{C}(H), \mathcal{C}(G), val(C(H)))$ defines a polynomial time reduction as claimed. □

It is easy to see that SMAD $\in$ NP. Because SUBGRAPH ISOMORPHISM is famously NP-complete, it follows

Corollary 2 SMAD *is NP-complete.*

The reduction given in the previous Lemma is robust enough to survive under various restrictions. For example, observe that if a graph G is a forest, then $C(G)$ is a poly-tree. It is well-known that SUBFOREST ISOMORPHISM is NP-complete. It follows that

Corollary 3 SMAD *restricted to instances where the given concept graphs are polytrees is NP-complete.*

Parameterized complexity

In the paper we make several tractability and intractability claims numbered from 1 to 6. We prove them subsequently. All proofs of intractability use some common assumption from parameterized complexity. The assumption that W[1] $\neq$ FPT is strong enough for all our purposes.

Claim 1 SMAD *is fp-intractable for parameter set* $\{h, a, f, s\}$.

Proof: It is enough to show that SMAD is NP-hard even when instances are restricted to those where all parameters are required to be bounded by a constant. Then it follows that the parameterized problem is complete for the huge class paraNP [1, Theorem 2.14] and is thus fixed-parameter tactable if and only if P $=$ NP.

This in turn follows by a reduction due to Veale et al. [2] from the NP-complete problem 3-DIMENSIONAL MATCHING which produces concept graphs such that $h = 1, a = 2, f = 1$ and $s = 0$. □

Remark 4 As a matter of fact, the reduction in [2] produces ordered concept graphs.

Claim 2 SMAD *is fp-intractable for parameter set* $\{n/h\}$.

Here the parameter is to be understood to be $\max\{n_1/h_1, n_2/h_2\}$.

Proof: For a concept graph $\mathcal{G} = (G, \lambda_A, \lambda_P, \lambda_B)$ define $\tilde{\mathcal{G}}$ as follows: say G has n vertices. We add to G a directed path with n vertices and an arc from the leaf of this path to a leaf of G. We label each new vertex with an own new predicate. All new arcs get label 1. Then $\tilde{\mathcal{G}}$ has $2n$ vertices and height n. Thus the parameter of this instance is at most 2.

An instance $(\mathcal{G}, \mathcal{G}', k)$ of SMAD is equivalent to $(\tilde{\mathcal{G}}, \tilde{\mathcal{G}}', k)$ (provided the new predicate labels chosen in the construction of $\tilde{\mathcal{G}}$ and $\tilde{\mathcal{G}}'$ are different) because no analogy morphism between $\tilde{\mathcal{G}}$ and $\tilde{\mathcal{G}}'$ can involve some of the new vertices.

But the instance $(\tilde{\mathcal{G}}, \tilde{\mathcal{G}}', k)$ has parameter at most 2, so paraNP-hardness follows as in the previous proof. □

Remark 5 The construction above preserves the property of being ordered, i.e. if $\mathcal{G}$ is ordered, then so is $\tilde{\mathcal{G}}$.

Claim 3 SMAD *is fp-tractable for parameter set* $\{n_1\}$.

Proof: As we have explained in the paper, this is trivial. □

Claim 4 SMAD *is fp-intractable for parameter set* $\{n_2, r, h, a, p\}$.

Proof: It is well-known that the parameterized problem

p-CLIQUE	
Input:	a graph G and a natural $k \in \mathbb{N}$.
Parameter:	k.
Problem:	does G contain a clique with k elements?

is W[1]-hard. It thus suffices to give a parameterized reduction from this problem.

Let C_k be a clique with k vertices. G has a k clique if and only if (G, C_k) is a "yes" instance of SUBGRAPH ISOMORPHISM, hence by Lemma 1, if and only if $(C_k, G, \mathit{val}(C_k))$ is a "yes" instance of SMAD. The parameters of the instance produced are $n_2 = k, a = 2, p = r = h = 1$, all in $O(k)$. □

Claim 5 SMAD *is fp-tractable for parameter set* $\{o\}$.

Proof: Let $(\mathcal{G}, \mathcal{G}', \ell)$ be an instance of SMAD. Let k denote the maximum number of leafs in one of the given concept graphs.

Let F be the set of bijections between a subset of leafs of G and a subset of leafs of G'. Let $g \in F$. We stepwise extend this morphism. For each vertices v of level one in G there is at most one vertex v' of level one in G' such that extending g by mapping v to v' is an analogy morphism. We make all possible such extensions. Then we proceed in the same way with the vertices in level two and so on. This way we generate in polynomial time an analogy morphism with the best possible value among those whose restriction to the leafs equal g.

We compute this value for each $g \in F$ and accept if we find a value $\geq \ell$. Doing this amounts to $|F|$ times a polynomial time computation. Because $|F|$ can be effectively bounded in k, this is fpt time. □

Remark 6 Claim 5 in the paper is weaker than what we prove here - in the paper we claimed the result only for ordered concept graphs.

Claim 6 *Claim 1 to 5 hold true when* SMAD *is restricted to instances with ordered concept graphs only.*

Proof: By Remarks 4 and 5 we are left to verify this for Claim 4. There in the proof we constructed an instance with unordered concept graphs $C(C_k)$ and $C(G)$.

Fix an arbitrary linear order $<$ on the vertices of G. We transform $C(G)$ to an ordered concept graph $C(G)'$ by labeling an arc $(\{v, v'\}, v)$ in $C(G)$ (for an edge $\{v, v'\}$ of G) with 1 if $v < v'$ and with 2 otherwise. Clearly any analogy morphism between *any* ordered version of $C(C_k)$ and $C(G)'$ is also an analogy morphism between $C(C_k)$ and $C(G)$ of the same value. Conversely if there is an analogy morphism between $C(C_k)$ and $C(G)$ then this is also an analogy morphism between $C(G)'$ and *some* ordered version of $C(C_k)$. Thus by the proof of Claim 4, we conclude that G has a k clique if and only if there is an ordered version $C(C_k)'$ of $C(C_k)$ such that $(C(C_k)', C(G)', \mathit{val}(C(C_k)))$ is a "yes" instance of SMAD.

For the sake of contradiction assume that there is an fpt algorithm $\mathbb{A}$ solving SMAD with parameter set $\{n_2, r, h, a, p\}$. We get a contradiction by deriving that p-CLIQUE would then also be fixed-parameter tractable. By the latter condition of the above equivalence we get an fpt algorithm solving p-CLIQUE by simulating $\mathbb{A}$ on input $(C(C_k)', C(G)', \mathit{val}(C(C_k)))$ for all possible ordered versions $C(C_k)'$ of $C(C_k)$. As the number of such ordered versions can be effectively bounded in k, this amounts to an fpt running time. $\square$

Literatur

[1] J. Flum and M. Grohe, *Parameterized complexity theory*, Springer, 2006.

[2] T. veale, D. O'Donoghue, M. T. Keane, *Computability as a limiting cognitive constraint:Complexity resources in metaphor comprehension about which cognitive linguistics should be aware*, in E. M. Higara, C. Sinha, S. Wilcox (eds.), *Cultural, psychological and typoogical issues in cognitive linguistics*, John Benjamins, Amsterdam, 1999.