Abstract Meaning Representation
– a survey –

Melanie Tosik

University of Potsdam, Department Linguistics
Seminar: Dependency Parsing (Dr. Željko Agić)
July 9, 2015

Abstract

The aim of this paper is to present a detailed overview of Abstract Meaning Representation (AMR), a semantic representation language introduced by Banarescu et al. (2013). In order to assess the various aspects of AMR, we first outline the nature and design principles behind it. This includes an in-depth description of the AMR format and its content, as well as a discussion of the limitations of AMR and primary remarks on annotation and evaluation. The second part of the paper surveys the current state-of-the-art for computation with AMRs, providing insights in AMR parsing, and current and future applications.

1 Introduction

Who did what to whom? A question that every human can easily answer in a given context, but is very difficult for computers to analyze directly. For more than two decades, natural language processing heavily relied on syntactic treebanks in order to enable computers to derive meaning from language. Following the release of the first large-scale treebank, the Penn Treebank (Marcus et al., 1993), many more syntactic treebanks have been developed for a wide variety of languages, and used to build state-of-the-art natural language processing systems such as part-of-speech (POS) taggers, parsers, semantic analyzers, and machine translation (MT) systems.

Moving from the analysis of grammatical structure to sentence semantics, however, we find that statistical parsers are ill-suited for producing meaning representations. In semantic analysis, we often encounter complex structures that are impossible to capture within the limits of tree structures. For example, semantic nodes will often be the argument of more than one predicate, and it will often be useful to exclude semantically vacuous words (like particles or complementizers), i.e. leave nodes unattached which do not add further meaning to the resulting representation.

To overcome this fundamental limitation and enable a more direct semantic analysis of whole sentences, emerging research is shifting towards parsing with graph-structured representations. Just as syntactic treebanks were of significant importance to improve upon syntactic parsers, current research on semantic parsing relies on sembanks: sets of English sentences paired with their corresponding semantic representations.

Typically, semantic parsing entails domain dependence: example application domains include ATIS, the *Air Travel Information Service* (Price, 1990), the *Robocup Coach Language* CLang (Chen et al., 2003), or *GeoQuery: A Database Query Application* (Wong and Mooney, 2006). However, there is a growing need for larger, broad-coverage sembanks. To this end, several projects have been initiated, including the Groningen Meaning Bank (GMB) (Basile et al., 2012), UCCA (Abend and Rappoport, 2013), the Semantic Treebank (ST) (Butler and Yoshimoto, 2012), the Prague Dependency Bank (Böhmová et al., 2001), and UNL (Uchida et al., 1996; Martins, 2012).

Taking another step forward, Banarescu et al. (2013) recently started annotating the logical meaning of sentences in Abstract Meaning Representation (AMR) – single rooted, directed graphs, that incorporate semantic roles, coreference, questions, modality, negation, and further linguistic phenomena. By providing a substantial corpus and a correctable, logical semantic input format, the creators of AMR are hoping to encourage significant advances in statistical natural language understanding (NLU), natural language generation (NLG), and statistical machine translation (SMT), inter alia.

2 AMR formalism

The Abstract Meaning Representation (AMR) language is presented in Banarescu et al. (2013), and described in more detail in the AMR annotation guidelines[1]. In a nutshell, AMR graphs are rooted, labeled, directed, acyclic graphs (DAGs), comprising whole sentences. They are intended to abstract away from syntactic representations, in the sense that sentences which are similar in meaning should be assigned the same AMR, even if they are not identically worded. By nature, the AMR language is biased towards English – it is not meant to function as an international auxiliary language.

2.1 Format

AMRs can be written in three different notations. Traditionally, AMRs can be represented as conjunctions of logical triples. For human reading and writing, the PENMAN notation (Matthiessen and Bateman, 1991) is adapted. For computer processing, a conventional graph notation is used. A basic example[2] of all three notations is illustrated in Figure 1.

2.2 Content

As illustrated in the example, AMR introduces variables (graph nodes) for entities, events, properties, and states. Each node in the graph represents a semantic concept. These concepts can either be English words (*prince*), PropBank framesets (*say-01*) (Palmer et al., 2005), or special keywords. In PENMAN notation, *(p / prince)* refers to an instance *p* of the concept *prince*. Edge labels denote the relations that hold between entities. For example, *(p / prince :mod (l / little))* denotes the modality of our prince being little. These tokens, including *:mod* or *:arg0* (typically used for agents), are referred to as AMR role tokens. It is worth noting that entities with multiple semantic roles are represented with only a single node in the graph. In such cases, variables are re-used and re-entrancies are annotated in the graph.

