Pacific Symposium on Biocomputing 6:520-531 (2001)

# PIES, a Protein Interaction Extraction System

Limsoon Wong
*Kent Ridge Digital Labs*
*21 Heng Mui Keng Terrace, Singapore 119613*
*Email:* limsoon@krdl.org.sg

**Abstract**

We consider the problem of extracting, manipulating, and managing biological pathways, especially protein-protein interaction pathways. We discuss here the Protein Interaction Extraction System (PIES). PIES is contructed on top of three main technologies: Kleisli, BioNLP, and Graphviz. Kleisli is a broad-scale data integration system that we use for downloading Medline abstracts and for general manipulation and management of pathway/interaction databases. BioNLP is a natural language-based information extraction module that we use for analysing Medline abstracts and to extract precise protein-protein and other interaction information. Graphviz is a graphical layout package developed for directed graphs that we use for visualization of the extracted pathways. PIES can be augmented with various means for extracting protein interaction information from sequence databases, for example, by using Kleisli's power to integrate sequence comparison tools to detect gene fusion events in sequence databases.

## 1 Introduction

The creation and visualization of a large complex pathway diagram is an exacting task. It must be assembled one interaction by one interaction mostly by hand and changes could not be easily made as the drawing progresses. Thus the finished diagram is updated very infrequently. There are some notable recent developments that add interesting features to software systems involving pathway diagrams. Nevertheless, they do not completely address the needs for the creation and visualization of pathway diagrams.

E-CELL [19] is a system for whole-cell simulation. It is a very sophisticated system that can simulate a virtual cell. KEGG [15] is another sophisticated system that contains encyclopedic details of genes and genomes, and it provides many query facilities such as a function to predict the possible pathways that a protein may participate in. However, the pathways in E-CELL and KEGG are assembled by hand and the layout of their pathways are done by hand. EcoCyc [8] is a system containing detailed pathway information of *E. coli* genes and metabolism. The pathways are manually curated though it has

an algorithm for drawing these pathways automatically [9]. BioJAKE [17] is a pure drawing system for metabolic pathways. It does allows the drawings to be edited and annotated in a convenient way. However, its pathways must be created by hand an interaction at a time and there is no means for large-scale manipulation of the created pathways. For example, it does not allow two separately created pathways to be integrated automatically. There are several proposals for automated extraction of interaction pathways from Medline abstracts based on natural language processing [14,16] or co-occurrences of keywords [18]. There are also several proposals for predicting protein interactions from sequence databases based on gene fusion events [11,4]. These are specific techniques for making good guesses on individual protein interactions. These individual predictions still need to be assembled into pathways and visualized and managed using other means.

We introduce here our Protein Interaction Extraction System (PIES) that aims to automate a large proportion of the tasks of extracting, manipulating, managing, and visualizing protein interaction pathways[a] Some existing systems available to us are already fairly useful for many of the individual functionality requirements of PIES. The BioNLP system [14] is a good tool for analysing Medline abstracts and extracting precise protein interaction information. The Graphviz system [6] is a good tool for drawing directed graphs that can be adapted for drawing interaction pathways. The Kleisli system [2] is an excellent tool for data intergation and database-style manipulation. We therefore decide to build PIES by combining these tools.

The rest of this paper is organized as follows. Section 2 walks the Reader through an demonstration of PIES and discusses the general features of the system. Section 3 describes the architecture and the implementation of PIES. Section 4 contains concluding remarks.

## 2 Demonstration

This section demonstrates PIES with a few screendumps. A typical session in using PIES involves the user providing an initial search specification. Then the PIES downloads Medline abstracts satisfying that specification, extracts interaction information from these abstracts, and presents the extracted information in both textual and graphical forms. The presentation is followed up by the user performing further manipulations such as modifying the extracted interaction pathway, saving, or printing it. Alternatively, the user starts from a previously saved pathway and asks PIES to update it with new information

[a] We use the term "protein interaction pathways" to refer generically to pathways for signal transduction, regulation, and so on where protein interaction is a central feature.

or to merge it with another pathway, which is either previously saved or extracted on-the-fly with new search specification. Let us start from the user going to PIES website and entering his search specification. In our example, he enters "calyculin", as depicted in Figure 1.

