# HTML Table Interpretation by Sibling Page Comparison in the Molecular Biology Domain

Cui Tao and David W. Embley
Department of Computer Science, Brigham Young University, Provo, UT, 84602, USA
ctao,embley@cs.byu.edu

## Abstract

*There are large and growing amount of biological data that reside in different online repositories. Many of these repositories represent their data in tables. In order to automatically understand these online pages, a system that can interpret tables is desired. However, the longstanding problem of automatic table interpretation still illudes us [12]. We offer a solution for the common special case in which so-called sibling pages are available. Sibling pages, which are the pages commonly generated by underlying web databases, are compared to identify and connect non-varying components (category labels) and varying components (data values). We tested our solution on 862 HTML tables. Experimental results show that the system can successfully identify sibling tables, generate structure patterns, interpret different tables using the generated patterns, and automatically adjust the structure patterns as needed.*
***Keywords:*** *Bioinformatics, table interpretation*

## 1 Introduction

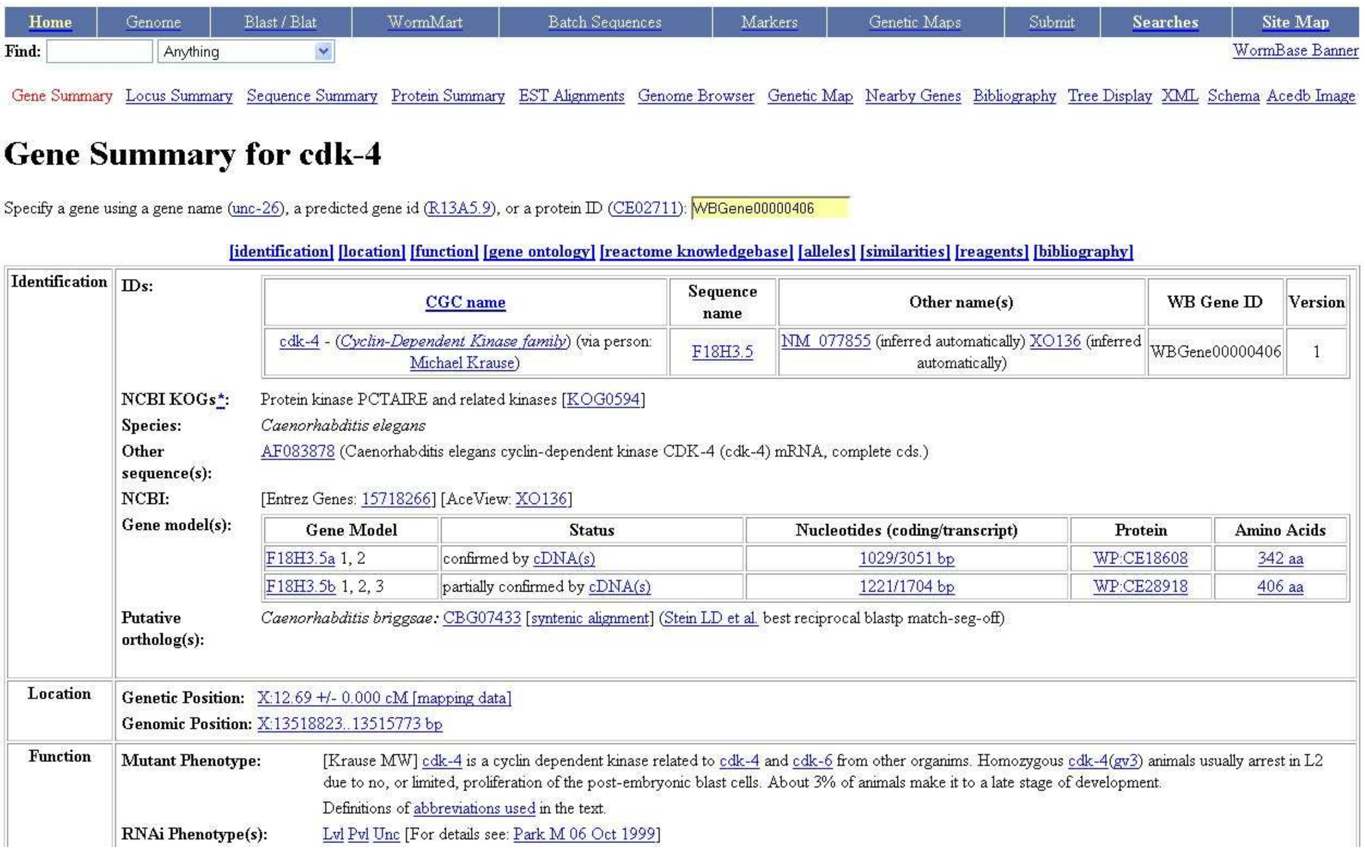
Catalyzed by world-wide research communities producing publicly available data, the volume of biological data is increasing at a rapid pace. Many online biological data repositories present their information in tables. Tables present information in a simplified and compact way in rows and columns. Data in one row/column usually belongs to the same category or provides values for the same concept. The labels of a row/column describe this category or concept.

