Analysis of Link Grammar on Biomedical Dependency Corpus Targeted at Protein-Protein Interactions

Sampo Pyysalo, Filip Ginter, Tapio Pahikkala, Jorma Boberg, Jouni Järvinen, Tapio Salakoski
Turku Centre for Computer Science (TUCS)
and Dept. of computer science, University of Turku
Lemminkäisenkatu 14A
20520 Turku, Finland,
first_name.last_name@it.utu.fi

Jeppe Koivula
MediCel Ltd.,
Haartmaninkatu 8
00290 Helsinki, Finland,
jeppe.koivula@medicel.com

Abstract

In this paper, we present an evaluation of the Link Grammar parser on a corpus consisting of sentences describing protein-protein interactions. We introduce the notion of an interaction subgraph, which is the subgraph of a dependency graph expressing a protein-protein interaction. We measure the performance of the parser for recovery of dependencies, fully correct linkages and interaction subgraphs. We analyze the causes of parser failure and report specific causes of error, and identify potential modifications to the grammar to address the identified issues. We also report and discuss the effect of an extension to the dictionary of the parser.

1 Introduction

The challenges of processing the vast amounts of biomedical publications available in databases such as MEDLINE have recently attracted a considerable interest in the Natural Language Processing (NLP) research community. The task of information extraction, commonly targeting entity relationships, such as protein-protein interactions, is an often studied problem to which various NLP methods have been applied, ranging from keyword-based methods (see, e.g., Ginter et al. (2004)) to full syntactic analysis as employed, for example, by Craven and Kumlien (1999), Temkin and Gilder (2003) and Daraselia et al. (2004).

In this paper, we focus on the syntactic analysis component of an information extraction system targeted to find protein-protein interactions from the dependency output produced by the Link Grammar[1] (LG) parser of Sleator and Temperley (1991). Two recent papers study LG in the context of biomedical NLP. The work by Szolovits (2003) proposes a fully automated method to extend the dictionary of the LG parser with the UMLS Specialist[2] lexicon, and Ding et al. (2003) perform a basic evaluation of LG performance on biomedical text. As both papers suggest, LG will require modifications in order to provide a correct analysis of grammatical phenomena that are rare in general English text, but common in biomedical language. Implementing such modifications is a major effort that requires a careful analysis of the performance of the LG parser to identify the most common causes of parsing failures and to target modification efforts.

While Szolovits (2003) does not attempt to evaluate parser performance at all and Ding et al. (2003) provide only an informal evaluation on manually simplified sentences, we focus on a more formal evaluation of the LG parser. For the purpose of this study and also for subsequent research of biomedical information extraction with the LG parser, we have developed a hand-annotated corpus consisting of unmodified sentences from publications. We use this corpus to evaluate the performance of the LG parser and to identify problems and potential improvements to the grammar and parser.

2 Link Grammar and parser

The Link Grammar and its parser represent an implementation of a dependency-based computational grammar. The result of LG analysis for a sentence is a labeled undirected simple graph, whose nodes represent the words of the sentence and whose edges and their labels express the grammatical relationships between the words. In LG terminology, the graph is called a *linkage*, and its edges are called *links*. The linkage must be planar (i.e., links must not cross) when drawn above the words of the sentence, and the labels of the links must satisfy the linking constraints specified for each word in the grammar. A connected linkage is termed *complete*.

[1] `http://www.link.cs.cmu.edu/link/`

[2] `http://www.nlm.nih.gov/research/umls/`

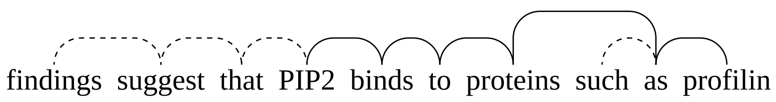
findings suggest that PIP2 binds to proteins such as profilin

Figure 1: Annotation example. The interaction of two proteins, *PIP2* and *profilin*, is stated by the words *binds to*. The links joining these words form the interaction subgraph (drawn with solid lines).

