On maximum spanning DAG algorithms for semantic DAG parsing

Natalie Schluter
Department of Computer Science
School of Technology, Malmö University
Malmö, Sweden
natalie.schluter@mah.se

Abstract

Consideration of the decoding problem in semantic parsing as finding a maximum spanning DAG of a weighted directed graph carries many complexities that haven't been fully addressed in the literature to date, among which are its actual appropriateness for the decoding task in semantic parsing, not to mention an explicit proof of its complexity (and its approximability). In this paper, we consider the objective function for the maximum spanning DAG problem, and what it means in terms of decoding for semantic parsing. In doing so, we give anecdotal evidence against its use in this task. In addition, we consider the only graph-based maximum spanning DAG approximation algorithm presented in the literature (without any approximation guarantee) to date and finally provide an approximation guarantee for it, showing that it is an $O(\frac{1}{n})$ factor approximation algorithm, where n is the size of the digraph's vertex set.

1 Introduction

Recent research in semantic parsing has moved attention towards recovering labeled digraph representations of the semantic relations corresponding to the linguistic agendas across a number of theories where simple tree representations are claimed not to be expressive enough to capture sentential meaning. As digraph structures presented in predominant semantic graph databases are mainly acyclic, the semantic parsing problem has sometimes become associated with a maximum spanning directed acyclic graph (MSDAG) decoding problem (McDonald and Pereira, 2006; Sagae and Tsujii, 2008; Titov et al., 2009), in analogy and perhaps as a generalisation of the maximum spanning tree decoding problem for syntactic dependency parsing.

The appropriateness of finding the MSDAG in decoding for semantic parsing has, however, never been fully motivated, and in fact carries more complexities than that of maximum spanning tree (MST) decoding for syntactic parsing. In this paper, we discuss the appropriateness of MSDAG decoding in semantic parsing, considering the possible objective functions and whether they match our linguistic goals for the decoding process. Our view is that they probably do not, in general.

In addition to the problem of not being sufficiently synchronised with our linguistic intuitions for the semantic parsing decoding problem, the MSDAG problem itself carries with it its own complexities, which are still in the process of becoming more understood in the algorithms research community. McDonald and Pereira (2006) claim that the MSDAG problem is **NP**-hard, citing (Heckerman et al., 1995); however, there is no MSDAG problem in this latter work, and no explicit reduction to any problem presented in (Heckerman et al., 1995) has been published to date. We point out that Schluter (submitted) explicitly provides a linear reduction to MSDAG from the problem of finding a minimum weighted directed multicut (IDMC), showing MSDAG's **NP**-hardness; this reduction also yields a result on the approximability of MSDAG, namely that it is **APX**-hard. We show in this paper that the approximation algorithm presented without any approximation guarantee in (McDonald and Pereira, 2006) is, in fact, a $O(\frac{1}{n})$ factor approximation algorithm, where n is the size of the graphs vertex set. This is not particularly surprising given the problem's **APX**-hardness.

Following some preliminaries on weighted digraphs (Section 2), we make the MSDAG problem precise through a discussion of the objective function in question and briefly question this ob-

Proceedings of the ACL 2014 Workshop on Semantic Parsing, pages 61–65,
Baltimore, Maryland USA, June 26 2014.

jective function with respect to decoding in semantic parsing (Section 3). Finally, we discuss the only other graph-based (approximation) algorithm in the literature and prove its approximation guarantee (Section 4), followed by some brief conclusions (Section 5).

2 Preliminaries

A *directed graph* (or *digraph*) G is a pair (V, E) where V is the set of *nodes* and E is the set of (*directed*) *edges*. $E \subset V \times V$ is a set of ordered pairs of vertices. For $u, v \in V$, if $(u, v) \in E$, then we say there is an "edge from u to v". If there is any confusion over the digraph we are referring to, then we disambiguate by using the notation $G := (E(G), V(G))$.

If all edges $e \in E(G)$ are associated with a real number, a weight $w : E(G) \to \mathbb{R}$, then we call the digraph *weighted*. In this paper, all digraphs are weighted and weights are positive.

For a subset of edges U of a weighted digraph, we set $w(U) := \sum_{e \in U} w(e)$. Similarly, for a weighted digraph G, we set $w(G) := \sum_{e \in E(G)} w(e)$.