Although a table with a simple row and column structure is common, tables can be much more complex. Figure 1 shows an example. The position of table category labels may vary in different tables. Labels commonly appear on the top or left. Occasionally, table designers position labels on the right side of a table. In long tables, labels sometimes appear at the end of a table or in the middle of a table, every few rows, in order to help a reader find the correspondence between labels and data. Sometimes tables are rearranged to fit the space available. Label-value pairs may appear in multiple columns across the pages or in multiple rows placed below on another down the page. Tables may themselves contain nested tables as does the table in Figure 1. These complexities make automatic table interpretation a challenging task.

## 2 Table Interpretation

To interpret a table is to properly associate table category labels with table data values. Using Figure 1 as an example, we see that *Identification*, *Location*, and *Function* are labels for the large rectangular table. Inside the right cell of the first row is another table with headers *IDs*, *NCBI KOGs*, *Species*, etc. Nested inside of this cell are two tables with labels *CGC name*, *Sequences name*, ... , *Version*, and *Gene Model*, *Status*, ..., and *Amino Acids*. The rest of the information in these tables are data values. Once category labels and data values are found, we want to properly associate them. For example, for the value $F_18H_{3.5}$, its associated label should be the sequences of labels *Identification*, *IDs*, and *Sequence name*. We associate one or more sequences of labels with each data value in a table (more, when the table is multi-dimensional). Borrowing notation from Wang [10], the representation of a label-value pair is look like: *Identification.IDs.Sequence name* $\rightarrow F8H_{3.5}$. The left hand side of the arrow is a sequence of one or more table labels, and the right hand side of the arrow is a data value.

Recent surveys [4, 12] describe the vast amount of research that has been done in table processing and illustrate the challenges of the table interpretation problem. We focus in this paper, however, only HTML tables. A number of HTML table extraction systems use machine learning to recognize tables in web pages (e.g. [3, 11]). Drawbacks of machine learning approaches, however, are that they need training data, and they need to be retrained for tables from different web sites. Other table interpretation systems work based on some simple assumptions and heuris-



Home | Genome | Blast / Blat | WormMart | Batch Sequences | Markers | Genetic Maps | Submit | Searches | Site Map

Find: Anything — WormBase Banner

Gene Summary  Locus Summary  Sequence Summary  Protein Summary  EST Alignments  Genome Browser  Genetic Map  Nearby Genes  Bibliography  Tree Display  XML  Schema  Acedb Image

**Gene Summary for cdk-4**

Specify a gene using a gene name (unc-26), a predicted gene id (R13A5.9), or a protein ID (CE02711): WBGene00000406

[identification] [location] [function] [gene ontology] [reactome knowledgebase] [alleles] [similarities] [reagents] [bibliography]