[1] `http://amr.isi.edu/language.html`

[2] Taken from public Release 1.4 of the AMR Bank (1,562 sentences from *The Little Prince*; November 14, 2014; `http://amr.isi.edu/download/amr-bank-v1.4.txt`)

English sentence: "I do not understand", said the little prince.

Logic format:

∃ s, p, l, u, -:
instance(s, say-01) ∧ instance(p, prince) ∧ instance(l, little) ∧ instance(u, understand) ∧ instance(-, -) ∧ arg0(s, p) ∧ arg1(s, u) ∧ (u, p) ∧ mod(p, l) ∧ polarity(u, -)

AMR format (PENMAN notation):

```
(s / say-01
  :arg0 (p / prince
           :mod (l / little))
  :arg1 (u / understand-01
           :arg0 p
           :polarity -))
```

Graph format:

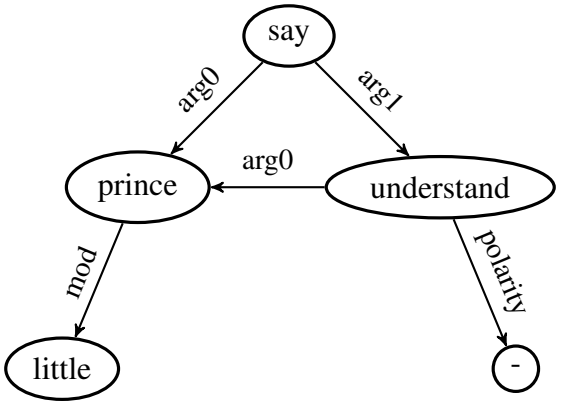


Figure 1: Example sentence and equivalent meaning representations

In sum, AMR uses approximately 100 relations. There are frame arguments (*:arg0*, *:arg1*, etc), general semantic relations (*:age*, *:destination*, *:location*, *:name*, ...), relations for quantities (*:unit*, *:scale*, *:quant*), relations for date-entities (*:day*, *:month*, *:time*, ...), and relations for lists (*:op1*, *:op2*, etc). In addition, AMR includes the inverses of all its relations (e.g. *:arg-0-of* or *:location-of*) and allows for reification, i.e. the modification of relations themselves. Banarescu et al. (2013) give a comprehensive list of example AMRs for frame arguments, general semantic relations, co-reference, inverse relations, modals and negation, questions, verbs, nouns, adjectives, prepositions, named entities, copula, and reification. All further phenomena are additionally covered in the AMR guidelines.

2.3 Limitations

In order to ensure a fast annotation process and obtain concise semantic representations, the AMR formalism abstracts away from morpho-syntactic idiosyncrasies such as word category or word order, and does not account for tense or number. Most function words (articles and prepositions) are omitted. There are no universal quantifiers. Since AMR heavily relies on Propbank framesets, it is subject to the corresponding Propbank constraints.

2.4 Annotation

The AMR Bank corpus is manually constructed by human annotators. It is comprised of several thousand English/AMR pairs, a subset of which is freely available for download[3]. To support annotators at different levels of experience, a web-based power editor has been developed[4]. The AMR Editor includes a manual, quick references, various examples with explanations, a search function, as well as appropriate suggestions, providing a wide range of support for annotators to choose the proper concepts, roles, and overall structure.

2.4.1 Evaluation

To enable the evaluation of whole-sentence semantic analysis, Cai and Knight (2013) introduce *Smatch* (for semantic match), a metric that calculates the degree of overlap between two semantic feature structures. With regard to AMR, it can be used to assess both semantic annotation agreement rates, as well as automatic parsing accuracy.