Due to the structural ambiguity of natural language, several linkages can typically be constructed for an input sentence. In such cases, the LG parser enumerates all linkages allowed by the grammar. A post-processing step is then employed to enforce a number of additional constraints. The number of linkages for some sentences can be very high, making post-processing and storage prohibitively expensive. This problem is addressed in the LG parser by defining $k_{\max}$, the maximal number of linkages to be post-processed. If the parsing algorithm produces more than $k_{\max}$ linkages, the output is reduced to $k_{\max}$ linkages by random sampling. The linkages are then ordered from best to worst using heuristic goodness criteria.

In order to be usable in practice, a parser is typically required to provide a partial analysis of a sentence for which it cannot construct a full analysis. If the LG parser cannot construct a complete linkage for a sentence, the connectedness requirement is relaxed so that some words do not belong to the linkage at all. The LG parser is also time-limited. If the full set of linkages cannot be constructed in a given time $t_{\max}$, the parser enters a *panic mode*, in which it performs an efficient but considerably restricted parse, resulting in reduced performance. The parameters $t_{\max}$ and $k_{\max}$ set the trade-off between the qualitative performance and the resource efficiency of the parser.

3 Corpus annotation and interaction subgraphs

To compile a corpus of sentences describing protein-protein interactions, we first selected pairs of proteins that are known to interact from the Database of Interacting Proteins[3]. We entered these pairs as search terms into the PubMed retrieval system. We then split the publication abstracts returned by the searches into sentences and included titles. These were again searched for the protein pairs. This gave us a set of 1927 sentences that contain the names of at least two proteins that are known to interact. A domain expert annotated these sentences for protein names and for words stating their interactions. Of these sentences, 1114 described at least one protein-protein interaction.

Thereafter, we performed a dependency analysis and produced annotation of dependencies. To minimize the amount of mistakes, each sentence was independently annotated by two annotators and differences were then resolved by discussion. The assigned dependency structure was produced according to the LG linkage conventions. Link types were not included in the annotation, and no cycles were introduced in the dependency graphs. All ambiguities where the LG parser is capable of at least enumerating all alternatives (such as prepositional phrase attachment) were enforced in the annotation. A random sample consisting of 300 sentences, including 28 publication titles, has so far been fully annotated, giving 7098 word-to-word dependencies. This set of sentences is the corpus we refer to in the following sections.

An information extraction system targeted at protein-protein interactions and their types needs to identify three constituents that express an interaction in a sentence: the proteins involved and the word or phrase that states their interaction and suggests the type of this interaction. To extract this information from a LG linkage, the links connecting these items must be recovered correctly by the parser. The following definition formalizes this notion.

Definition 1 (Interaction subgraph) *The interaction subgraph for an interaction between two proteins A and B in a linkage $\mathcal{L}$ is the minimal connected subgraph of $\mathcal{L}$ that contains A, B, and the word or phrase that states their interaction.*

The recovery of a connected component containing the protein names and the interaction word is not sufficient: by the definition of a complete linkage, such a component is always present. Consequently, the exact set of links

[3] `http://dip.doe-mbi.ucla.edu/`

that forms the interaction subgraph must be recovered.

For each interaction stated in a sentence, the corpus annotation specifies the proteins involved and the interaction word. The interaction subgraph for each interaction can thus be extracted automatically from the corpus. Because the corpus does not contain cyclic dependencies, the interaction subgraphs are unique. 366 interaction subgraphs were identified from the corpus, one for each described interaction. The interaction subgraphs can be partially overlapping, because a single link can be part of more than one interaction subgraph. Figure 1 shows an example of an annotated text fragment.

4 Evaluation criteria

We evaluated the performance of the LG parser according to the following three quantitative criteria:

- Number of dependencies recovered
- Number of fully correct linkages
- Number of interaction subgraphs recovered

The number of recovered dependencies gives an estimate of the probability that a dependency will be correctly identified by the LG parser (this criterion is also employed by, e.g., Collins et al. (1999)). The number of fully correct linkages, i.e. linkages where all annotated dependencies are recovered, measures the fraction of sentences that are parsed without error. However, a fully correct linkage is not necessary to extract protein-protein interactions from a sentence; to estimate how many interactions can potentially be recovered, we measure the number of interaction subgraphs for which all dependencies were recovered.

For each criterion, we measure the performance for the *first linkage* returned by the parser. However, the first linkage as ordered by the heuristics of the LG parser was often not the best (according to the criteria above) of the linkages returned by the parser. To separate the effect of the heuristics from overall LG performance, we identify separately for each of the three criteria the *best linkage* among the linkages returned by the parser, and we also report performance for the best linkages.