**Identification**

**IDs:**

| CGC name | Sequence name | Other name(s) | WB Gene ID | Version |
|---|---|---|---|---|
| cdk-4 - (*Cyclin-Dependent Kinase family*) (via person: Michael Krause) | F18H3.5 | NM_077855 (inferred automatically) XO136 (inferred automatically) | WBGene00000406 | 1 |

| | |
|---|---|
| **NCBI KOGs*:** | Protein kinase PCTAIRE and related kinases [KOG0594] |
| **Species:** | *Caenorhabditis elegans* |
| **Other sequence(s):** | AF083878 (Caenorhabditis elegans cyclin-dependent kinase CDK-4 (cdk-4) mRNA, complete cds.) |
| **NCBI:** | [Entrez Genes: 15718266] [AceView: XO136] |

**Gene model(s):**

| Gene Model | Status | Nucleotides (coding/transcript) | Protein | Amino Acids |
|---|---|---|---|---|
| F18H3.5a 1, 2 | confirmed by cDNA(s) | 1029/3051 bp | WP:CE18608 | 342 aa |
| F18H3.5b 1, 2, 3 | partially confirmed by cDNA(s) | 1221/1704 bp | WP:CE28918 | 406 aa |

**Putative ortholog(s):** *Caenorhabditis briggsae:* CBG07433 [syntenic alignment] (Stein LD et al. best reciprocal blastp match-seg-off)

**Location**

**Genetic Position:** X:12.69 +/- 0.000 cM [mapping data]

**Genomic Position:** X:13518823..13515773 bp

**Function**

**Mutant Phenotype:** [Krause MW] cdk-4 is a cyclin dependent kinase related to cdk-4 and cdk-6 from other organims. Homozygous cdk-4(gv3) animals usually arrest in L2 due to no, or limited, proliferation of the post-embryonic blast cells. About 3% of animals make it to a late stage of development.

Definitions of abbreviations used in the text.