PIES proceeds to obtain all abstracts from Medline satisfying the search specification "calyculin". Each abstract is analysed to identify sentences that mention interaction of proteins.[b] We consider only inhibit vs activate type of interactions. Each such sentence is considered an "evidence" for an interaction. The "actor" and "patient" of each interaction are identified. These interaction evidence sentences are then grouped by actor and patient. The result is then displayed in textual form. Figure 2 depicts a small fragment of this result. It shows an inhibition interaction, where "calyculin A" is the actor and "serine/threonine phosphatase" is the patient, and four of the many evidence sentences for this interaction.

PIES puts these interactions together into a pathway diagram where the nodes are the proteins and the directed edges denote interactions (green arrows for "activate" and red arrows for "inhibit") between the connected proteins. PIES then performs a visually pleasant graphical layout of the pathway diagram and displays it as a GIF file, as depicted in Figure 3. In addition, PIES also produces a Postscript file of the diagram. This Postscript file is "tiled" automatically by PIES so that a large pathway diagram can be printed on several pages that can be placed side-by-side to re-assemble the diagram.

Once the interactions and evidence sentences are extracted and the pathways assembled, the user can choose to manipulate the pathways further. He can do so by selecting appropriate options from a menu, such as further expanding the pathways by supplying additional keywords to search Medline, modifying the nodes and edges of the pathway diagram, or exporting the extracted results into an interaction database.

## 3 Implementation

The architecture of PIES is shown in Figure 4. The main components are the Kleisli Query System[2,3,20], which accepts the user's input and performs the downloading of appropriate Medline abstracts and various transformations on the interaction database corresponding the various manipulation commands from the user. The BioNLP module[14] analyses the Medline abstracts to extract from them protein interactions and evidence sentences for these interactions. The MkGIF module generates from these protein interactions a directed graph

[b] We use the term "protein" in this work. Our approach has been generalized to included drugs and other objects that participate in an interaction.

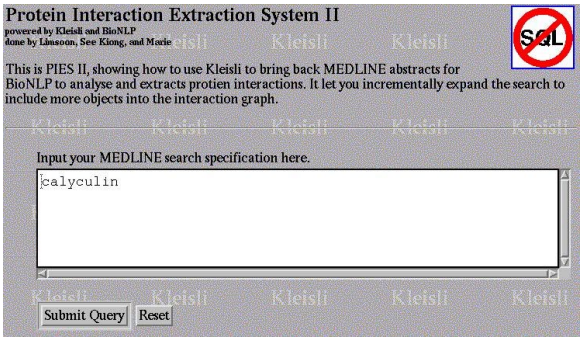


Figure 1: Example input to PIES.

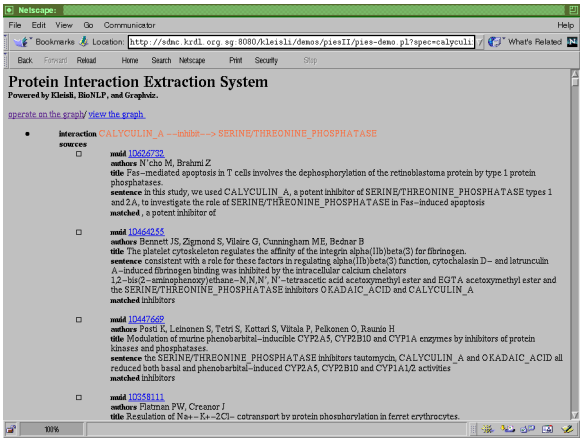


Figure 2: Example interaction and evidence extracted by PIES.

specification, The Graphviz package[6] uses this directed graph specification to perform an automatic layout of the pathway diagram and produces the GIF and Postscript files. The CPL2Perl module is used for interfacing the BioNLP and MkGIF modules to the interaction database produced by Kleisli. The implementation of these modules are now briefly described.

