GenePath: A System for Inference of Genetic Networks and Proposal of Genetic Experiments

Blaž Zupan[1,6], Ivan Bratko[1,2], Janez Demšar[1], Peter Juvan[1], Tomaž Curk[1], Urban Borštnik[1], J. Robert Beck[3], John Halter[4], Adam Kuspa[5,6], Gad Shaulsky[6]

[1] Faculty of Computer and Information Science, University of Ljubljana, Trzaska 25, SI-1000 Ljubljana, Slovenia

[2] Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia

[3] Fox Chase Cancer Center, 7701 Burholme Avenue, Philadelphia, PA 19111

Departments of [4] PM&R, Molecular and Cellular Biology and Division of Neuroscience, [5] Biochemistry and Molecular Biology, and [6] Molecular and Human Genetics, all Baylor College of Medicine, 1 Baylor Plaza, Houston, TX 77030

Correspondence:

Dr. Blaž Zupan
Faculty of Computer and Information Science
University of Ljubljana
Tržaška 25
SI-1000 Ljubljana, Slovenia
phone: +386 1 4768 402
fax: +386 1 425 1038
e-mail: blaz.zupan@fri.uni-lj.si
www: magix.fri.uni-lj.si/blaz

Abstract

A genetic network is a formalism that is often used in biology to represent causalities and reason about biological phenomena related to genetic regulation. We present GenePath, a computer-based system that supports the inference of genetic networks from a set of genetic experiments. Implemented in Prolog, GenePath uses abductive inference to elucidate network constraints based on background knowledge and experimental results. Additionally, it can propose genetic experiments that may further refine the discovered network and establish relations between genes that could not be related based on the original experimental data. We illustrate GenePath's approach and utility on analysis of data on aggregation and sporulation of the soil amoeba *Dictyostelium discoideum*.

Keywords

genetic networks
functional genomics
abduction
bioinformatics
knowledge discovery
background knowledge

1 Introduction

Genes are basic units of genetic material. They are coded in double-stranded DNAs and reside within chromosomes of each living cell. Generally, each gene encodes a specific protein. In the process of protein synthesis a gene is first transcribed to a single-stranded RNA molecule (mRNA), which is then translated to a protein. Genes can excite or inhibit each other at various levels. The molecular mechanisms of excitation or inhibition can involve direct regulation of gene expression where the translation product (protein) of one gene directly regulated the transcription of another gene through protein-DNA interactions. They can also involve protein-RNA interactions that regulate mRNA translation, or protein-protein interactions that regulate protein activity, stability or localization. In this manuscript we refer to excitatory or inhibitory relationships between genes regardless of the molecular mechanism.

Biologists often use genetic networks to express and study relations between genes and the biological processes they regulate. Genetic networks are models that, in their often very simplified way, describe some biological phenomenon from the viewpoint of interactions between genes. They provide a high-level view and disregard most details on how exactly one gene regulates the activity of another. They include genes under study and some biological processes (in most cases only one) that is regulated by these genes. A genetic network is a graph whose nodes correspond to genes and biological processes, and arcs correspond to influences of genes on other genes and on biological processes. In this paper we consider the most commonly used and reported type of genetic networks, *qualitative genetic networks*. In qualitative genetic networks, the interaction between elements of the network is simplified to sole inhibition (negative influence) or excitation of activity (positive influence), while the magnitude of the influence or the particular regulation function are not shown.

Consider for example the genetic network shown in Figure 1. It includes four genes (*regA*, *pufA*, *pkaR* and *pkaC*) and a single biological process (*agg*, aggregation of soil amoeba *D. discoideum*). Arcs marked with $\dashv$ denote inhibition, and those with $\rightarrow$ denote excitation. Hypothetically, keeping all other activities constant, activity of *pkaR* should rise upon elevated activity of *regA*, or decrease under the reduction of activity of

regA. Conversely, and according to the network, activity of *pkaC* should decrease under increase of activity of *pkaR* (or *pufA*). Notice that the qualitative network does not denote to what extent and exactly how the activity of one gene should change to notice the change of activity of other genes. It depicts only the type of relation. Perhaps even more importantly, genetic networks encode the genetic pathways, *e.g.*, chains (paths) of gene regulation. For instance, from Figure 1 it is clear that gene *pkaC* is regulated by gene *regA*, but the regulation is not direct as it goes through another gene (*pkaR*). On the other hand, the genes *pufA* and *pkaR* directly influence the activity of *pkaC*.

The notion of "directness" is based on the closed-world assumption: genetic networks consider only a (small) subset of the cell's genes and ignore all other genes. For instance, the network from Figure 1 depicts only relations between *regA*, *pufA*, *pkaR*, *pkaC* and aggregation. Therefore, the arrow between *regA* and *pufA* only states that none of the other considered genes mediates between *regA* and *pufA*. There may be other genes in the cell – ones that are not included in the study and the network – on the path from *regA* to *pufA*.

[Figure 1 about here.]