We denote the size of a set S, by $|S|$.

For $G = (V, E)$, let $u_1, \ldots, u_k \in V$ and $(u_i, u_{i+1}) \in E$, for each $i \in [k-1]$, then we say that there is *path* (also *directed path*, or *dipath*) of length $(k-1)$ from u_1 to u_k in G. If also $(u_k, u_1) \in E$, then we say that $u_1, u_2, \ldots, u_k, u_1$ is a *cycle* of length k in G.

A *directed acyclic graph* (DAG) is a directed graph with no cycles. There is a special kind of DAG, which has a special node called a *root* with no incoming edges and in which there is a unique path from the root to all nodes; this is called a *tree*.

Finally, a *tournament* is a digraph in which all pairs of vertices are connected by exactly one edge. If, in addition, the edges are weighted, then we call the digraph a *weighted tournament*.

3 Objective functions for finding an MSDAG of a digraph

We first make precise the objective function for the MSDAG problem, by considering two separate objective functions, one additive and one multiplicative, over the weights of edges in the optimal solution:

$$D^* := \underset{D \text{ a spanning DAG of } G}{\arg\max} \sum_{e \in E(D)} w(e), \text{ and} \quad (1)$$

$$D^* := \underset{D \text{ a spanning DAG of } G}{\arg\max} \prod_{e \in E(D)} w(e). \quad (2)$$

Maximising Equation (2) amounts to concurrently minimising the number of edges of weight less than 1 in the optimal solution and maximising the number of edges of weight at least 1. In fact, if all edge weights are less than 1, then this problem reduces to finding the MST. However, the objective in semantic parsing in adopting the MSDAG problem for decoding is to increase power from finding only MSTs. Therefore, this version of the problem is not the subject of this paper. If a graph has even one edge of weight greater than 1, then all edges of lesser weights should be discarded, and for the remaining subgraph, maximising Equations (1) or (2) is equivalent.

Maximising Equation (1) is complex and under certain restrictions on edge weights may optimise for simply the number of edges (subject to being a DAG). For example, if the difference between any two edge weights is less than $\frac{1}{|E(G)|} \times w(e)$ for the smallest weighted e in $E(G)$, then the problem reduces to finding the spanning DAG with the greatest number of edges, as shown by Proposition 1.

Proposition 1. *Let G be a weighted digraph, with minimum edge weight M. Suppose the difference in weight between any two edges of G is at most $\frac{1}{|E(G)|} \times M$. Then an MSDAG for G maximises the number of edges of any spanning DAG for G.*

Proof. Suppose D_1, D_2 are spanning DAGs for G, such that (without loss of generality) $|E(D_1)| = |E(D_2)| + 1$, but that D_2 is an MSDAG and that D_1 is not. We derive the following contradiction.

$$\begin{aligned} w(D_2) &= \sum_{e \in E(D_2)} w(e) \\ &\leq |E(D_2)| \cdot M + |E(D_2)| \cdot \left(\frac{1}{|E(G)|} \cdot M \right) \\ &< |E(D_2)| \cdot M + M \\ &= |E(D_1)| \cdot M \\ &\leq \sum_{e \in E(D_2)} w(e) \\ &= w(D_1) \end{aligned}$$

□

Admittedly, for arbitrary edge weights, the relation between the sum of edge weights and number of edges is more intricate, and it is this problem that we refer to as the MSDAG problem in this paper. However, the maximisation of the number of edges in the MSDAG does play a role when

using Equation (1) as the objective function, and this may be inappropriate for decoding in semantic parsing.

3.1 Two linguistic issues of in MSDAG decoding for semantic parsing

We can identify two important related issues against linguistic motivation for the use of MSDAG algorithms in decoding for semantic parsing. The first problem is inherited from that of the arc-factored model in syntactic parsing, and the second problem is due to MSDAGs constrained maximisation of edges discussed above.

In the arc-factored syntactic parsing paradigm, it was shown that the MST algorithm could be used for exact inference (McDonald et al., 2005). However, one problem with this paradigm was that edges of the inferred solution did not linguistically constrain each other. So, for example, a verb can be assigned two separate subject dependents, which is linguistically absurd. Use of the MSDAG algorithm in semantic parsing corresponds, in fact, to a generalisation of the arc-factored syntactic parsing paradigm to semantic parsing. As such, the problem of a lack of linguistic constraints among edges is inherited by the arc-factored semantic parsing paradigm.