### *3.1 Creating Interaction Database*

The input to PIES is expected to be an interaction database in the form of a data file, a set of Medline abstracts in the form of a data file, or a Medline search specification in a form identical to that supported by Entrez[12]. The last type of input is the more interesting one, since it means PIES has to search Medline via the Entrez website for abstracts satisfying the specification.

This download step is performed by the Kleisli Query System. The Kleisli Query System[2,3,20] is an advanced broad-scale integration technology that has proved useful in the bioinformatics arena[1,10]. Many bioinformatics problems require access to data sources that are high in volume, highly heterogeneous and complex, constantly evolving, and geographically dispersed. Solutions to these problems usually involve multiple carefully sequenced steps and require information to be passed smoothly between the steps. Kleisli is designed to handle these requirements directly by providing a high-level query language, CPL, that can be used to express complicated transformation across multiple data sources in a clear and simple way. Kleisli/CPL itself is implemented using SML[13], a practical functional programming language. The capability of Kleisli to express complicated transformation is precisely what we need to create the initial interaction database from the Medline search specification. It is also what we exploit later to implement the various pathway manipulation functions of PIES.

**Kleisli Script 3.1** *The Kleisli script[c] for downloading Medline abstracts and creating the interaction database.*

```
writefile ml-get-abstract-general (SPEC) to "ARTICLES" using stdout;
writefile
  {(#muid:x.#muid, #sentence:x.#sentence,
    #matched:x.#matched, #interaction:y)
```

[c] The syntax of the Kleisli/CPL query language is as follows. $(\#l_1 : e_1, ..., \#l_n : e_n)$ constructs a record with fields $l_1, ..., l_n$ with values $e_1, ..., e_n$ respectively. $e.\#l$ returns the value in the $l$ field of the record $e$. $f(x)$ applies the function $f$ to the object $x$. $\{f(x) \mid \backslash x \leftarrow e, C(x)\}$ constructs a set consisting of every object $f(x)$ where $x$ is an object in the set $e$ and the test $C(x)$ is true. The query language also has a rich set of operators such as `ml-get-abstract-general` which accesses Medline and returns the set of abstracts satifying a search specification.

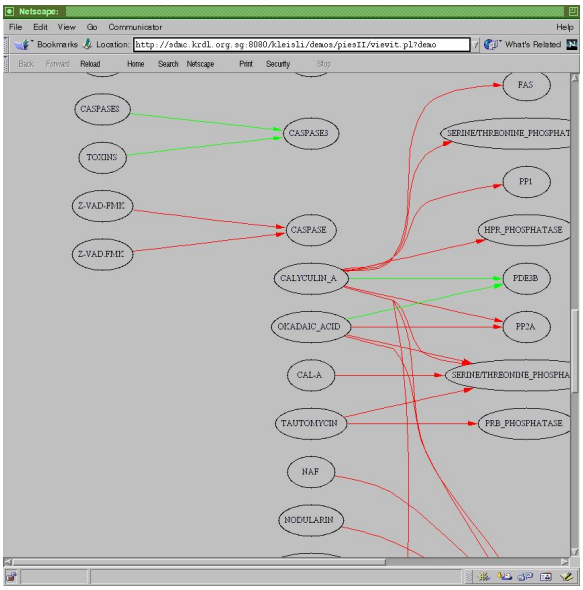


Figure 3: Example interaction graph produced by PIES.

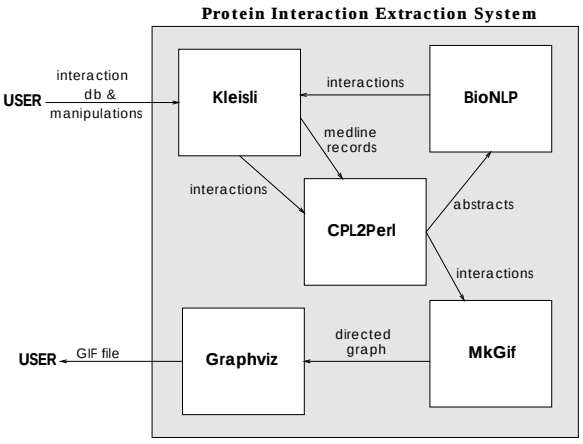


Figure 4: The architecture of PIES.

```
  | \x <- process (BioNLP, ["ARTICLES.val"], "") using syscall-co,
    \y <- x.#interactions }
to "INTERACTIONS" using stdout;
```