As mentioned beforehand, AMRs can be represented as conjunctions of logical triples, taking one of these forms: *relation(variable, concept)*, or *relation(variable1, variable2)*. Take for example the following two sentences and their corresponding logical triples:

(1) The girl is reading the book.

$$\Updownarrow$$

instance(a0, read) $\wedge$ instance(a1, girl) $\wedge$ instance(a2, book) $\wedge$ arg0(a0, a1) $\wedge$ arg1(a0, a2)

(2) The boy is holding the book.

$$\Updownarrow$$

instance(b0, hold) $\wedge$ instance(b1, boy) $\wedge$ instance(b2, book) $\wedge$ arg0(b0, b1) $\wedge$ arg1(b0, b2)

The *Smatch* score now computes the maximum match number of triples among all possible variable mappings, and gets precision, recall, and F1 score. To obtain the variable mapping which yields the highest F1 score, *Smatch* executes a brief search. For our example, we would obtain the following variable mapping:

a0(read)–b0(hold) a1(girl)–b1(boy) a2(book)–b2(book)

Thus, there are **matched** and unmatched triples:

instance(a0, read) $\wedge$ instance(a1, girl) $\wedge$ **instance(a2, book)** $\wedge$ **arg0(a0, a1)** $\wedge$ **arg1(a0, a2)**
instance(b0, hold) $\wedge$ instance(b1, boy) $\wedge$ **instance(b2, book)** $\wedge$ **arg0(b0, b1)** $\wedge$ **arg1(b0, b2)**

We can compute:

$$F1(\text{Precision}, \text{Recall}) = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \times \frac{3}{5} \times \frac{3}{5}}{\frac{3}{5} + \frac{3}{5}} = 0.6$$

[3] http://amr.isi.edu/download.html
[4] http://amr.isi.edu/editor.html

3 Algorithms and applications

The second part of this paper will survey the state-of-the-art for computation with AMRs. We start by introducing the first approach to parsing English text into AMR graphs, before moving on to cross-lingual applications of AMR, with a focus on AMR-based machine translation (MT).

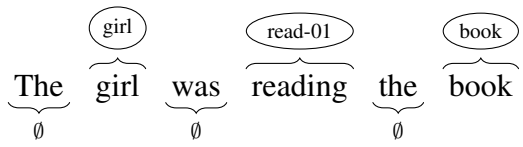
3.1 Parsing

Flanigan et al. (2014) introduce the first approach to parse sentences into AMR, which requires algorithms for alignment, structured prediction, and statistical learning. Their method is based on a novel algorithm for finding a maximum spanning, connected subgraph, and approximates a constrained optimization problem using Lagrangian relaxation. The statistical parameters of the JAMR system[5] are learned from the annotated English/AMR pairs in the AMR Bank corpus. The system is evaluated using the *Smatch* score.

In order to automatically parse English into AMR, JAMR employs a two-part algorithm. First, it identifies the key concepts using a semi-Markov model. Second, it identifies the relations between the concepts by searching for the maximum spanning, connected subgraph (MSCG). The MSCG algorithm returns the connected subgraph with the maximum sum of edge weights among all connected subgraph of the input graph. It is thus similar to the widely used maximum spanning tree (MST) tree described in Mcdonald et al. (2005). Lagrangian relaxation (LR) (Geoffrion, 1974; Fisher, 2004) is applied in order to preserve the linguistically inspired constraints imposed by the AMR language.

Concept identification

In the given approach, an AMR parse is represented as a graph $G = \langle V, E \rangle$. To identify concepts, an input sentence **w** is first segmented into contiguous spans. Afterwards, each span is mapped to a graph fragment corresponding to a concept from a concept set F (or $\emptyset$, if no concept is invoked by a given span). For example:



Like in the example, graph fragments are often single, labeled concept nodes, but can comprise multiple nodes and edges if applicable. Given a sequence of spans **b** and a sequence of concept frames **c**, both of arbitrary length *k*, a labeling score can be computed as follows:

$$score(\mathbf{b}, \mathbf{c}; \boldsymbol{\theta}) = \sum_{i=1}^{k} \boldsymbol{\theta}^T \boldsymbol{f}(\boldsymbol{w}_{b_{i-1}:b_i}, b_{i-1}, b_i, c_i)$$

Each phrase $\boldsymbol{w}_{b_{i-1}:b_i}$ is assigned a concept graph fragment $c_i \in clex(\boldsymbol{w}_{b_{i-1}:b_i}) \cup \{\emptyset\}$. $\boldsymbol{f}$ is a feature vector. Features are the probability of a fragment given words, the length of the matching span (number of tokens), NER (binary; 1 if entity, 0 otherwise), and bias (binary; 1 if concept graph from F, 0 otherwise).