A geneticist's main tool to investigate biological phenomena and build corresponding genetic networks is inducing mutations. There are two most often used types of mutations. In gene inactivation, geneticists remove a gene from the DNA (this is also called a "knock-out") or use some other means to prevent it from being expressed. An opposite mutation that is often harder to induce in the laboratory, is activation of a gene (also referred to as *overexpression*), that results in a gene being significantly more active than under normal conditions. Although an organism can have more than one gene mutated, geneticists mostly use only single and double mutants since inducing a greater number of mutations can be technically difficult. Geneticists then infer genetic networks by comparing the resulting biological processes on different strains of mutated organisms, or by comparing mutant organisms to organisms without any artificially induced mutations (so-called *wild-type* organisms).

For instance, upon starvation, the soil amoeba *Dictyostelium discoideum* in aggregates

and forms a multi-cellular organism. But a mutant with a knocked-out gene *pkaC* does not aggregate; this provides the evidence that *pkaC* essential for the aggregation process and that its effect to aggregation is positive (excitatory). Conversely, a mutant with a knocked-out *pufA* aggregates excessively, also indicating the involvement of this gene in the aggregation process, but this time with a negative (inhibitory) effect. Determining the order of genes in the network and discovering the potential parallelism of effect of genes requires at least a triplet of experiments where at least one should have more than a single mutated gene. The logic and inference patterns behind this are explained later in the paper.

At this point we should note that at present biological experiments with mutants are time consuming. It usually takes several weeks from preparing a mutant organism to obtaining experimental results in wet labs. For this reason, the largest of genetic networks reported to-date include no more than several tens of genes, and collecting experiments that provide the evidence often takes several man-years of effort. Biologists therefore reason about a limited number of genes at a time, and usually construct their networks from no more than a few tens of experiments. But even for these, the analysis of the experimental data is not trivial. The combinatorial explosion of number of possible relations and gene orderings calls for a tool that would assist in construction of genetic networks and that would incorporate the basic principles from genetic data analysis.

In this paper, we report on GenePath (`http://genepath.org`), a program that may be regarded as an intelligent assistant in genetic data analysis. It constructs genetic networks from gathered experimental results and the already known relations between genes and processes (*background knowledge*). Besides that, it can propose new experiments to refine the discovered networks or gather additional proofs for its relations. Its primary source of data are genetic experiments, in which genes are either knocked out or activated, and the behavior of the system – qualitative descriptions of biological processes – under such mutations. To construct a genetic network, GenePath uses a set of expert-defined inference patterns in the form "IF a certain combination of experiments is found in the data, THEN a certain relationship between genes and a biological processes is concluded". When applied to the data, these inference pat-

terns identify constraints for the corresponding genetic network. GenePath then uses these constraints to construct a genetic network that explains the data. The biological motivation of GenePath and details on patterns it uses for abduction were first described in [23]; the current paper focuses on methodological issues, and extends original GenePath framework with qualitative reasoning and methods to propose alternative networks and additional experiments.

To propose genetic networks, GenePath performs *abductive* reasoning, as opposed to the more common *deductive* reasoning. Deductive reasoning starts with some given logical formula A and derives a new formula B such that B logically follows from A. That is, if A is true, then B must be true. In a sense, deductive reasoning does not produce any new information: all the information contained in the result B is already implied by the information given in A. On the other hand, abductive reasoning produces results that do not necessarily follow from the given information, but are in some respect relevant to it. Thus, abduction can result in a new, useful information, but that information is not necessarily true. The most common application of abductive reasoning is generation of explanation for observations. We have some background knowledge BK and some experimental observations E. The observations E do not logically follow from the existing knowledge BK, so we say that BK does not (entirely) explain the observations (although the two are not in contradiction). We are looking for a hypothesis H, so that BK together with H imply E (formally, $BK \cup H \models E$). That is, H (in the context of BK) explains the given experiments. In GenePath, E represents the genetic experiments, BK is prior biological knowledge, and H is the desired genetic network.

The logic behind GenePath's algorithm is the same as used by geneticists world-wide when they manually construct network models (see, for example, [2]). A major contribution of our work is in the formalisation and partial automation of geneticists' reasoning. Thus, a geneticist can follow the reasoning performed by GenePath and clearly understand the justification, in terms of experimental data, for including particular parts of the constructed network. The approach was successfully tested on several real-life applications. Two such applications – aggregation and sporulation of the soil amoeba *Dictyostelium discoideum* – are reported in this paper. Another original contribution of GenePath is originated from its particular implementation and

interface: GenePath allows the user to examine the experimental evidence and the logic that were used to determine each particular relationship between genes. This allows a geneticist to trace back every finding (relation) to the set of original experiments that provided the evidence for it. Furthermore, GenePath can identify which couples of genes could not be related based on experimental data. It can then propose new experiments that would make it possible to relate each of these gene couples, and propose, depending on the outcome of such experiments, revisions of the genetic network. As such, GenePath is not intended to replace the researcher, but rather to support the processes of cataloging and interpreting genetic experiments, the derivation of genetic networks, and proposal of new experiments.

