The Complexity of Phrase Alignment Problems

John DeNero and Dan Klein
Computer Science Division, EECS Department
University of California at Berkeley
{denero, klein}@cs.berkeley.edu

Abstract

Many phrase alignment models operate over the combinatorial space of *bijective phrase alignments*. We prove that finding an optimal alignment in this space is NP-hard, while computing alignment expectations is #P-hard. On the other hand, we show that the problem of finding an optimal alignment can be cast as an integer linear program, which provides a simple, declarative approach to Viterbi inference for phrase alignment models that is empirically quite efficient.

1 Introduction

Learning in phrase alignment models generally requires computing either Viterbi phrase alignments or expectations of alignment links. For some restricted combinatorial spaces of alignments—those that arise in ITG-based phrase models (Cherry and Lin, 2007) or local distortion models (Zens et al., 2004)—inference can be accomplished using polynomial time dynamic programs. However, for more permissive models such as Marcu and Wong (2002) and DeNero et al. (2006), which operate over the full space of *bijective phrase alignments* (see below), no polynomial time algorithms for exact inference have been exhibited. Indeed, Marcu and Wong (2002) conjectures that none exist. In this paper, we show that Viterbi inference in this full space is NP-hard, while computing expectations is #P-hard.

On the other hand, we give a compact formulation of Viterbi inference as an integer linear program (ILP). Using this formulation, exact solutions to the Viterbi search problem can be found by highly optimized, general purpose ILP solvers. While ILP is of course also NP-hard, we show that, empirically, exact solutions are found very quickly for most problem instances. In an experiment intended to illustrate the practicality of the ILP approach, we show speed and search accuracy results for aligning phrases under a standard phrase translation model.

2 Phrase Alignment Problems

Rather than focus on a particular model, we describe four problems that arise in training phrase alignment models.

2.1 Weighted Sentence Pairs

A *sentence pair* consists of two word sequences, $\mathbf{e}$ and $\mathbf{f}$. A set of phrases $\{e_{ij}\}$ contains all spans e_{ij} from between-word positions i to j of $\mathbf{e}$. A *link* is an aligned pair of phrases, denoted (e_{ij}, f_{kl}).[1]

Let a *weighted sentence pair* additionally include a real-valued function $\phi : \{e_{ij}\} \times \{f_{kl}\} \rightarrow \mathbb{R}$, which scores links. $\phi(e_{ij}, f_{kl})$ can be sentence-specific, for example encoding the product of a translation model and a distortion model for (e_{ij}, f_{kl}). We impose no additional restrictions on ϕ for our analysis.

2.2 Bijective Phrase Alignments

An alignment is a set of links. Given a weighted sentence pair, we will consider the space of bijective phrase alignments $\mathcal{A}$: those $\mathbf{a} \subset \{e_{ij}\} \times \{f_{kl}\}$ that use each word token in exactly one link. We first define the notion of a partition: $\sqcup_i S_i = T$ means S_i are *pairwise disjoint* and cover T. Then, we can formally define the set of bijective phrase alignments:

$$\mathcal{A} = \left\{ \mathbf{a} : \bigsqcup_{(e_{ij}, f_{kl}) \in \mathbf{a}} e_{ij} = \mathbf{e} \, ; \bigsqcup_{(e_{ij}, f_{kl}) \in \mathbf{a}} f_{kl} = \mathbf{f} \right\}$$

[1] As in parsing, the position between each word is assigned an index, where 0 is to the left of the first word. In this paper, we assume all phrases have length at least one: $j > i$ and $l > k$.

Proceedings of ACL-08: HLT, Short Papers (Companion Volume), pages 25–28,
Columbus, Ohio, USA, June 2008.

Both the conditional model of DeNero et al. (2006) and the joint model of Marcu and Wong (2002) operate in $\mathcal{A}$, as does the phrase-based decoding framework of Koehn et al. (2003).