[5] `http://github.com/jflanigan/jamr`

Relation identification

In the second step of the algorithm, relations between concepts are identified by adding edges among the concept subgraph fragments identified in the first step. The task is to construct a subgraph $G = \langle V_G, E_G \rangle$ that is preserving, simple, connected, and deterministic. All four constraints follow from the definition of AMR. A score to decompose by edges can be defined with a linear parametrization:

$$score(E_G; \boldsymbol{\psi}) = \sum_{e \in E_G} \boldsymbol{\psi}^T \boldsymbol{g}(e)$$

For an overview of the feature set and further technical details, please refer to the original paper by Flanigan et al. (2014). In conclusion, there are a number of subsequent steps ensuring that the resulting graph G satisfied the imposed constraints: the initialization step ensures the preserving constraint is satisfied, the pre-processing step ensures the graph is simple, and the core algorithm ensures that the graph is connected.

Alignments

The semantic parser is trained on the set of annotated English/AMR examples contained in the AMR Bank corpus. However, these annotations do not include alignment links between the words in the input sentences and the corresponding concepts in the AMR graphs – for each English sentence, one AMR graph is given, but without any information as to how one representation was derived from the other. But, in order to train a semantic parser for AMR, we need to know which spans of words in the input sentence invoke which concepts in the corresponding graph, i.e. we need alignment links between each English token and its AMR representation.

To solve the alignment task, Flanigan et al. (2014) develop a heuristic aligner that uses a set of rules to greedily align concepts to spans. For example, they define rules to account for named entities, date entities, minus polarity tokens, quantities, or persons. Using the full set of rules, the aligner achieves 92% precision, 89% recall, and 90% F1 score on a set of 200 hand-aligned sentences from the training data. However, their method is restricted to the pre-defined set of alignment rules, and thus will not automatically improve as more parallel AMR/English data becomes available.

Overcoming this limitation, Pourdamghani et al. (2014) recently published an alternative method for aligning English/AMR pairs at token level. Preceding the alignment phase, a series of preprocessing steps is applied. Most importantly, this includes linearizing the AMR graphs into strings and stemming all tokens into their first four letters.

The training phase is based on the IBM translation models and the definition of a generative model from AMR graphs to strings. Previously hidden alignment links are subsequently uncovered using the Expectation-Maximization (EM) algorithm. In order to allow for symmetrical parameter learning during EM training, the objective function of the IBM models is modified accordingly. Pourdamghani et al. (2014) report 86.5% and 83.1% alignment F1 score on development and test set, respectively.

3.2 Cross-lingual applications

This alignment problem is closely related to the problem of statistical machine translation (SMT): the translation of text from one language to another, based on parameterized statistical models. Machine translation was one of the first applications envisioned for computers (Weaver, 1955), and remains a commercially and academically interesting challenge in natural language processing (NLP). The first MT system was introduced by IBM in 1954, and a basic word-for-word translation system. Since then, various approaches to machine translation have been implemented, ranging from word-to-word translation, syntactic transfer, interlingual approaches, controlled language, and example-based translation to statistical machine translation.

Even though it was already claimed in the early stages of MT that a semantic model is necessary to achieve human-like translation (Weaver, 1955), most research in the field was concerned with phrase-based approaches to MT. Statistical phrase-based MT systems typically rely on parallel corpora to efficiently estimate translation probabilities between two languages. Given sufficient training data, phrase-based systems are then able to automatically resolve some common ambiguities.

However, they operate under the assumption that surface phrases can be translated without incorporating any notion of syntax or semantics. Thus, they are likely to fail when it comes to processing non-local phenomena such as argument re-orderings across languages, deep embeddings, anaphora, or empty categories. The availability of syntactically-annotated, parallel data has been steadily increasing over the past decades, allowing for syntactical information to be directly integrated into data-driven MT systems. With regard to meaning representation, on the other hand, many problems remain unsolved. Consider, for example, the following translation using a state-of-the-art German $\rightarrow$ English SMT system[6]:

Du fehlst mir. $\rightarrow$ *You're missing me.*

While the translation might appear to be correct on the surface, it turns out to be wrong. In the given input sentence, the correct translation would be, *I miss you*. The example illustrates how SMT systems are frequently unable to preserve even the most basic meaning structures when confronted with verbs that realize their arguments differently across two languages.