However, MSDAG decoding for semantic parsing suffers from a further problem. In MST decoding, the only constraint is really that the output should have a tree structure; this places a precise restriction on the number of edges in the output (i.e., $n - 1$), unlike for MSDAGs. From our discussion above, we know that the MSDAG problem is closely related to a constrained maximisation of edges. In particular, a candidate solution s to the problem that is not optimal in terms of total weight may, however, be linguistically optimal; adding further edges to s would increase weight, but may be linguistically undesirable.

Consider the tree at the top of Figure 1, for the example *John loves Mary*. In decoding, this tree could be our linguistic optimal, however according to our additive objective function, it is more likely for us to obtain either of the bottom DAGs, which is clearly not what is wanted in semantic parsing.

4 Related Research and an Approximation Guarantee

The only algorithm presented to date for MSDAG is an approximation algorithm proposed by McDonald and Pereira (2006), given without any approximation guarantee. The algorithm first constructs an MST of the weighted digraph, and then greedily attempts to add remaining edges to the MST in order of descending weight, so long as no cycle is introduced. Only part of this algorithm is greedy, so we will refer to it as **`semi-greedy-MSDAG`**. Given the fact that MSDAG is **APX**-hard (Schluter, submitted), the following approximation guarantee is not surprising.

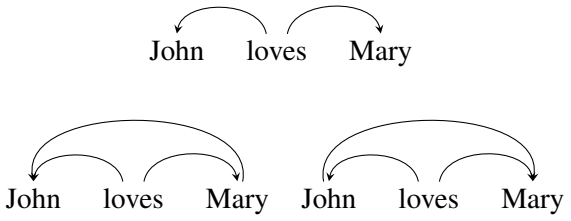

Figure 1: Possible spanning DAGs for *John loves Mary*.

Theorem 2. ***`semi-greedy-MSDAG`*** *is an* $O(\frac{1}{n})$ *factor approximation algorithm for MSDAG.*

Proof. We separate the proof into two parts. In Part 1, we first consider an easy worst case scenario for an upper bound on the error for **`semi-greedy-MSDAG`**, without any consideration for whether such a graph actually exists. Following this in Part 2, we construct a family of graphs to show that this bound is tight (i.e., that the algorithm exhibits worst imaginable behaviour for this family of graphs).

Part 1. For G a digraph, let D be the output of **`semi-greedy-MSDAG`** on G, and D^* be an MSDAG for G. The worst case is (bounded by the case) where the algorithm finds an MST T^* for G but then cannot introduce any extra edges to obtain a spanning DAG of higher weight, because the addition of any extra edges would induce a cycle. For G's nodes, we suppose that $|V(G)| > 3$. For edges, we suppose that all the edges in T^* have equally the largest weight, say w_{max}, of any edge in $E(G)$, and that all other edges in $E(G)$ have weight $O(w_{max})$. We can do this, because it gives an advantage to T^*.

We suppose also that the underlying undirected graph of G is complete and that the true MSDAG for G is $D^* := (V(G), E(G) - E(T^*))$.

This clearly must be the worst imaginable case: that T^* shares no edges with D^*, but that D^* con-

tains every other possible edge in the graph, with the weight of every edge in D^* being at most the weight of the maximum weighted edge of those of T^* (remember we are trying to favour T^*). No other imaginable case could introduce more edges to D^* without inducing a cycle. So, for all G,

$$w(D^*) = O\left(\frac{(n-1)^2 \cdot w_{max}}{2}\right) = O(n^2 \cdot w_{max}),$$

and we had that $w(T^*) = w(D) = O(n \cdot w_{max})$. So at very worst, **semi-greedy-MSDAG** finds a spanning DAG D of weight within $O(\frac{1}{n})$ of the optimal.

Part 2. We now show that this bound is tight. We construct a family of graphs $G_n = (V_n, E_n)$ as follows. $V_n := \{v_0, v_1, v_2, \ldots, v_n\}$, with $n > 3$, and we suppose that n is even. Let $c \in \mathbb{R}$ be some constant such that $c > 3$. We place the following edges in E_n:

(E1) (v_i, v_{i+1}) of weight c for all $i \in \{0, \ldots, n-1\}$ into E_n, creating a path from v_0 to v_n of length n where every edge has weight c, and

(E2) (v_i, v_j) of weight $c-1$ for all $j \in \{2, i-1\}$, $i \in \{2, \ldots, n\}$.