2.3 Problem Definitions

For a weighted sentence pair $(\mathbf{e}, \mathbf{f}, \phi)$, let the score of an alignment be the product of its link scores:

$$\phi(\mathbf{a}) = \prod_{(e_{ij}, f_{kl}) \in \mathbf{a}} \phi(e_{ij}, f_{kl}).$$

Four related problems involving scored alignments arise when training phrase alignment models.

OPTIMIZATION, $\mathcal{O}$: Given $(\mathbf{e}, \mathbf{f}, \phi)$, find the highest scoring alignment $\mathbf{a}$.

DECISION, $\mathcal{D}$: Given $(\mathbf{e}, \mathbf{f}, \phi)$, decide if there is an alignment $\mathbf{a}$ with $\phi(\mathbf{a}) \geq 1$.

$\mathcal{O}$ arises in the popular Viterbi approximation to EM (Hard EM) that assumes probability mass is concentrated at the mode of the posterior distribution over alignments. $\mathcal{D}$ is the corresponding decision problem for $\mathcal{O}$, useful in analysis.

EXPECTATION, $\mathcal{E}$: Given a weighted sentence pair $(\mathbf{e}, \mathbf{f}, \phi)$ and indices i, j, k, l, compute $\sum_{\mathbf{a}} \phi(\mathbf{a})$ over all $\mathbf{a} \in \mathcal{A}$ such that $(e_{ij}, f_{kl}) \in \mathbf{a}$.

SUM, $\mathcal{S}$: Given $(\mathbf{e}, \mathbf{f}, \phi)$, compute $\sum_{\mathbf{a} \in \mathcal{A}} \phi(\mathbf{a})$.

$\mathcal{E}$ arises in computing sufficient statistics for re-estimating phrase translation probabilities (E-step) when training models. The existence of a polynomial time algorithm for $\mathcal{E}$ implies a polynomial time algorithm for $\mathcal{S}$, because $\mathcal{A} = \bigcup_{j=1}^{|\mathbf{e}|} \bigcup_{k=0}^{|\mathbf{f}|-1} \bigcup_{l=k+1}^{|\mathbf{f}|} \{\mathbf{a} : (e_{0j}, f_{kl}) \in \mathbf{a}, \mathbf{a} \in \mathcal{A}\}$.

3 Complexity of Inference in $\mathcal{A}$

For the space $\mathcal{A}$ of bijective alignments, problems $\mathcal{E}$ and $\mathcal{O}$ have long been suspected of being NP-hard, first asserted but not proven in Marcu and Wong (2002). We give a novel proof that $\mathcal{O}$ is NP-hard, showing that $\mathcal{D}$ is NP-complete by reduction from SAT, the boolean satisfiability problem. This result holds despite the fact that the related problem of finding an optimal matching in a weighted bipartite graph (the ASSIGNMENT problem) is polynomial-time solvable using the Hungarian algorithm.

3.1 Reducing Satisfiability to $\mathcal{D}$

A reduction proof of NP-completeness gives a construction by which a known NP-complete problem can be solved via a newly proposed problem. From a SAT instance, we construct a weighted sentence pair for which alignments with positive score correspond exactly to the SAT solutions. Since SAT is NP-complete and our construction requires only polynomial time, we conclude that $\mathcal{D}$ is NP-complete.[2]

SAT: Given vectors of boolean variables $\mathbf{v} = (v)$ and propositional clauses[3] $\mathbf{C} = (C)$, decide whether there exists an assignment to $\mathbf{v}$ that simultaneously satisfies each clause in $\mathbf{C}$.

For a SAT instance $(\mathbf{v}, \mathbf{C})$, we construct $\mathbf{f}$ to contain one word for each clause, and $\mathbf{e}$ to contain several copies of the literals that appear in those clauses. ϕ scores only alignments from clauses to literals that satisfy the clauses. The crux of the construction lies in ensuring that no variable is assigned both *true* and *false*. The details of constructing such a weighted sentence pair $\text{wsp}(\mathbf{v}, \mathbf{C}) = (\mathbf{e}, \mathbf{f}, \phi)$, described below, are also depicted in figure 1.