We further divide the parsed sentences into three categories: (1) sentences for which the time $t_{\max}$ for producing a normal parse was exhausted and the parser entered *panic* mode, (2) sentences where linkages were *sampled* because more than $k_{\max}$ linkages were produced, and (3) *stable* sentences for which neither of these occurred. A full analysis of all linkages that the grammar allows is only possible for stable sentences. For sentences in the other two categories, random effects may affect the results: sentences for which more than $k_{\max}$ linkages are produced are subject to randomness in sampling, and sentences where the parser enters panic mode were always subject to subsequent sampling in our experiments.

5 Evaluation

To evaluate the ability of the LG parser to produce correct linkages, we increased the number of stable sentences by setting the $t_{\max}$ parameter to 10 minutes and the $k_{\max}$ parameter to 10000 instead of using the defaults $t_{\max} = 30$ seconds and $k_{\max} = 1000$. When parsing the corpus using these parameters, 28 sentences fell into the panic category, 61 into the sampled category, and 211 were stable. The measured parser performance for the corpus is presented in Table 1.

While the fraction of sentences that have a fully correct linkage as the first linkage is quite low (approximately 7%), for 28% of sentences the parser is capable of producing a fully correct linkage. Performance was especially poor for the publication titles in the corpus. Because titles are typically fragments not containing a verb, and LG is designed to model full clauses, the parser failed to produce a fully correct linkage for any of the titles.

The performance for recovered interaction subgraphs is more encouraging, as 25% of the subgraphs were recovered in the first linkage and more than half in the best linkage. Yet many interaction subgraphs remain unrecovered by the parser: the results suggest an upper limit of approximately 60% to the fraction of protein-protein interactions that can be recovered from any linkage produced by the unmodified LG. In the following sections we further analyze the reasons why the parser fails to recover all dependencies.

5.1 Panics

No fully correct linkages and very few interaction subgraphs were found in the panic mode. This effect may be partly due to the complexity of the sentences for which the parser en-

		Category			
Criterion	Linkage	Stable	Sampled	Panic	Overall
Dependency	First linkage	3242 (80.0%)	1376 (74.3%)	569 (52.3%)	5187 (73.1%)
	Best linkage	3601 (86.6%)	1576 (85.0%)	620 (57.0%)	5797 (81.7%)
	Total	4157	1853	1088	7098
Fully correct	First linkage	22 (10.4%)	0 (0.0%)	0 (0.0%)	22 (7.3%)
	Best linkage	79 (37.4%)	6 (9.8%)	0 (0.0%)	85 (28.3%)
	Total	211	61	28	300
Interaction subgraph	First linkage	75 (30.5%)	16 (20.2%)	0 (0.0%)	91 (24.9%)
	Best linkage	156 (63.4%)	49 (62.0%)	4 (9.8%)	209 (57.1%)
	Total	246	79	41	366

Table 1: Parser performance. The fraction of fulfilled criteria is shown by category (the criteria and categories are explained in Section 4). The total rows give the number of criteria for each category, and the overall column gives combined results for all categories.

tered panic mode. The effect of panics can be better estimated by forcing the parser to bypass standard parsing and to directly apply panic options. For the 272 sentences where the parser did not enter the panic mode, 77% of dependencies were recovered in the first linkage. When these sentences were parsed in forced panic mode, 67% of dependencies were recovered, suggesting that on average parses in panic mode recover approximately 10% fewer dependencies than in standard parsing mode. Similarly, the number of fully correct first linkages decreased from 22 to 6 and the number of interaction subgraphs recovered in the first linkage from 91 to 65. These numbers indicate that panics are a significant cause of error.

Experiments indicate than on a 1GHz machine approximately 40% of sentences can be fully parsed in under a second, 80% in under 10 seconds and 90% within 10 minutes; yet approximately 5% of sentences take more than an hour to fully parse. With $t_{\max}$ set to 10 minutes, the total parsing time was 165 minutes.

Long parsing times are caused by ambiguous sentences for which the parser creates thousands or even millions of alternative linkages. In addition to simply increasing the time limit, the fraction of sentences where the parser enters the panic mode could therefore be reduced by reducing the ambiguity of the sentences, for example, by extending the dictionary of the parser (see Section 7).