We start our description of GenePath with an example of genetic data on aggregation of *Dictyostelium discoideum* that is used throughout the paper to illustrate the utility of developed techniques. A general description of GenePath's framework is given next, followed by a description of particular mechanisms for abduction of relations, construction of genetic networks and proposal of genetic experiments. Finally, we mention another successful case of GenePath's use (sporulation of *Dictyostelium*), provide some discussion of experimental results, utility and computational efficiency of GenePath, and conclude with some ideas for potential extensions of developed methodology.

It should be noted that the examples of *Dictyostelium* aggregation and sporulation in this paper are not useful only as illustrations of GenePath's mechanisms, but they are of realistic complexity with respect to genetic research and both represent relevant and currently investigated genetic problems. The complexity of these examples and the resulting genetic networks are indicative of the complexity of genetic network theories in current research in functional genomics based on mutation experiments. At the end of the paper we also discuss the scalability of GenePath to much larger datasets with, say, a thousand genes. However, for reasons mentioned earlier, there is relatively little practical need, in mutation-based research, for scalability beyond a few tens of genes which is easily accommodated within GenePath's present capability.

2 Data and Background Knowledge

Experiments that GenePath can consider are represented as tuples of mutations and outcomes. A mutation is specified as a set of one or more genes with information on the type of mutation. A gene can be either knocked-out or overexpressed. The outcome of an experiment is a geneticist's description of biological processes, also termed as a phenotype (e.g., "cells grow", "cells do not grow", etc). It should be noted that the abstract representation used in GenePath is precisely the representation used by the geneticists when manually constructing genetic theories.

A data set that will be used to illustrate some of GenePath's functionality is given in Table 1. The data comes from the studies of *Dictyostelium discoideum*, an organism that has a particularly interesting developmental cycle from single independent cells to a multicellular slug-like form (Figure 2). The transition from ordinary single cells to a multicellular structure is of great interest for biologists studying the evolution and development of multicellular organisms [22, 17, 16]. This phenomenon is still a topic of ongoing research.

[Table 1 about here.]

The example data set focuses on cell aggregation. Observed aggregation was either normal ("+"), increased ("±"), excessive ("++"), or absent ("-"). The first row of Table 1 corresponds to the wild type (experiment with organism with no mutation). The remaining rows correspond to 14 mutation experiments. Mutations were performed on six genes: *yakA*, *pufA*, *pkaR*, *pkaC*, *acaA*, and *regA*. In each experiment (except for the wild type) selected genes were either knocked-out or overexpressed (denoted with "-" or "+", respectively). Note that the aggregation level is an ordinal variable, with the order being -, +, ±, ++.

[Figure 2 about here.]

Background knowledge may be given to GenePath in the form of known relations between genes. For our example, GenePath was informed that it is already known that

acaA inhibits *pkaR* and that *pkaR* inhibits *pkaC*. Therefore, the following two relations form our background knowledge:

$$acaA \dashv pkaR$$
$$pkaR \dashv pkaC$$

3 Overview of GenePath

GenePath implements a framework for reasoning about genetic experiments and hypothesizing genetic networks. This framework, illustrated in Figure 3, consists of the following entities:

- *genetic data*, *i.e.*, experiments with mutations and corresponding outcomes,
- *background knowledge* in the form of known relations between genes and biological processes,
- *expert-defined reasoning patterns*, used by GenePath to abduce gene relations from genetic experiments,
- *abductive inference engine* that matches the encoded patterns with genetic data to obtain constraints over the genetic network,
- *network synthesis* that constructs a network (hypothesis) that is consistent with genetic data, abduced constraints and background knowledge, and
- *engine for proposal of genetic experiments* that, given existing genetic data and abduced constrains, proposes sets of additional experiments that would refine the derived network and provide evidence for relations that could not be established from the existing data set.

[Figure 3 about here.]

Overall, Genepath works as follows. It takes experimental data (such as those in Table 1) and abduces relations between genes and outcomes (step (a) in Figure 3). In this step, GenePath uses abductive inference patterns (described in Section 4) designed to closely mimic the reasoning of expert geneticists. Discovered relations and relations from background knowledge form a set of constraints that a derived network must satisfy. Genepath finds and shows one such network (b). It can then propose sets of new genetic experiments that would provide additional information needed to refine the network (c). The user may review the proposed sets of experiments and decide which ones to perform in the laboratory (d). New experimental results are then added to the experimental data, and the cycle may be repeated from step (a).

4 Expert-Defined Abduction Patterns

GenePath's inference patterns define how various relations between genes may be abduced from the data. These inference patterns are implemented as clauses in Prolog programming language and are used to determine the relation between two genes or the relation between a gene and a biological process. Every relation found from data is accompanied by the evidence in the form of the name of the inference pattern that was used to find the relation and the experiments that support it. Where the pattern includes gene's influence, this can be either "excites" or "inhibits" and is determined directly from the data. GenePath is currently capable of abducing the following types of relations among genes and biological processes:

1. *influences*: `GeneA` influences `GeneB` or biological process `BP`. The influence can be either excitation, written as `GeneA` $\rightarrow$ `GeneB`, or inhibition, written as `GeneA` $\dashv$ `GeneB`.