So, in addition to the path defined by the edges in (E1), $G_n - \{v_0\}$ contains a weighted tournament on $n-1$ nodes, such that if $j < i$, then there is an edge from i to j.

Let us denote the MST of G_n by T_n^* and the maximal spanning tree obtainable by **semi-greedy-MSDAG**, by D_n. We will show that the (unique) MSDAG of G_n is the graph D_n^* that we construct below.

It is easy to see that the MST T_n^* of G_n consists of the edges in (E1), and that no further edges can be added to T_n^* without creating a cycle. So, $D_n = T_n^*$.

On the other hand, we claim that there is a unique D_n^* consisting of:

1. the edge (v_0, v_1),
2. the edges (v_{2i-1}, v_{2i}) for all $i \in \{1, \ldots, n/2\}$ into $E(D_n^*)$, that is every second edge in the path described in (E1), and
3. all the edges from (E2) except for (v_{2i}, v_{2i-1}) for all $i \in \{1, \ldots, n/2\}$.

We can easily see that D_n^* is at least maximal. The only edges not in D_n^* are ones that are parallel to other edges. So, introducing any other edge from (E2) would mean removing an edge from (E1), which would decrease D_n^*'s overall weight. Moreover, notice that introducing any other edge from (E1), say (v_{k-1}, v_k) would require removing two edges (from (E2)), either (v_k, v_{k-1}) and (v_{k+1}, v_{k-1}) or (v_k, v_{k-1}) and (v_k, v_{k+1}), to avoid cycles in D_n^*, but this also decreases overall weight. We extend these two simple facts in the remainder of the proof, showing that D^*, in addition to being maximal, is also a global maximum.

We prove the result by induction on n (with n even), that D_n^* is the MSDAG for G_n. We take $n = 4$ as our base case.

For G_4 (see Figure 2), $E(G_4) - E(D_4^*)$ contains only three edges: the edge (v_2, v_3) of weight c and the edges (v_4, v_3) and (v_2, v_1) of weight $(c-1)$. Following the same principles above, adding the edge (v_2, v_1) would entail removing an edge of higher weight; the same is true of adding the edge (v_4, v_3). No further edges in either case could be added to make up this difference and achieve greater weight than D_4^*. So the only option is to add the edge (v_2, v_3). However, this would entail removing either the two edges (v_3, v_2) and (v_4, v_2) or (v_3, v_2) and (v_3, v_4) from D_4^*, both of which actions results in lower overall weight.

Now suppose D_{n-2}^* is optimal (for $n \geq 6$, n even). We show that this implies D_n^* is optimal (with n even). Consider the two subgraphs G_{n-2} and H of G_n induced by the subsets of $V(G_n)$, $V(G_{n-2}) = \{v_0, \ldots, v_{n-2}\}$ and $V(H) := \{v_{n-1}, v_n\}$ respectively (so $H = (V(H), \{(v_{n-1}, v_n), (v_{n-1}, v_n)\})$). We are assuming that the MSDAG of G_{n-2} is D_{n-2}^*. Moreover, the MSDAG of H is a single edge, $D_H := (V(H), \{(v_{n-1}, v_n)\})$.

D_n^* includes the MSDAGs (D_{n-2}^* and D_H) of these two digraphs, so for these parts of D_n^*, we have reached an upper bound for optimality. Now we consider the optimal way to connect D_{n-2}^* and D_H to create D_n^*.

Let C denote the set of edges in G_n which connect D_H to G_{n-2} and vice versa. $C = \{(v_{n-2}, v_{n-1})\} \cup \{(u_n, u_i) \mid 1 \leq i < n\} \cup \{(u_{n-1}, u_i) \mid 1 \leq i < n-1\}$. Note that the only edge from C not included in D_n^* is $e_C := (v_{n-2}, v_{n-1})$. By the discussion above, we know that including e_C would mean excluding two other edges from C of weight at least $(c-1)$, which cannot be optimal. Therefore D_n^* must be optimal.