5.2 Heuristics

When several linkages are produced for a sentence, the LG parser applies heuristics to order the sentences so that linkages that are more likely to be correct are presented first. The heuristics are based on examination and intuitions on general English, and may not be optimal for biomedical text. Note in Table 1 that both for recovered full linkages and interaction subgraphs, the number of items that were recovered in the best linkage is more than twice the number recovered in the first linkage, suggesting that a better ordering heuristic could dramatically improve the performance of the parser. Such improvements could perhaps be achieved by tuning the heuristics to the domain or by adopting a probabilistic ordering model.

6 Failure analysis

A significant fraction of dependencies were not recovered in any linkage, even in sentences where resources were not exhausted. In order to identify reasons for the parser failing to recover the correct dependencies, we analyze sentences for which it is certain that the grammar cannot produce a fully correct linkage. We thus analyzed the 132 stable sentences for which some dependencies were not recovered.

For each sentence, we attempt to identify the reason for the failure of the parser. For each identified reason, we manually edit the sentence to remove the source of failure. We repeat this procedure until the parser is capable of producing a correct parse for the sentence. Note that this implies that also the interaction subgraphs in the sentence are correctly recovered, and therefore the reasons for failures to recover interaction subgraphs are a subset of the identified issues. The results of the analysis are

Reason for failure	Cases
Unknown grammatical structure	72 (34.4%)
Dictionary issue	54 (25.8%)
Unknown word handling	35 (16.7%)
Sentence fragment	27 (12.9%)
Ungrammatical sentence	17 (8.1%)
Other	4 (1.9%)

Table 2: Results of failure analysis

summarized in Table 2. In many of the sentences, more than one reason for parser failure was found; in total 209 issues were identified in the 132 sentences. The results are described in more detail in the following sections.

6.1 Fragments and ungrammatical sentences

As some of the analyzed sentences were taken from publication titles, not all of them were full clauses. To identify further problems when parsing fragments not containing a verb, the phrase "is explained" and required determiners were added to these fragments, a technique used also by Ding et al. (2003). The completed fragments were then analyzed for potential further problems.

A number of other ungrammatical sentences were also encountered. The most common problem was the omission of determiners, but some other issues such as missing possessive markers and errors in agreement (e.g., "expressions. . . has") were also encountered.

Ungrammatical sentences pose interesting challenges for parsing. Because many authors are not native English speakers, a greater tolerance for grammatical mistakes should allow the parser to identify the intended parse for more sentences. Similarly, the ability to parse publication titles would extend the applicability of the parser; in some cases it may be possible to extract information concerning the key findings of a publication from the title. However, while relaxing completeness and correctness requirements, such as mandatory determiners and subject-predicate agreement, would allow the parser to create a complete linkage for more sentences, it would also be expected to lead to increased ambiguity for all sentences, and subsequent difficulties in identifying the correct linkage. If the ability to parse titles is considered important, a potential solution not incurring this cost would be to develop a separate version of the grammar for parsing titles.

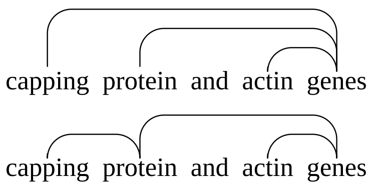
capping protein and actin genes
capping protein and actin genes

Figure 2: Multiple modifier coordination problem. Above: correct linkage disallowed by the LG parser. Below: solution by chaining modifiers.

6.2 Unknown grammatical structures

The method of the LG implementation for parsing coordinations was found to be a frequent cause of failures. A specific coordination problem occurs with multiple noun-modifiers: the parser assumes that coordinated constituents can be connected to the rest of the sentence through exactly one word, and the grammar attaches all noun-modifiers to the head. Biomedical texts frequently contain phrases that cause these requirements to conflict: for example, in the phrase "capping protein and actin genes" (where "capping protein genes" and "actin genes" is the intended parse), the parser allows only one of the words "capping" and "protein" to connect to the word "genes", and is thus unable to produce the correct linkage (for illustration, see Figure 2(a)).