*In the first line, Kleisli accesses Entrez to obtain all Medline records corresponding to the search specification* `SPEC` *and writes the resulting set to the file* `ARTICLES`. *In the next few lines, Kleisli performs a call to BioNLP to process* `ARTICLES` *and writes the results to the file* `INTERACTIONS`. *BioNLP is discussed in a later section. The files* `ARTICLES` *and* `INTERACTIONS` *are used to derive a pathway diagram of these interactions extracted from the Medline abstracts. They constitute a logical representation (the interaction database) of the pathway diagram. This logical representation is the means through which we manipulate the pathway diagram.* □

### *3.2 Extracting Interaction Information*

We now briefly outline the implementation of BioNLP. Given an abstract, BioNLP breaks it up into sentences and deals with each sentence separately. It discards all sentences that do not mention function words such as "inhibit", "activate", and other equivalent words. Those sentences that are retained are assumed to mention protein interactions. BioNLP then tries to spot words that are likely to be names of proteins in these sentences using a set of lexical rules similar to those of Fukuda[5]. BioNLP also tries to normalize these names using a dictionary so that different names of the same protein are mapped to a standard name whenever possible. Once the proteins are identified, BioNLP tries to fit the sentence using a set of templates in order to determine which protein plays the role of the "actor" (or subject) and which protein plays the role of the "patient" ( or object) in the interaction mentioned in this sentence. The details are described in Ng and Wong[14].

It is worth stressing that BioNLP extracts the direction of interactions; that is, who inhibits whom and who activates whom. This level of information is in contrast to co-occurrence-based methods[18] that simply say two proteins interact but without giving the direction of the interaction.

### *3.3 Presenting Evidence*

As mentioned earlier, PIES produces two kinds of output: a textual display of the interactions extracted and their evidence, and a graphical display in the form of a GIF file or a Postscript file. The textual output is subject of this subsection.

Recall that in the file `INTERACTIONS`, each record stores one interaction and one evidence for that interaction. such a "flat" form is very convenient for

database storage and for many database-style operations to be described in a later subsection. However, it is not the natural way for presenting evidence for an interaction. A more natural way is to group all the evidence for one interaction together in one place and show them as a group. So PIES uses Kleisli to derive from the file `INTERACTIONS` an output file `OUTPUT` by grouping all evidence for each interaction in one place.

### *3.4 Drawing Pathway Diagram*

The GIF- or Postscript-formatted output of the interactions is a pathway diagram. The production of this diagram involves three modules of PIES: CPL2Perl, MkGIF, and Graphviz. The input to this step is the `OUTPUT` file, which is stored in Kleisli's data exchange format. The CPL2Perl module, which is a generic parser for the Kleisli Exchange Format, is used to convert this data into a structured Perl object. The MkGIF module, which is implemented in Perl, takes this Perl object and produces a directed graph specification file. Finally, the Graphviz module takes this directed graph specification and generates a layout for the pathway diagram in GIF and Postscript formats. Let us now proceed to the details.

The problem of "optimizing" the drawing of a pathway diagram, making it as visually attractive as possible, is a non-trivial one[9]. Nevertheless, it is possible to efficiently produce fairly neat pathway diagrams using a four-pass heuristic-based technique originally developed for directed graphs[7]. Very roughly, the technique works as follow. In the first pass, an optimal rank assignment for the nodes is found using a network simplex algorithm. If we think of the diagram as being divided into vertical columns in a left-to-right orientation, the rank assignment corresponds to the assignment of nodes to columns. In the second pass, the vertex order within each rank is set by an iterative heuristic for local transpositions to reduce crossings. If we think of the vertical columns as being divided into rows, the vertex order corresponds to the assignment of nodes in that column to these rows. In the third pass, optimal coordinates for nodes are found by constructing and ranking an auxiliary graph. In the last past, splines are made to draw the links. This algorithm makes neat drawings efficiently. It is implemented in the Graphviz module[6].