2. *not_influences*: `GeneA` does not influence a biological process `BP`.

3. *parallel*: both `GeneA` and `GeneB` influence a biological process `BP`, but are on separate (parallel) paths in the genetic network, *i.e.*, neither of the two genes influences the other. This is written as `GeneA` || `GeneB`.

4. *epistatic*: `GeneA` precedes `GeneB` in a genetic network (both genes are therefore on the same path, but `GeneB` is closer to the node that represents a biological process); we also say that `GeneB` is epistatic to `GeneA`, or also that `GeneB` is on the same path both to the biological process `BP` but downstram from `GeneA` .

One or more inference patterns exist for inferring each type of relation. For instance, there are (at the time of writing of this paper) two patterns for epistasis and one for parallelism. GenePath's patterns are in detail described in [23]; to give an example and illustrate how they are used on the data, we will bellow two distinct patterns, one for parallelism and one for epistasis.

It should be emphasized that the inference patterns have been designed with the aim of closely mimicking geneticists' reasoning when they manually construct genetic networks. They have strong empirical verification, but they have not been formally proven sound and/or complete. Therefore we view these patterns as rules for abductive inference.

4.1 Inference Pattern `parDiff` for Parallel Relation

There is currently a sole pattern for inferring parallel relation, called `parDiff`. This pattern determines the thruth of the following predicate:

```
parallel(GeneA, GeneB, parDiff, Exp1, Exp2, Exp3,BP)
```

The predicate states that genes `GeneA` nad `GeneB` are on parallel paths to biological process `BP`. Experiments `Exp1`, `Exp2` and `Exp3` provide evidence for this relation between `GeneA` and `GeneB`. The inference rule for this predicate is: *let us have three experiments: single mutation of* `GeneA` *and phenotype* `PA` *(*`Exp1`*), single mutation of* `GeneB` *and phenotype* `PB` *(*`Exp2`*), and double mutation of both* `GeneA` *and* `GeneB` *and phenotype* `PAB` *(*`Exp3`*). Phenotypes* `PA`*,* `PAB`*, and* `PB` *are all descriptions of the same biological process* `BP`*. If* `PA` *and* `PB` *are both different from the wild type, and* $\texttt{PA} \neq \texttt{PAB}$ *and* $\texttt{PB} \neq \texttt{PAB}$ *then* `GeneA` *and* `GeneB` *are on parallel paths to biological process* `BP`*.*

In other words and less formally, pattern `parDiff` says that two genes are in parallel pathways if mutations in either gene have an effect on the biological process and the phenotype of the double mutant is different from either mutation alone.

For instance, using the pattern `parDiff`, genes *yakA* and *pkaR* are found to act in parallel pathways because the outcomes of the single gene mutations in experiments 2 and 4 are different from each other and from the outcome of the double gene mutation in experiment 12.

GenePath has thus found that *yakA* should be on a different path to aggregation than *pkaR*, or, formally *yakA* || *pkaR*. Notice also that the evidence for the above is stated in terms of corresponding trio of experiments (experiments 2, 4 and 12 from Table 1) that matched the `parDiff` pattern.

4.2 Inference Pattern `epMut` for Epistatic Relation

One of the most important patterns for the construction of genetic networks in GenePath is `epMut`, which is one of the two patterns for epistasis. Epistatic relations between genes are represented by the predicate:

```
epistatic(GeneA, GeneB, Influence, epMut, Exp1, Exp2, Exp3, BP)
```

This says that `GeneB` is epistatic to `GeneA`, that is, `GeneB` follows `GeneA` on one path to the biological process `BP` in a genetic network. The argument `Influence` gives the type of influence of `GeneA` or `GeneB` (either excitatory or inhibitory). Experiments `Exp1`, `Exp2` and `Exp3` provide evidence for this relation.

The corresponding inference rule for this pattern is: *if two different mutations (experiments* `Exp1` *and* `Exp2` *with mutations of* `GeneA` *and* `GeneB`*, respectively) result in two different phenotypes of biological process* `BP` *and the phenotype of the double gene mutant (experiment* `Exp3` *where both* `GeneA` *and* `GeneB` *are mutated) is the same as one of the single gene mutant with mutation in* `GeneB`*, then (*`GeneB`*) is epistatic and is considered to act after* `GeneA`*. In other words,* `GeneA` *precedes* `GeneB` *for the biological process* `BP`*.*

Three sets of experiments matching `epMut` were found in our sample data in Table 1, identifying three important gene-to-gene relations:

```
epistatic(regA, pkaC, inhibits, epMut, 7, 5, 10, aggregation)
epistatic(yakA, pkaC, excites, epMut, 2, 9, 15, aggregation)
epistatic(yakA, pufA, inhibits, epMut, 2, 3, 11, aggregation)
```

Note that for every relation like those above, GenePath always gives the name of the inference pattern and the experiments that were involved and can be used as an evidence for the relation hypothesized.