To tackle this issue, Jones et al. (2012) present the first semantics-based SMT system. Using a graph-structured meaning representation, the system first analyzes the source language into the meaning representations, and subsequently decodes the meaning representations into the target language. Similarly to AMR, the system assumes that meaning representations are directed, acyclic graphs. Since there are no further restrictions with regard to the details of the formalism, it is flexible enough to handle a number of existing meaning representations, e.g. the language of the *GeoQuery* corpus (Wong and Mooney, 2006).

To manipulate the graph structures involved, Jones et al. (2012) use hyper-edge replacement grammars (HRGs) (Drewes et al., 1997). In the original paper, they demonstrate that the proposed systems is in fact able to capture semantic abstractions by analyzing the input text into meaning and then back into text in the target language.

[6]Google Translate (May 25, 2015)

Xue et al. (2014) investigate to which extent AMR could serve as such a transfer layer in MT systems. By abstracting away from morpho-syntactical idiosyncrasies, AMR eliminates several sources of cross-lingual differences, and it would be interesting to know to what extent it was possible to obtain structurally compatible AMRs for a sentence in multiple languages[7].

In a qualitative comparison, Xue et al. (2014) compare AMRs for English to Chinese and Czech, and report three possible scenarios. First, translations of the same sentence are annotated with structurally identical AMRs, i.e. both AMRs are perfectly aligned. Second, two differing AMRs are obtained, but the differences are due to inconsistencies in the annotation process. Theoretically, such mismatches could be reconciled by refining the annotation standards. Third, the resulting AMRs differ because of different lexicalizations in the input and target language. This issue cannot be resolved unless a higher level of abstraction in the AMR language is introduced. Interestingly, Xue et al. (2014) also find that there are more structurally incompatible AMRs for Czech than for Chinese, when compared to the English counterparts. Manual inspection of the annotated Czech data suggests that more than half of the sentences profoundly differ from the English translations.

4 Summary and outlook

The Abstract Meaning Representation (AMR) formalism is rapidly emerging as an important practical form of structured sentence semantics. Due to the availability of large-scale annotated corpora, it has potential as a convergence point for NLP research. AMRs are rooted, labeled graphs, capturing meaning on sentence level while abstracting away from morpho-syntactical idiosyncrasies. Nodes in the graph denote semantic concepts, while edge labels denote the relations that hold between concepts.

The AMB Bank corpus is manually constructed by human annotators and already comprises several thousand sentences. Traditionally, AMRs can also be represented as conjunction of logical triples. Based on this logical format, Cai and Knight (2013) define the *Smatch* score. It can be used to evaluate inter-annotator agreement (IAA), as well as AMR parsing accuracy.

The first approach to automatic AMR parsing has been introduced by Flanigan et al. (2014). Their algorithm first identifies the concepts using a semi-Markov model, and then the relations between these by searching for a maximum spanning connected subgraph (MSCG). In order to obtain English/AMR alignments in the pre-processing step, Flanigan et al. (2014) present a heuristic aligner based on a pre-defined set of rules. Pourdamghani et al. (2014) propose a statistical alignment model, which – unlike the heuristic approach – will automatically improve as more data becomes available.

While the potential applications of AMR are manifold, current research is exploring the use of AMR in statistical machine translation (SMT). Jones et al. (2012) present the first semantics-based SMT, using a graph-structured meaning representation as transfer layer between the source and the target language. Xue et al. (2014) subsequently investigate the use of AMR as mediator between multiple languages, suggesting that a higher level of abstraction might be necessary to account for lexicalization differences between languages.

[7] Structurally compatible AMRs refers to AMRs with all concepts and relations aligned.

From here, there are many directions future AMR work might take. Most importantly, further enhancing the AMR Bank corpus will lead to shared tasks on natural language understanding (NLU) and generation (NLG), thus advancing the field and driving new interests in graph-based semantic parsing. Even though prototypes for AMR-based SMT systems already exist, future improvements are to be expected at this end as well. Finally, the AMR language is frequently subject to change – ultimately, it might include more relations, entity normalization, quantification, or temporal relations. Additionally, a comprehensive list of more abstract frames is imaginable.

References

Abend, O. and A. Rappoport (2013, March). UCCA: A Semantics-based Grammatical Annotation Scheme. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Long Papers*, pp. 112. Association for Computational Linguistics.

Banarescu, L., C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider (2013). Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, Sofia, Bulgaria, pp. 178–186. Association for Computational Linguistics.