However, the Graphiz module was implemented as a general package for the automatic layout of directed graphs so that it can serve as a building blocks for many other applications. As a consequence, it accepts a general directed graph specification in a form that is very different from the `OUTPUT` file. In essence, it accepts a list of arcs of the form $x \to y$, which specifies an arc is to be drawn from the node $x$ to the node $y$. So we develop a simple package

MkGIF in Perl for converting `OUTPUT` to the directed graph format needed by Graphviz. The implementation of MkGIF is greatly simplified by the presence of a general parser for Kleisli's data exchange format, CPL2Perl, which converts file layout in Kleisli's data exchange format into structured native Perl object. An example pathway diagram produced is shown in Figure 3.

### *3.5 Manipulation*

Once the pathway and its logical representation (the file `INTERACTIONS`) have been produced, PIES allows the user to perform many operations. We describe several of these operations. They are all implemented typically with a CPL query script that is much less than 10 lines in length!

The simpler operations are those on nodes and edges. The operations available on nodes include: to rename a node, to delete a node, and to duplicate a node. The operations available on edges include: to delete an edge, to reverse the direction of an edge, to reverse the action of an edge (ie. turn the edge into "inhibit" or to "activate".) These are operations primarily for the purpose of correcting possible mistakes made by the BioNLP module in recognizing protein interaction.

The more sophisticated operations are those on entire pathway diagrams. The operations on the entire pathway diagram include: to reduce the diagram around the neighbourhood of several selected nodes, to expand the diagram by using additional keywords to download new Medline abstracts and extract interactions from them, to merge the current pathway diagram with a previously saved one, to save or export the current pathway diagram in a form readable by BioJAKE[17]. There is also a special operation to compare the current pathway diagram with a previous one and display the new interactions and new evidence. As these are considerably more interesting operations, we discuss them further delow.

The user or his colleagues may have another interaction database obtained for some other reasons or proteins. It is likely that he may some time want to merge the second interaction database with the one he is currently working on. The operation to automate the process of merging two interaction databases is most useful for this purpose.

The user upon exploring the interaction database, is likely to want to push further up or down stream from an interaction. He is also likely to be inspired to ask for other interaction information. The operation that allows him to specify additional keywords to download new Medline abstracts and to automatically integrate additional interaction information extracted from these abstracts into his interaction database is very useful in this situation.

The potential size of the interaction database makes the operation that reduces the diagram around the neighbourhood of several nodes useful. For example, the user can specify the names of a few proteins and a radius, then this operation can automatically zooms in on these proteins and extract a subdiagram consisting all of interactions within the specified radius of these proteins. Thus the intermediaries and neighbourhood of interactions involving his proteins can be readily picked out from a large interaction database. To implement this operation, we first derive an undirected graph from the pathway diagram. Then we compute a partial transitive closure of this graph beginning from the specified proteins and up to the specified radius. Then we select from the logical representation of the pathway diagram only those records where both the actor and patient are within this partial transitive closure.

The current interaction database of the user may be generated by expanding on a previous interaction database—either by the specification of additional keywords, the merger with another interaction database, or by re-runing the same keywords to get newer Medline abstracts. It is often useful to find out exactly which interactions are new and exactly which interactions have new evidence. The special operation to compare two interaction database and highlight their differences is useful for this purpose.

## 4 Remarks

PIES automates the task of creating and visualizing pathways on-the-fly, as well as supports sophisticated large-scale manipulations of pathways including automatic integration of interaction pathway databases. Everything described here is fully operational and web access can be arranged on a case-by-case basis with the author. We now mention some aspects that PIES can be improved upon.