5 Synthesis of Genetic Networks

GenePath constructs a genetic network by considering all the relations it found as constraints over the possible networks and attempting to find a network that would satisfy all the constraints.

The mutual consistency of abduced network constraints is tested first. If conflicts are found – such as, for instance, a gene reported not to influence the biological process and at the same type involved in some epistasis relation for the same process – they are shown to the domain expert and resolved either by removal of one of the constraints in conflict, or through proper revision of the data (if the error has been found there). For our example on aggregation of *Dictyostelium*, no conflicts in abduced constraints were detected.

The construction of the genetic network is based on identification of *directly* related genes. Two genes are assumed to be in direct relation if they have been found to be in epistatic relation and if no other gene is found that precedes one of them and is at the same time preceded by the other.

For instance, the following epistatic relations were found in the data (*i.e.*, from background knowledge and Table 1): $acaA \rightarrow pkaC$, $acaA \dashv pkaR$, $pkaR \dashv pkaC$. Could $acaA$ and $pkaC$ be directly related? They are candidates for this because $acaA \rightarrow$

pkaC, but *pkaR* is epistatic to *acaA* (*acaA* $\dashv$ *pkaR*) and inhibits *pkaC* (*pkaR* $\dashv$ *pkaC*), so *acaA* and *pkaC* cannot be directly related. On the other hand, *acaA* and *pkaR* are directly related because *pkaR* is epistatic to *acaA* and there is no evidence for intervening genes. Similarly, we find that *pkaR* and *pkaC* are directly related, so a fragment of the network we have just logically inferred is *acaA* $\dashv$ *pkaR* $\dashv$ *pkaC*.

After finding directly related genes, GenePath constructs the hypothesized genetic network. It places each gene as a node in a graph, drawing a corresponding edge between nodes that represent directly related genes. Genes that influence the outcome but are not followed by any other gene in the network are directly related to the biological process (a special node in the network) with an edge that shows their influence. In our example, *pkaC* and *pufA* are the only such genes and are connected to aggregation with an edge for excitation and inhibition, respectively. The final genetic network inferred by GenePath is as presented in Figure 4.

[Figure 4 about here.]

The constructed genetic network was examined by participating expert (GS). He immediately realized that, while the network was in general consistent with his domain knowledge, the direct influence of *pufA* on aggregation was not expected. Reviewing the relations that GenePath found showed that the reason for that was an undetermined relation between *pufA* and *pkaC*, and this was due to lack of appropriate experimental data.

Notice that for a specific set of network constraints, GenePath constructs a single network that is consistent with all the constraints. The network does not contain cycles and includes a minimal number of edges with respect to the constraints it has to satisfy. It follows the requirement that for any pair of directly related genes there *has* to be an epistatic relation that relates these genes, and hence any edge (relation) in GenePath's networks can be traced-back to its "cause" in the data. As edges between genes in GenePath's networks represent only direct relations, the process of network construction in GenePath (beyond determination of direct relations) is straightforward and deterministic.

An implicit bias that GenePath uses is that of Occam's razor: GenePath would construct a network consistent with all abduced constraints but with a minimal number of edges, *e.g.*, GenePath constructs the simplest hypothesis. Let us write this as a theorem and prove it:

Theorem. Over possible networks that would explain all the relations found in the genetic data, GenePath networks are minimal with respect to the number of edges.

Proof. Each direct relation between two genes is represented with a sole edge in GenePath's network. Removing such an edge would cause inconsistency with the data, since the network would not be able to explain the effect that an upstream gene has on the epistatic one. All genes that are not directly related to any other gene but do influence the biological process are represented in GenePath's network with a single edge between each such gene and biological process. Any other inclusion of these genes in the network would require at least one edge, hence the number of edges can not be decreased in this way. □

Let us illustrate the above concepts through a simple example. Suppose that, examining the data that included genes A, B and C and a biological process BP, GenePath abduced that B is epistatic to A, and that all genes influence the biological process BP in a positive way (excitation). There are five networks (Figure 5) that are consistent with these constraints and include a minimal number of edges: gene B should be epistatic to A, but the position of gene C can vary. Yet, only the network from Figure 5.e includes relations that can be proven by the data. For instance, for the network from Figure 5.a there is no evidence in the data that gene C is epistatic to either A or B. Similarly, for the network from Figure 5.d there is no evidence for epistasis of B over C. For this example, GenePath determines that genes A and B are in direct relation, and because there are no epistatic relation between gene C and any other gene, places gene C in a parallel path.

[Figure 5 about here.]

6 Proposal of Genetic Experiments

The nature of experimental work in genetics and biology is most often incremental. Experiments are usually costly, and after performing the initial ones, these are analyzed and decisions are taken about which experiments to perform next. To support such scientific discovery process, a particular module in GenePath was implemented that can propose additional experiments that would (potentially) refine the network and resolve ambiguities.

After analyzing the initial data set and developing a genetic network that can explain the experimental results, the following set of questions may be raised:

1. Which are the genes for which no relations have been determined from the data?
2. Which additional experiments that can be performed to establish the unknown relations?
3. Based on the possible results of new experiments, what would be the corresponding revised genetic network?