**RNAi Phenotype(s):** Lvl Pvl Unc [For details see: Park M 06 Oct 1999]

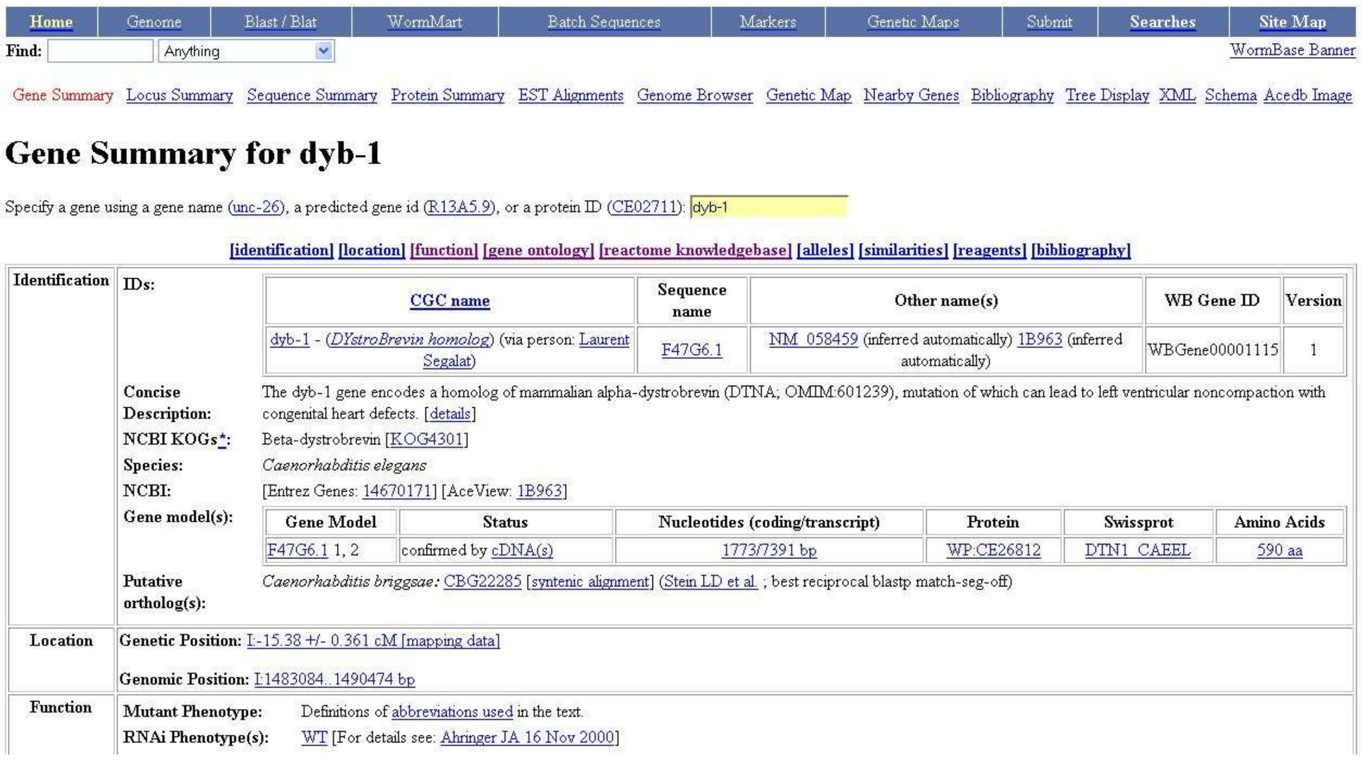
**Figure 1. A sample page from [1].**

tics (e.g. [2, 5, 8]). These simple assumptions (labels are either the first row or the first column) are easily broken in complex tables. More sophisticated table interpretation techniques have appeared in recent papers [6, 7, 9]. None of this research makes use of sibling tables, but is complementary to our work and could potentially be used in conjunction with our work in future efforts to improve results for certain cases.

## 3 Sibling Page Comparison

If we have another page, such as the one in Figure 2, that has the same structure as the one in Figure 1, the system maybe able to obtain enough information about the structure to make automatic interpretation possible. Molecular biology web resources usually generate output pages after receiving a user query by placing the results into a predefined page structure. Thus, pages from the same web site are usually structured in the same way. We call pages that are from the same web site and have similar structures *sibling pages*. The two pages in Figures 1 and 2 are a pair of sibling pages. They have the same basic structure, with the same top banners that appear in all the pages from this web site, with the same table title (*Gene Summary for* some particular gene), and a table that contains information about the gene. Corresponding tables in sibling pages are called *sibling tables*. If we compare the two large tables in the main part of the sibling pages, we can see that the first columns of each table are exactly the same. If we look at the cells under the *Identification* label in the two tables, both contain another table with two columns. In both cases, the first column contains identical labels *IDs*, *NCBI KOGs*, ..., *Putative ortholoy(s)*. Further, the tables under *Identification.IDs* also have identical header rows. The data rows, however, vary considerably. General speaking, we can look for commonalities to find labels and look for variations to find values.

Given that we can find most of the label and data cells in this way, our next task is to infer the general structure pattern of the web site and of the individual tables embedded within pages of the web site. For each table, we first locate the positions of values and labels. For example, consider the two nested tables in Figures 1 and 2 that start with *CGC name*. The top rows of the tables on the two pages are identical while the two second rows vary considerably. We thus determine that the first row is a row of labels and the second row is a row of values. Depends on the posi-



| Home | Genome | Blast / Blat | WormMart | Batch Sequences | Markers | Genetic Maps | Submit | Searches | Site Map |
|---|---|---|---|---|---|---|---|---|---|

Find: Anything WormBase Banner

Gene Summary Locus Summary Sequence Summary Protein Summary EST Alignments Genome Browser Genetic Map Nearby Genes Bibliography Tree Display XML Schema Acedb Image

**Gene Summary for dyb-1**

Specify a gene using a gene name (unc-26), a predicted gene id (R13A5.9), or a protein ID (CE02711): dyb-1