This multiple modifier coordination issue could be addressed by modifying the grammar to chain modifiers (Figure 2(b)). This alternative model is adopted by another major dependency grammar, the EngCG-based Connexor Machinese. The problem could also be addressed by altering the coordination handling system in the parser.

Other identified grammatical structures not known to the parser were number postmodifiers to nouns (e.g., "serine 38"), specifiers in parentheses (e.g., "profilin mutant (H119E)"), coordination with the phrase "but not", and various unknown uses of colons and quotes. Single instances of several distinct unknown grammatical structures were also noted (e.g., "5 to 10", "as expected from", "most concentrated in"). Most of these issues can be addressed by local modifications to the grammar.

6.3 Unknown word handling

The LG parser assigns unknown words to categories based on morphological or other surface clues when possible. For remaining unknown words, parses are attempted by assigning the words to the generic noun, verb and adjective types in all possible combinations.

Some problems with the unknown word processing method were encountered during analysis; for example, the assumption that unknown capitalized words are proper nouns often caused failures, especially in sentences beginning with an unknown word. Similarly, the assumption that words containing a hyphen behave as adjectives was violated by a number of unknown verbs (e.g., “cross-links”).

Another problem that was noted occurred with lowercase unknown words that should be treated as proper nouns: because LG does not allow unknown lowercase words to act as proper nouns, the parser assigns incorrect structure to a number of phrases containing words such as “actin”. Improving unknown word handling requires some modifications to the LG parser.

6.4 Dictionary issues

Cases where the LG dictionary contains a word, but not in the sense in which it appears in a sentence, almost always lead to errors. For example, the LG dictionary does not contain the word “assembly” in the sense “construction”, causing the parser to erroneously require a determiner for “protein assembly”[4]. A related frequent problem occurred with proper names headed by a common noun, where the parser expects a determiner for such names (e.g., “myosin heavy chain”), and fails when one is not present. These issues are mostly straightforward to address in the grammar, but difficult to identify automatically.

6.5 Biomedical entity names

Many of the causes for parser failure discussed above are related to the presence of biomedical entity names. While the causes for failures relating to names can be addressed in the grammar, the existence of biomedical named entity (NE) recognition systems (for a recent survey, see, e.g., Bunescu et al. (2004)) suggests an alternative solution: NEs could be identified in preprocessing, and treated as single (proper noun) tokens during the parse. During failure analysis, 59 cases (28% of all cases) were noted where this procedure would have eliminated the error, assuming that no errors are made in NE recognition. However, the performance of current NE recognition systems is not perfect, and it is not clear what the effect of adopting such a method would be on parser performance.

7 Dictionary extension

Szolovits (2003) describes an automatic method for mapping lexical information from one lexicon to another, and applies this method to augment the LG dictionary with terms from the extensive UMLS Specialist lexicon. The extension introduces more than 125,000 new words into the LG dictionary, more than tripling its size. We evaluated the effect of this dictionary extension on LG parser performance using the criteria described above. The fraction of distinct tokens in the corpus found in the parser dictionary increased from 52% to 72% with the dictionary extension, representing a significant reduction in uncertainty. This decrease was coupled with a 32% reduction in total parsing time.

Because the LG parser is unable to produce any linkage for sentences where it cannot identify a verb (even incorrectly), extending the dictionary significantly reduced the ability of LG to extract dependencies in titles, where the fraction of recovered dependencies fell from the already low value of 67% to 55%.

For the sentences excluding titles, the benefits of the dictionary extension were most significant for sentences that were in the panic category when using the unextended LG dictionary; 12 of these 28 sentences could be parsed without panic with the dictionary extension. In the first linkage of these sentences, the fraction of recovered dependencies increased by 8%, and the fraction of recovered interaction subgraphs increased from zero to 15% with the dictionary extension.

The overall effect of the dictionary extension was positive but modest, with no more than 2.5% improvement for either the first or best linkages for any criterion, despite the threefold increase in dictionary size. This result agrees with the failure analysis: most problems cannot be removed by extending the dictionary and must instead be addressed by modifications of the grammar or parser.