1. $\mathbf{f}$ contains a word for each C, followed by an assignment word for each variable, $\text{assign}(v)$.
2. $\mathbf{e}$ contains $c(\ell)$ consecutive words for each literal ℓ, where $c(\ell)$ is the number of times that ℓ appears in the clauses.

Then, we set $\phi(\cdot, \cdot) = 0$ everywhere except:

3. For all clauses C and each satisfying literal ℓ, and each one-word phrase e in $\mathbf{e}$ containing ℓ, $\phi(e, f_C) = 1$. f_C is the one-word phrase containing C in $\mathbf{f}$.
4. The $\text{assign}(v)$ words in $\mathbf{f}$ align to longer phrases of literals and serve to consistently assign each variable by using up inconsistent literals. They also align to unused literals to yield a bijection. Let $e^k_{[\ell]}$ be the phrase in $\mathbf{e}$ containing all literals ℓ and k negations of ℓ. $f_{\text{assign}(v)}$ is the one-word phrase for $\text{assign}(v)$. Then, $\phi(e^k_{[\ell]}, f_{\text{assign}(v)}) = 1$ for $\ell \in \{v, \bar{v}\}$ and all applicable k.

[2] Note that D is trivially in NP: given an alignment $\mathbf{a}$, it is easy to determine whether or not $\phi(\mathbf{a}) \geq 1$.

[3] A clause is a disjunction of literals. A literal is a bare variable v_n or its negation $\bar{v}_n$. For instance, $v_2 \vee \bar{v}_7 \vee \bar{v}_9$ is a clause.

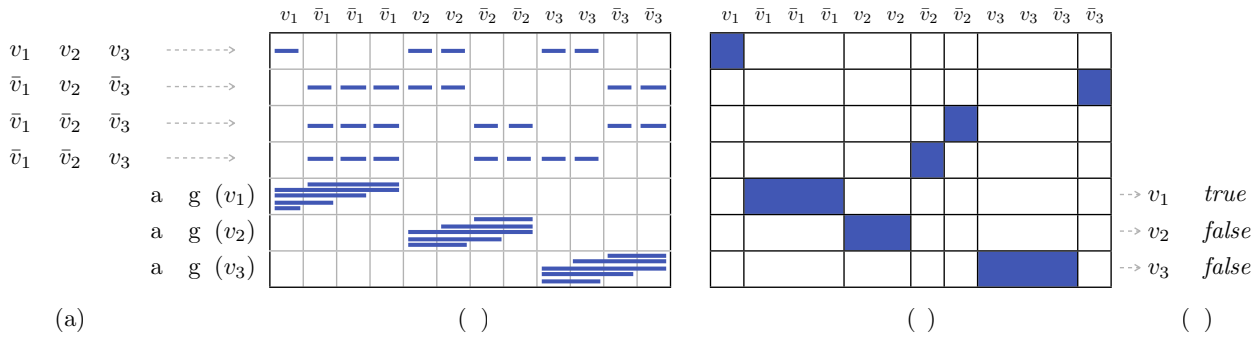

Figure 1: (a) The clauses of an example SAT instance with $\mathbf{v} = (v_1, v_2, v_3)$. (b) The weighted sentence pair wsp$(\mathbf{v}, \mathbf{C})$ constructed from the SAT instance. All links that have $\phi = 1$ are marked with a blue horizontal stripe. Stripes in the last three rows demarcate the alignment options for each assign(v_n), which consume all words for some literal. (c) A bijective alignment with score 1. (d) The corresponding satisfying assignment for the original SAT instance.