[identification] [location] [function] [gene ontology] [reactome knowledgebase] [alleles] [similarities] [reagents] [bibliography]

**Identification**

**IDs:**

| CGC name | Sequence name | Other name(s) | WB Gene ID | Version |
|---|---|---|---|---|
| dyb-1 - (*DYstroBrevin homolog*) (via person: Laurent Segalat) | F47G6.1 | NM_058459 (inferred automatically) 1B963 (inferred automatically) | WBGene00001115 | 1 |

**Concise Description:** The dyb-1 gene encodes a homolog of mammalian alpha-dystrobrevin (DTNA; OMIM:601239), mutation of which can lead to left ventricular noncompaction with congenital heart defects. [details]

**NCBI KOGs*:** Beta-dystrobrevin [KOG4301]

**Species:** *Caenorhabditis elegans*

**NCBI:** [Entrez Genes: 14670171] [AceView: 1B963]

**Gene model(s):**

| Gene Model | Status | Nucleotides (coding/transcript) | Protein | Swissprot | Amino Acids |
|---|---|---|---|---|---|
| F47G6.1 1, 2 | confirmed by cDNA(s) | 1773/7391 bp | WP:CE26812 | DTN1_CAEEL | 590 aa |

**Putative ortholog(s):** *Caenorhabditis briggsae:* CBG22285 [syntenic alignment] (Stein LD et al. ; best reciprocal blastp match-seg-off)

**Location**

**Genetic Position:** I:-15.38 +/- 0.361 cM [mapping data]

**Genomic Position:** I:1483084..1490474 bp

**Function**

**Mutant Phenotype:** Definitions of abbreviations used in the text.

**RNAi Phenotype(s):** WT [For details see: Ahringer JA 16 Nov 2000]

**Figure 2. A second sample page from [1].**

tion of labels and values, we try to match the table with a pre-defined structure pattern templates or a combination of several pattern templates. Once we found a match, we generated a structure pattern for the set of sibling tables, which records the location of the table in each sibling page, the position of labels and values, and how the labels and values associate to each other.

Although we look for commonalities to find labels and look for variations to find data values, we must be careful about being too strict. Sometimes there are additional or missing label-value pairs. The two tables beginning with *Gene Model* Figures 1 and 2 do not share exactly the same structure. The table in Figure 1 has five columns and three rows, while the table in Figure 2 has six columns and two rows. Although they are not exactly the same, we can still identify the structure pattern by comparing them. The top rows in the two tables are very similar. Observe that the table in Figure 2 only has an additional *Swissprot* column inserted between the *Protein* and *Amino Acids* columns. Although the labels for the two tables are not identical, we can still tell that they are table headers.

In addition to discovering the structure pattern for a web site, we can also dynamically adjust the pattern if the system encounters a table that varies from the pattern. There are two ways to adjust a structure pattern: (1) adjust the location to locate a table. If the table in the recorded location does not match with the pattern, we check for the tables in the neighbor positions and see if we could find a match. If so, we add the new position in the pattern as an alternative location where we could possibly locate the sibling table in a new encountered sibling page. (2) adjust the information of labels. If there is an additional or missing label, the system can change the pattern by either adding the new label and marking it optional or marking the missing label optional. For example, if we had not seen the extra *Swissprot* column in our initial pair of sibling pages, the system can add *Swissprot* as a new label and mark it as optional.

## 4 Experimental Results

We collected 100 sibling pages from 10 different web sites in the molecular biology domain for a total of 862 HTML tables. Among these tables, the system falsely classified three pairs of layout tables as data tables. The system, however, successfully eliminated these false sibling pairs during pattern generation because it was unable to find a matching pattern. No false patterns were generated. The system was able to recognized 28 of 29 structure patterns. The system missed one pattern because the table contained too many empty cells. If we had considered empty cells as mismatches, the system would have correctly recognize this pattern. As the system processed additional sibling pages, it found 5 additional sibling tables and correctly interpreted all but one of them. The failure was caused by labels that varied across sibling tables causing them in some cases to look like values. There were 5 location adjustments and 12 label adjustments—all of them correct. One table was interpreted only partially correctly because the system considered the irrelevant information *To Top* as a header.