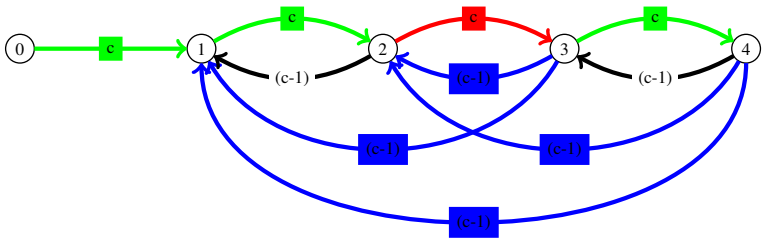

Figure 2: G_4, with D_4^* in blue and green and T_4^* in red and green.

So we have constructed a family of graphs G_n where $w(D_n) = w(T_n^*) = nc$ and

$$\begin{aligned} w(D_n^*) &= \left(\sum_{i=0}^{n-1} i(c-1) + (n \cdot c)\right) - (\frac{n}{2} \cdot c) \\ &= \frac{n(n-1)}{2}(c-1) - \frac{n}{2} \cdot c. \end{aligned}$$

This completes the proof that **semi-greedy-MSDAG** is an $O\left(\frac{nc}{\frac{n(n-1)}{2}(c-1)-\frac{n}{2}\cdot c}\right) = O(\frac{1}{n})$ factor approximation algorithm for MSDAG. □

Now let us consider the version of the statement of the MSDAG problem that, rather than maximising the weight of a spanning DAG D^* of a weighted digraph G, looks to minimise the weight of the set C^* of edges that must be removed from G in order for $G - C^*$ to be an MSDAG for G. Clearly these problems are identical. We refer to the minimisation version of the statement as $MSDAG^C$, and to C^* as the complement (in G) of the MSDAG $D^* := G - C^*$. Also, let **semi-greedy-MSDAG**^{C} be the same algorithm as **semi-greedy-MSDAG** except that it outputs C^* rather than D^*.

Using the same graphs and proof structure as in the proof of Theorem 2, the following theorem can be shown.

Theorem 3. ***semi-greedy-MSDAG***^{C} *is an $O(n)$ factor approximation algorithm for MSDAG*^{C}.

5 Conclusions and Open Questions

This paper provides some philosophical and mathematical foundations for the MSDAG problem as decoding in semantic parsing. We have put forward the view that the objective in semantic parsing is not in fact to find the MSDAG, however it remains open as to whether this mismatch can be tolerated, given empirical evidence of MSDAG decoding's utility in semantic parsing. We have also pointed to an explicit proof of the **APX**-hardness (that of (Schluter, submitted)) of MSDAG and given an approximation guarantee of the only published approximation algorithm for this problem.

In particular, Schluter (submitted) provides an approximation preserving reduction from MSDAG^C to IDMC. Moreover, the best known approximation ratio for IDMC is $O(n^{\frac{11}{23}})$ (Agarwal et al., 2007), which yields a better (in terms of worst case error) approximation algorithm for MSDAG^C. An interesting open problem would compare these two decoding approximation algorithms empirically for semantic parsing decoding and in terms of expected performance (or error) both in general as well as specifically for semantic parsing decoding.

References

Amit Agarwal, Noga Alon, and Moses Charikar. 2007. Improved approximation for directed cut problems. In *Proceedings of STOC*, San Diego, CA.

D. Heckerman, D. Geiger, and D. M. Chickering. 1995. Learning bayesian networks: The combination of knowledge and statistical data. Technical report, Microsoft Research. MSR-TR-94-09.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proc. of EACL*, pages 81–88.

R. McDonald, F. Pereira, K. Ribarov, and J. Haji. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proc. of HLT-EMNLP*, pages 523–530, Vancouver, BC, Canada.

Kenji Sagae and Jun'ichi Tsujii. 2008. Shift-reduce dependency dag parsing. In *22nd International Conference on Computational Linguistics (Coling 2008)*, Manchester, UK.

Natalie Schluter. submitted. On the complexity of finding a maximum spanning dag and other dag parsing related restrictions.

Ivan Titov, James Henderson, Paola Merlo, and Gabrielle Musillo. 2009. Online graph planarization for synchronous parsing of semantic and syntactic dependencies. In *Proceedings of IJCAI 2009*, pages 1562–1567.