Notice that, in principle, GenePath constructs genetic networks from epistatic and parallel relations. Both types of relations relate a pair of genes. For some pairs of genes such relations cannot be determined from the experimental data using GenePath's inference patterns, nor are they given as backgound knowledge. For instance, for *Dictyostelium*'s aggregation (Table 1) there is a number of such gene pairs, including *yakA*-*acaA*, *yakA*-*regA*, *pufA*-*pkaR*, *pufA*-*pkaC*, *pufA*-*acaA*, *pufA*-*regA*, and *acaA*-*regA*.

Given two genes for which a relation could not be determined from the data, GenePath can examine the patterns for epistasis and parallelism and check if some pattern would match the data, provided that an additional experiment(s) would be performed. Notice that the existing expert-defined patterns determine the relations between two genes given a set of two or three experiments. When looking for new relations and proposing experiments, we request that some experiments required by a pattern are already in

the data base; other experiments required for the pattern are thus hypothesized and form a set of candidate (proposed) experiments for that specific relation.

For example, for *Dictyostelium* aggregation, there is insufficient data (Table 1) to establish the relation between *regA* and *yakA*. However, notice that for the epistasis pattern `epMut`, the first two experiments that match this pattern already exist: *yakA*- with no aggregation (experiment 2) and *regA*- with excessive aggregation (experiment 7). Following the definition of the pattern, if we perform a third experiment that includes both mutations (*regA*- and *yakA*-) and if its outcome is excessive aggregation then we can conclude that `regA` is epistatic to `yakA`. However, if the phenotype of this experiment would be no aggregation, then we could conclude that `yakA` is epistatic to `regA`.

In general, for each couple of genes, a specific relation may be supported by different sets of experiments, of which some may not be present in the data. These missing experiments are candidates for new experiments. As genetic experiments are in principle costly (if nothing else, it may take a couple of weeks to perform them), proposals for relations that require fewest new experiments are preferred. Furthermore, we favor those new experiments that require fewer genetic mutations (it is usually easier to obtain single then double or triple mutant), and those that contain inactivation mutations as opposed to activation mutations (knocking-out a gene is easier than overexpressing it).

Following the above heuristics, we have constructed an *ad-hoc* cost function elicited from geneticists, that estimates the complexity of proposed experiments. The cost function is based on additive penalties. For instance, a penalty of 20 is assigned for each experiment, 30 for each double and 90 for each triple mutant, 15 for each gain-of-function mutation, and -20 if epistasis and -25 if parallel relation can be concluded from the experiments. To keep all overall penalties above 0, the base (default) penalty is 45. Notice that while this function is purely heuristic and is not theoretically founded, in all observed cases they ranked the experiments in agreement with the geneticists.

Using this methodology, GenePath can find couples of (yet) unrelated genes, construct all sets of experiments that would relate genes in each couple, and present only experiment sets with the lowest complexity scores (penalties). However, a different, more

focused approach proved to be more practical. When a geneticist is presented a genetic network and a list of unrelated genes, it seems quite straightforward for him to pick an interesting couple of unrelated genes and use GenePath to try to establish some relation and present the result of such analysis – proposal of genetic experiments – starting with the simplest proposals. This process can be repeated for different couples of unrelated genes, possibly each time picking a proposed experiment, adding it to the database and observing a new resulting network.

For illustration, consider again the data on *Dictyostelium* aggregation (Table 1). An interesting and unrelated gene couple picked by participating geneticists was *pufA* and *pkaC*. The simplest experiment proposed by GenePath was:

```
New experiment a: pkaC-, pufA-
Cost 30
Case aggregation of
       [++] then pkaC -| pufA (epMut, 5, 3, a)
        [-] then pufA -| pkaC (epMut, 3, 5, a)
  [+], [+-] then pufA || pkaC (parDiff, 3, 5, a)
```

Here, the first line describes an experiment; it is labeled `a` and involves knock-out mutations of *pkaC* and *pufA*. The associated penalty (second line) of the proposed experiment is 30. The type of relation between these two genes depends on the outcome of the proposed experiment (aggregation). If the resulting aggregation is excessive then *pkaC* inhibits *pufA*, if the resulting mutant shows no ability to aggregate then *pufA* inhibits *pkaC*, and for the other two outcomes the two genes act in parallel. Notice that GenePath shows the support for each relation by stating which pattern and experiments (together with the new experiment `a`) would be used for reaching the conclusion.