PIES currently relies on natural language analysis of biomedical literature for extracting interaction information. Recently methods [11,4] based on detecting gene fusion events in sequence databases for predicting protein interactions have been proposed. These methods can be straightforwardly implemented using the Kleisli system. They are thus prime candidates for incorporation into PIES to endow it with the ability to automatically construct possible pathways from sequence databases.

PIES have good functionalities when it comes to manipulation of pathways. However, currently it does not support explicit annotations by the user on individual protein interaction. Such annotations are a useful addition to the evidence sentences extracted automatically by PIES. It would be useful for PIES to support this. Currently, this support is only an indirect one: PIES

exports its interaction pathway database in a format that the BioJAKE system can handle. User can then add his annotations and re-export the annotation database back to PIES.

The textual display of PIES does provide information on the context of the extracted information in the form of evidence sentences and links to the original Medline abstracts. However, the graphical display does not provide this information. Further research should be carried out on the graphical presentation of the context information in a visually appealing and explicit manner.

Finally, the BioNLP module currently specializes in extracting inhibit vs activate type of interactions. While its specificity on abstracts discussing these type of interaction appears high, a formal accuracy study remains to be done.

## Acknowledgements

We thank See-Kiong Ng for useful discussion and for the BioNLP module.

## References

1. D. Benton. Bioinformatics — principles and potential of a new multidisciplinary tool. *Trends in Biotechnology*, 14:261–272, 1996.
2. S.Y. Chung and L. Wong. Kleisli: a new tool for data integration in biology. *Trends in Biotechnology*, 17(9):351–355, 1999.
3. S. Davidson et. al. BioKleisli: A digital library for biomedical researchers. *International Journal of Digital Libraries*, 1(1):36–53, 1997.
4. A.J. Enright et. al. Protein interaction maps for complete genomes based on gene fusion events. *Nature*, 402:86–90, 1999.
5. K. Fukuda et. al. Toward information extraction: Identifying protein names from biological papers. *Pacific Symposium on Biocomputing*, 707–718, 1998.
6. E.R. Gansner and S.C. North. An open graph visualization system and its applications to software engineering. *Software—Practice and Experience*. To appear.
7. E.R. Gansner et al. A technique for drawing directed graphs. *IEEE Trans. Software Engineering*, 19(3):214–230, 1993.
8. P. Karp et al. EcoCyc: Encyclopedia of the escherichia coli genes and metabolism. *Nucleic Acids Research*, 27(1):55–58, 1999.
9. P. Karp and S. Paley. Automated drawing of metabolic pathways. In *Proceedings of 3rd International Conference on Bioinformatics and Genome Research*, 1994.

10. P. Karp. Database links are a foundation for interoperability. *Trends in Biotechnology*, 14:273–279, 1996.
11. E.M. Marcotte et. al. Detecting protein function and protein-protein interactions from genome sequences. *Science*, 285:751–753, 1999.
12. J. McEntyre. Linking up with Entrez. *TIG*, 14(1):39–40, 1998.
13. R. Milner, M. Tofte, R. Harper. *The Definition of Standard ML*. MIT Press, 1990.
14. S.K. Ng and M. Wong. Toward routine automatic pathway discovery from on-line scientific text abstracts. *Genome Informatics*, 10:104–112, 1999.
15. H. Ogata et al. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Res.*, 27(1):29–34, 1999.
16. T.C. Rindflesch et. al. EDGAR: Extraction of drugs, genes, and relations from biomedical literature. *Pacific Symposium on Biocomputing*, 517–528, 2000.
17. W. Salamonsen et. al. BioJAKE: A tool for the creation, visualization and manipulation of metabolic pathways. *Pacific Symposium on Biocomputing*, 392–400, 1999.
18. B.J. Stapley and G. Benoit. Biobibliometrics: Information retrieval and visualization from co-occurrences of gene names in medline asbtracts. *Pacific Symposium on Biocomputing*, 529–540, 2000.
19. M. Tomita et. al. E-CELL: Software environment for whole-cell simulation. *Bioinformatics*, 15(1):72–84, 1999.
20. L. Wong. Kleisli, a functional query system. *Journal of Functional Programming*, 10(1):19–56, 2000.