The time for the pattern generation given a pair of sibling pages consists of three parts: (1) the time to read and parse the two pages and locate all the HTML tables, (2) the time for sibling table comparisons, and (3) the time to select from pre-defined structure templates and generate the pattern. The complexity of parsing and locating HTML tables is O($n$), where $n$ is the number of HTML tags. The simple tree matching algorithm has time complexity O($m^2$), where $m$ is the number of nodes in each tree. To find the best match for each HTML table, the time complexity is O($k(m)^2$), where $k$ is the number of HTML tables in one sibling page. The time complexity for finding the correct pattern for each matched sibling table is O($pl$), where $p$ is the number of pattern templates and $l$ is the number of leaf nodes (cells) in the HTML table. If there is pattern combination involved, this complexity increase multiply. The time needed for the initial pattern generation for a pair of sibling pages is on average below or about one second, but reached a maximum of 15 seconds for a complicated web site where pages have more than with more than 20 tables on a Pentium 4 computer running at 3.2 GHz.

## 5 Conclusion

Many online biological repositories present their information in tables with complicated structures. In this paper, we introduced a system that can successfully interpret these tables automatically. Our system works based on sibling page comparison. By comparing sibling pages from the same site, we are able to find the location of table headers and data entries, and further we are be able to infer the general pattern for all pages from the same site.

## References

[1] Worm base! http://www.wormbase.org, 2005.

[2] H. Chen, S. Tsai, and J. Tsai. Mining tables from large scale HTML texts. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING'00)*, pages 166–172, Saarbrcken, German, Jul–Aug 2000.

[3] W. W. Cohen, M. Hurst, and L.S. Jensen. A flexible learning system for wrapping tables and lists in HTML documents. In *Proceedings of the International World Wide Web Conference (WWW'02)*, pages 232–241, Honolulu, Hawaii, May 2002.

[4] D. W. Embley, M. Hurst, D. Lopresti, and G. Nagy. Table processing paradigms: A research survey. *International Journal of Document Analysis and Recognition*, 8(2), 2006.

[5] D.W. Embley, C. Tao, and S.W. Liddle. Automating the extraction of data from html tables with unknown structure. *Data and Knowledge Engineering*, 54(1):3–28, July 2005.

[6] W. Gatterbauer and P. Bohunsky. Table extraction using spatial reasoning on the CSS2 visual box model. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI'06)*, Boston, Massachusetts, July 2006. (to appear).

[7] W. Gatterbauer, B. Krupl, W. Holzinger, and M. Herzog. Web information extraction using eupeptic data in web tables. In *Proceedings of the 1st International Workshop on Representation and Analysis of Web Space (RAWS'05)*, pages 41–48, Prague, Czech Republic, September 2005.

[8] S. Lim and Y. Ng. An automated approach for retrieving heirarchical data from HTML tables. In *Proceedings of the Eighth International Conference on Informaiton and Knowledge management (CIKM'99)*, pages 466–474, Kansas City, Missouri, November 1999.

[9] A. Pivk, P. Cimiano, and Y. Sure. From tables to frames. In *Proceedings of the Third International Semantic Web Conference (ISWC'04)*, volume 3298 of *Lecture Notes in Computer Science*, pages 166–181, Hiroshima, Japan, November 2004. Springer Verlag.

[10] X. Wang. *Tabular abstraction, editing, and formatting*. PhD thesis, Univeristy of Waterloo, 1996.

[11] Y. Wang and J. Hu. A machine learning based approach for table detection on the web. In *Proceedings of the 11th international conference on World Wide Web (WWW'02)*, pages 242–250, Honolulu, Hawaii, 2002.

[12] R. Zanibbi, D. Blostein, and J.R. Cordy. A survey of table recognition. *International Journal of Document Analysis and Recognition*, 7(1):1 – 16, 2004.