The proposal was presented to the participating geneticist. His comment was: *An outcome with no aggregation is exactly what I would expect. The experiment was never done because we obtained biochemical data that showed directly that the protein encoded by* pufA *binds mRNA of* pkaC *and prevents translation into protein.*

Another proposal for the same gene couple, ranked next in the list of proposals, was:

```
New experiment a: pkaC+, pufA+
New experiment b: pufA+
Cost 95
Case aggregation of
   [-,-], [+,+], [+-,+-]   then pkaC -| pufA (epMut, 9, b, a)
   [++,-], [++,+], [++,+-] then pufA -| pkaC (epMut, b, 9, a)
   [-,+], [-,+-], [+,-],
   [+,+-], [+-,-], [+-,+]  then pufA || pkaC (parDiff, b, 9, a)
```

This proposal assumes two new experiments (experiment `a` with a double and `b` with a single mutant). Based on their outcome, different relations are concluded again. For instance, if neither of the mutants can aggregate (denoted with `[-,-]`), the relation *pkaC* inhibits *pufA* is concluded. However, if the double mutant does not aggregate but the single mutant does (`[-,+]`), then the two genes act in parallel. Notice further that, compared to the previous proposal, this one scores worse in terms of complexity, as it includes two experiments that also rely on gain-of-function mutants. Geneticist's comment on this proposal confirmed the utility of our complexity heuristics: *This experimental proposal is too complicated. Usually, activation of a gene (especially by overexpression) is not as reliable as inactivating it. Activating two genes in one strain is therefore undesirable.*

Another couple of genes that participating geneticists considered interesting to find a relation for were *regA* and *pkaR*. Two simplest proposals for their relation by GenePath were:

```
New experiment a: pkaR+
New experiment b: pkaR+, regA-
Cost 80
Case aggregation of
   [-,++], [+,++], [+-,++] then pkaR -> regA (epMut, a, 7, b)
   [-,-], [+,+], [+-,+-]   then regA -> pkaR (epMut, 7, a, b)
   [-,+], [-,+-], [+,-],
   [+,+-], [+-,-], [+-,+]  then pkaR || regA (parDiff, a, 7, b)
```

```
New experiment a: pkaR-, regA+
New experiment b: regA+
Cost 80
Case aggregation of
   [-,++], [+,++], [+-,++] then regA -> pkaR (epMut, b, 4, a)
   [-,-], [+,+], [+-,+-]   then pkaR -> regA (epMut, 4, b, a)
   [-,+], [-,+-], [+,-],
   [+,+-], [+-,-], [+-,+]  then pkaR || regA (parDiff, 4, b, a)
```

Following the first suggestion, the two experiments were indeed performed in laboratories at Baylor College of Medicine, and the result for both mutants (i.e., for mutant with *pkaR* + and mutant with *pkaR* + *regA*-) was reduced aggregation. Hence, we can conclude that *regA* is epistatic to *pkaR* (*regA* $\rightarrow$ *pkaR*).

Overall, there are therefore three experiments that we can add to our experiment base for aggregation and that help to relate *pkaC* and *pufA* on one hand and *pkaR* and *regA* on the other. With these, and with new relations (*pufA* $\dashv$ *pkaC*, *regA* $\rightarrow$ *pkaR*) the new resulting genetic network is as shown in Figure 6. This network is in complete accordance with current genetic hypotheses over the regulation of aggregation in *Dictyostelium* [17, 16, 10, 13].

[Figure 6 about here.]

7 Another Case Study: Sporulation of *Dictyostelium*

For another case study that we report in this paper, GenePath was used to infer a genetic network that controls sporulation in *Dictyostelium*. Sporulation is a phase that follows aggregation, where *Dictyostelium* amoeba specialize to either those forming stalks or spores (Figure 2). The data (Table 2) include 19 experiments involving 6 genes and the biological process (sporulation) was graded from absence of sporulation through slow and normal (wild-type) sporulation to rapid sporulation. A single epistatic relation was given as a prior knowledge: *pkaR* $\dashv$ *pkaC*.

[Table 2 about here.]

[Table 3 about here.]

GenePath found that all genes used in Table 2 influence the biological process of sporulation: while *pkaR* and *regA* inhibit it, all other genes excite sporulation. No experimental support was found for parallelism. A number of epistatic relations were found (Table 3). A genetic network constructed from these constraints is given in Figure 7.a. It was also observed (by GenePath, and by expert's overview of network and constraints) that relations between *dhkA* and *tagC*, and *tagB* and *tagC* could not be determined from the data. Of these two, the relation between *dhkA* and *tagC* was explored, and GenePath proposed two relatively simple sets of experiments to relate them:

```
New experiment a: dhkA-, tagC-
Cost 30
Case sporulation of
                [no] then dhkA -| tagC (epMut, a, 2, 14)
              [slow] then tagC -| dhkA (epMut, a, 14, 2)
   [normal], [rapid] then dhkA || tagC (parDiff, 2, 14, a)


New experiment a: dhkA+, tagC-
Cost 45
Case sporulation of
                [no] then dhkA -> tagC (epMut, a, 19, 14)
            [normal] then tagC -> dhkA (epMut, a, 14, 19)
     [slow], [rapid] then dhkA || tagC (parDiff, 19, 14, a)
```

The first of the two experiments proposed has a better score than the second, since both mutations in the experiment in the first set are knock-outs, while an overexpression mutation is present in the second experiment. Notice that possible conclusions from the first suggestion are different from the second (the first includes only inhibitory, the second only excitatory epistatic relations). When reviewed by the geneticist, he

preferred the second experiment and predicted a normal phenotype, thus establishing the relation $tagC \rightarrow dhkA$. This experiment was not yet performed, but its expected results are in accordance with previous biological findings [14]. Adding this experiment to the data resulted in genetic network as shown in Figure 7.b. This network is consistent with the present biological knowledge on regulatory mechanisms for sporulation of *Dictyostelium*.