Claim 1. *If wsp*$(\mathbf{v}, \mathbf{C})$ *has an alignment* $\mathbf{a}$ *with* $\phi(\mathbf{a}) \geq 1$*, then* $(\mathbf{v}, \mathbf{C})$ *is satisfiable.*

Proof. The score implies that $\mathbf{f}$ aligns using all one-word phrases and $\forall a_i \in \mathbf{a}, \phi(a_i) = 1$. By condition 4, each $f_{\text{assign}(v)}$ aligns to all $\bar{v}$ or all v in $\mathbf{e}$. Then, assign each v to *true* if $f_{\text{assign}(v)}$ aligns to all $\bar{v}$, and *false* otherwise. By condition 3, each C must align to a satisfying literal, while condition 4 assures that all available literals are consistent with this assignment to $\mathbf{v}$, which therefore satisfies $\mathbf{C}$. □

Claim 2. *If* $(\mathbf{v}, \mathbf{C})$ *is satisfiable, then wsp*$(\mathbf{v}, \mathbf{C})$ *has an alignment* $\mathbf{a}$ *with* $\phi(\mathbf{a}) = 1$.

Proof. We construct such an alignment $\mathbf{a}$ from the satisfying assignment $\mathbf{v}$. For each C, we choose a satisfying literal ℓ consistent with the assignment. Align f_C to the first available ℓ token in $\mathbf{e}$ if the corresponding v is *true*, or the last if v is *false*. Align each $f_{\text{assign}(v)}$ to all remaining literals for v. □

Claims 1 and 2 together show that $\mathcal{D}$ is NP-complete, and therefore that $\mathcal{O}$ is NP-hard.

3.2 Reducing Perfect Matching to $\mathcal{S}$

With another construction, we can show that $\mathcal{S}$ is #P-hard, meaning that it is at least as hard as any #P-complete problem. #P is a class of counting problems related to NP, and #P-hard problems are NP-hard as well.

COUNTING PERFECT MATCHINGS, CPM
Given a bipartite graph G with $2n$ vertices, count the number of matchings of size n.

For a bipartite graph G with edge set $E = \{(v_j, v_l)\}$, we construct $\mathbf{e}$ and $\mathbf{f}$ with n words each, and set $\phi(e_{j-1\ j}, f_{l-1\ l}) = 1$ and 0 otherwise. The number of perfect matchings in G is the sum $\mathcal{S}$ for this weighted sentence pair. CPM is #P-complete (Valiant, 1979), so $\mathcal{S}$ (and hence $\mathcal{E}$) is #P-hard.

4 Solving the Optimization Problem

Although $\mathcal{O}$ is NP-hard, we present an approach to solving it using integer linear programming (ILP).

4.1 Previous Inference Approaches

Marcu and Wong (2002) describes an approximation to $\mathcal{O}$. Given a weighted sentence pair, high scoring phrases are linked together greedily to reach an initial alignment. Then, local operators are applied to hill-climb $\mathcal{A}$ in search of the maximum $\mathbf{a}$. This procedure also approximates $\mathcal{E}$ by collecting weighted counts as the space is traversed.

DeNero et al. (2006) instead proposes an exponential-time dynamic program to systematically explore $\mathcal{A}$, which can in principle solve either $\mathcal{O}$ or $\mathcal{E}$. In practice, however, the space of alignments has to be pruned severely using word alignments to control the running time of EM.

Notably, neither of these inference approaches offers any test to know if the optimal alignment is ever found. Furthermore, they both require small data sets due to computational expense.

4.2 Alignment via an Integer Program

We cast $\mathcal{O}$ as an ILP problem, for which many optimization techniques are well known. First, we in-

troduce binary indicator variables $a_{i,j,k,l}$ denoting whether $(e_{ij}, f_{kl}) \in \mathbf{a}$. Furthermore, we introduce binary indicators $e_{i,j}$ and $f_{k,l}$ that denote whether some $(e_{ij}, \cdot)$ or $(\cdot, f_{kl})$ appears in $\mathbf{a}$, respectively. Finally, we represent the weight function ϕ as a weight vector in the program: $w_{i,j,k,l} = \log \phi(e_{ij}, f_{kl})$.