Basile, V., J. Bos, K. Evang, and N. Venhuizen (2012). Developing a large semantically annotated corpus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*, Istanbul, Turkey, pp. 3196–3200.

Böhmová, A., J. Hajič, E. Hajičová, and B. Hladká (2001). The Prague Dependency Treebank: Three-Level Annotation Scenario. In A. Abeillé (Ed.), *Treebanks: Building and Using Syntactically Annotated Corpora*. Kluwer Academic Publishers.

Butler, A. and K. Yoshimoto (2012). Banking Meaning Representations from Treebanks. *Linguistic Issues in Language Technology 7*(1).

Cai, S. and K. Knight (2013). Smatch: an Evaluation Metric for Semantic Feature Structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers*, pp. 748–752.

Chen, M., K. Dorer, E. Foroughi, F. Heintz, Z. Huang, S. Kapetanakis, K. Kostiadis, J. Kummeneje, J. Murray, I. Noda, O. Obst, P. Riley, T. Steffens, Y. Wang, and X. Yin (2003, Februar). *Users Manual: RoboCup Soccer Server — for Soccer Server Version 7.07 and Later*. The RoboCup Federation.

Drewes, F., H.-J. Kreowski, and A. Habel (1997). Handbook of graph grammars and computing by graph transformation. Chapter Hyperedge Replacement Graph Grammars, pp. 95–162. River Edge, NJ, USA: World Scientific Publishing Co., Inc.

Fisher, M. L. (2004). The Lagrangian Relaxation Method for Solving Integer Programming Problems. *Manage. Sci. 50*(12 Supplement), 1861–1871.

Flanigan, J., S. Thomson, J. Carbonell, C. Dyer, and A. N. Smith (2014). A Discriminative Graph–Based Parser for the Abstract Meaning Representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1426–1436. Association for Computational Linguistics.

Geoffrion, A. (1974). Lagrangean relaxation for integer programming. In *Approaches to Integer Programming*, Volume 2 of *Mathematical Programming Studies*, pp. 82–114. Springer Berlin Heidelberg.

Jones, B., J. Andreas, D. Bauer, K. M. Hermann, and K. Knight (2012, December). Semantics-Based Machine Translation with Hyperedge Replacement Grammars. In *Proceedings of COLING 2012*, pp. 1359–1376.

Marcus, M. P., M. A. Marcinkiewicz, and B. Santorini (1993, June). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics 19*(2), 313–330.

Martins, R. (2012, may). Le Petit Prince in UNL. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.

Matthiessen, C. and J. A. Bateman (1991). *Text generation and systemic-functional linguistics: experiences from English and Japanese*. Pinter Publishers.

Mcdonald, R., F. Pereira, K. Ribarov, and J. Hajič (2005). Non-projective dependency parsing using spanning tree algorithms. In *In Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pp. 523–530.

Palmer, M., D. Gildea, and P. Kingsbury (2005, March). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics 31*(1), 71–106.

Pourdamghani, N., Y. Gao, U. Hermjakob, and K. Knight (2014). Aligning English Strings with Abstract Meaning Representation Graphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, pp. 425–429. Association for Computational Linguistics.

Price, P. J. (1990). Evaluation of Spoken Language Systems: The ATIS Domain. In *Proceedings of the Speech and Natural Language Workshop*, Hidden Valley, PA, pp. 91–95.

Uchida, H., M. Zhu, and T. D. Senta (1996). UNL: Universal Networking Language – an electronic language for communication, understanding and collaboration. Technical report, IAS/UNU Tokyo.

Weaver, W. (1949/1955). Translation. In W. N. Locke and A. D. Boothe (Eds.), *Machine Translation of Languages*, Cambridge, MA, pp. 15–23. MIT Press. Reprinted from a memorandum written by Weaver in 1949.

Wong, Y. W. and R. J. Mooney (2006). Learning for Semantic Parsing with Statistical Machine Translation. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, Stroudsburg, PA, USA, pp. 439–446. Association for Computational Linguistics.

Xue, N., O. Bojar, J. Hajič, M. Palmer, Z. Urešová, and X. Zhang (2014). Not an Interlingua, But Close: Comparison of English AMRs to Chinese and Czech. In N. Calzolari, K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis (Eds.), *LREC*, pp. 1765–1772. European Language Resources Association (ELRA).