8 Conclusion

We have presented an analysis of Link Grammar performance using a custom dependency corpus targeted at protein-protein interactions. We introduced the concept of the interaction

[4] 30 distinct problematic word definitions were identified, including “breakdown”, “composed”, “factor”, “half”, “independent”, “localized”, “parallel”, “promoter”, “segment”, “upstream” and “via”.

subgraph and reported parser performance for three criteria: recovery of dependencies, interaction subgraphs and fully correct linkages. While LG was able to recover 73% of dependencies in the first linkage, only 7% of sentences had a fully correct first linkage. However, fully correct linkages are not required for information extraction, and we found that 25% of interaction subgraphs were recovered in the first linkage.

Resource exhaustion was found to be a significant cause of poor performance. Furthermore, an evaluation of performance in the case when optimal heuristics for ordering linkages are applied indicated that the fraction of recovered interaction subgraphs could be more than doubled (to 57%) by optimal heuristics.

To further analyze the cases where the parser cannot produce a correct linkage, we carefully examined the sentences and were able to identify five problem types. For each identified case, we discussed potential modifications for addressing the problems. We also considered the possibility of using a named entity recognition system to improve parser performance and found that 28% of LG failures would be avoided by a flawless named entity recognition system.

We evaluated the effect of the dictionary extension proposed by Szolovits (2003), and found that while it significantly reduced ambiguity and improved performance for the most ambiguous sentences, overall improvement was only 2.5%. This indicates that extending the dictionary is not sufficient to address the performance problems and that modifications to the grammar and parser are necessary.

The quantitative analysis of LG performance confirms that, in its current state, LG is not well suited to the IE task discussed. However, in the failure analysis we have identified a number of specific issues and problematic areas for LG in parsing biomedical publications, and suggested improvements for adapting the parser to this domain. The examination and implementation of these improvements is a natural follow-up of this study. Our initial experiments suggest that it is indeed possible to implement general solutions to many of the discussed problems, and such modifications would be expected to lead to improved applicability of LG to the biomedical domain.

9 Acknowledgments

This work has been supported by Tekes, the Finnish National Technology Agency.

References

Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. 2004 (to appear). Comparative experiments on learning information extractors for proteins and their interactions. *Artificial Intelligence in Medicine. Special Issue on Summarization and Information Extraction from Medical Documents.*

Michael Collins, Jan Hajic, Lance Ramshaw, and Christoph Tillmann. 1999. A statistical parser for Czech. In *37th Annual Meeting of the Association for Computational Linguistics*, pages 505–512. Association for Computational Linguistics, Somerset, New Jersey.

Mark Craven and Johan Kumlien. 1999. Constructing biological knowledge bases by extracting information from text sources. In T. Lengauer, R. Schneider, P. Bork, D. Brutlag, J. Glasgow, H HW Mewes, and Zimmer R., editors, *Proceedings of the 7th International Conference on Intelligent Systems in Molecular Biology*, pages 77–86. AAAI Press, Menlo Park, CA.

Nikolai Daraselia, Anton Yuryev, Sergei Egorov, Svetalana Novichkova, Alexander Nikitin, and Ilya Mazo. 2004. Extracting human protein interactions from MEDLINE using a full-sentence parser. *Bioinformatics*, 20(5):604–611.

Jing Ding, Daniel Berleant, Jun Xu, and Andy W. Fulmer. 2003. Extracting biochemical interactions from medline using a link grammar parser. In B. Werner, editor, *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence*, pages 467–471. IEEE Computer Society, Los Alamitos, CA.

Filip Ginter, Tapio Pahikkala, Sampo Pyysalo, Jorma Boberg, Jouni Järvinen, and Tapio Salakoski. 2004. Extracting protein-protein interaction sentences by applying rough set data analysis. In S. Tsumoto, R. Slowinski, J. Komorowski, and J.W. Grzymala-Busse, editors, *Lecture Notes in Computer Science 3066*. Springer, Heidelberg.

Daniel D. Sleator and Davy Temperley. 1991. Parsing english with a link grammar. Technical Report CMU-CS-91-196, Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

Peter Szolovits. 2003. Adding a medical lexicon to an english parser. In Mark Musen, editor, *Proceedings of the 2003 AMIA Annual Symposium*, pages 639–643. American Medical Informatics Association, Bethesda, MD.

Joshua M. Temkin and Mark R. Gilder. 2003. Extraction of protein interaction information from unstructured text using a context-free grammar. *Bioinformatics*, 19(16):2046–2053.