Now, we can express an integer program that, when optimized, will yield the optimal alignment of our weighted sentence pair.

$$\max \quad \sum_{i,j,k,l} w_{i,j,k,l} \cdot a_{i,j,k,l}$$

$$\text{s.t.} \quad \sum_{i,j:i<x\leq j} e_{i,j} = 1 \qquad \forall x : 1 \leq x \leq |e| \quad (1)$$

$$\sum_{k,l:k<y\leq l} f_{k,l} = 1 \qquad \forall y : 1 \leq y \leq |f| \quad (2)$$

$$e_{i,j} = \sum_{k,l} a_{i,j,k,l} \qquad \forall i, j \quad (3)$$

$$f_{k,l} = \sum_{i,j} a_{i,j,k,l} \qquad \forall k, l \quad (4)$$

with the following constraints on index variables:

$$0 \leq i < |\mathbf{e}|, \quad 0 < j \leq |\mathbf{e}|, \quad i < j$$
$$0 \leq k < |\mathbf{f}|, \quad 0 < l \leq |\mathbf{f}|, \quad k < l \quad .$$

The objective function is $\log \phi(\mathbf{a})$ for $\mathbf{a}$ implied by $\{a_{i,j,k,l} = 1\}$. Constraint equation 1 ensures that the English phrases form a partition of **e** – each word in **e** appears in exactly one phrase – as does equation 2 for **f**. Constraint equation 3 ensures that each phrase in the chosen partition of **e** appears in exactly one link, and that phrases not in the partition are not aligned (and likewise constraint 4 for **f**).

5 Applications

The need to find an optimal phrase alignment for a weighted sentence pair arises in at least two applications. First, a generative phrase alignment model can be trained with Viterbi EM by finding optimal phrase alignments of a training corpus (approximate E-step), then re-estimating phrase translation parameters from those alignments (M-step).

Second, this is an algorithm for *forced decoding*: finding the optimal phrase-based derivation of a particular target sentence. Forced decoding arises in online discriminative training, where model updates are made toward the most likely derivation of a gold translation (Liang et al., 2006).

Sentences per hour on a four-core server	20,000
Frequency of optimal solutions found	93.4%
Frequency of ϵ-optimal solutions found	99.2%

Table 1: The solver, tuned for speed, regularly reports solutions that are within 10^{-5} of optimal.

Using an off-the-shelf ILP solver,[4] we were able to quickly and reliably find the globally optimal phrase alignment under $\phi(e_{ij}, f_{kl})$ derived from the Moses pipeline (Koehn et al., 2007).[5] Table 1 shows that finding the optimal phrase alignment is accurate and efficient.[6] Hence, this simple search technique effectively addresses the intractability challenges inherent in evaluating new phrase alignment ideas.

References

Colin Cherry and Dekang Lin. 2007. Inversion transduction grammar for joint phrasal translation modeling. In *NAACL-HLT Workshop on Syntax and Structure in Statistical Translation*.

John DeNero, Dan Gillick, James Zhang, and Dan Klein. 2006. Why generative phrase models underperform surface heuristics. In *NAACL Workshop on Statistical Machine Translation*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *HLT-NAACL*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL*.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *ACL*.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *EMNLP*.

Leslie G. Valiant. 1979. The complexity of computing the permanent. In *Theoretical Computer Science 8*.

Richard Zens, Hermann Ney, Taro Watanabeand, and E. Sumita. 2004. Reordering constraints for phrase based statistical machine translation. In *Coling*.

[4] We used Mosek: `www.mosek.com`.

[5] $\phi(e_{ij}, f_{kl})$ was estimated using the relative frequency of phrases extracted by the default Moses training script. We evaluated on English-Spanish Europarl, sentences up to length 25.

[6] ILP solvers include many parameters that trade off speed for accuracy. Substantial speed gains also follow from explicitly pruning the values of ILP variables based on prior information.