[Figure 7 about here.]

8 Implementation and Web-Based Interface

GenePath is implemented in Prolog, and its substantial part is available through web-based interface (`http://genepath.org`).

8.1 Implementation in Prolog

GenePath's core is implemented in Prolog, a declarative computer language often associated with development of Artificial Intelligence-based applications [3]. Programming in Prolog is based on stating logical statements about the domain, where the data and the program share the same syntax. For instance, our aggregation data set (Table 1) is encoded as a set of Prolog clauses, like:

```
genes([yakA, pufA, pkaR, pkaC, acaA, regA]).
phenotypes([aggregation]).
outcome_values(aggregation, ['-', '+', '++']).
exp(1,  [],      aggregation, '+').
exp(2,  [yakA:-], aggregation, '-').
exp(3,  [pufA:-], aggregation, '++').
exp(4,  [pkaR:-], aggregation, '++').
exp(5,  [pkaC:-], aggregation, '-').
exp(6,  [acaA:-], aggregation, '-').
...
exp(15, [yakA:-, pkaC:+], aggregation, '++').
```

Notice that our sample data set includes a single biological process – in general, the number of biological processes used in GenePath is not limited. The above Prolog code lists gene names used in the problem, states the name of the biological process and the possible phenotypes (outcomes) associated with it, and gives the experiments. When describing experiments, the colon between the name of the gene and the type of mutation was added to simplify the rest of the Prolog code; the knock-out of gene *pufA*, for instance, is not described with the usual notation `pufA-`, but with `pufA:-`. In several places in the code above, Prolog's notation for lists is used. In this notation, the members of a list are enumerated and enclosed in square brackets. For example, the list of items `a, b, c` is written as `[a,b,c]`.

Prolog was chosen as a GenePath's programming language for its straightforwardness in defining and reasoning with inference patterns that are used to identify gene-to-gene and gene-to-outcome relations. For instance, consider the pattern `parDiff` ("two genes are in parallel paths IF mutations in either gene have an effect on the biological process AND the phenotype of the double mutant is different from either mutation alone") can be expressed in Prolog as (text following "%" is a comment, and not part of the code):

```
parallel( GeneA, GeneB, Evidence, BP)  :-    % GeneA and GeneB are on parallel paths if:
  exp(Exp1,[GeneA:MutA], BP, Out1),          % There is single mut. experiment with GeneA
  exp(Exp2,[GeneB:MutB], BP, Out2),          % There is single mut. experiment with GeneB
  exp(Exp3,[GeneA:MutA,GeneB:MutB],BP,Out3), % There is corresponding double mutant
  Out1 \== Out2,                             % The outcomes of Exp1 and Exp2 are different
  Out3 \== Out1, Out3 \== Out2,              % The outcome of the double mutant is also different
  Evidence = (parDiff, Exp1, Exp2, Exp3).    % The evidence comprises the three experiments
```

Assume that the data of Table 1 has been loaded into Prolog. An example of a query that uses the above definition and asks for genes that may be on parallel paths to aggregation is:

```
?- parallel(A, B, Evidence, aggregation).
```

One of Prolog's responses would be:

```
A = pkaC,
B = regA,
Evidence = (parDiff, 10, 5, 7)
```

GenePath's code to find all possible relations between genes and outcomes, and to derive the corresponding plausible genetic network is more technical and complex, and follows general ideas about graph theory programming in Prolog [3]. Nevertheless, GenePath's Prolog implementation currently consists of no more than five hundred lines of code, and its relative simplicity is helpful in testing and prototyping new ideas and patterns.

8.2 Web-Based Interface

While the Prolog programming language proved an excellent environment for prototyping and testing both GenePath and its qualitative reasoning part GeneHyp, attempts to convince (even a participating) geneticists to use the program in its native environment were futile (as expected). Since our goal is to develop a broadly accessible and easy-to-use intelligent assistant for explorative analysis of genetic data, we consequently designed a web-based interface, where Prolog's core resides on the server, and is accessed through a server's Visual Basic interface to provide a pure HTML-based solution for clients. Currently, a web-based interface is available only for abductive inference part, and can be freely accessed at `http://genepath.org`.

A web-based interface is comprised of several screens that allow a geneticist to state genes and outcomes, define the background knowledge and list genetic experiments. The data can be saved to a local file on a client, making it available for future uploads and changes. GenePath uses a single window for result analysis (Figure 8), where a hypothesized network is displayed together with a choice of abduced constraints. Importantly, the user can request, by a mouse click, evidence for a specific constraint, obtaining a window that describes the reasoning behind this constraint in plain English (Figure 9).

[Figure 8 about here.]

[Figure 9 about here.]

9 Discussion

Besides the *Dictyostelium* aggregation and sporulation, in its two years-long development and testing GenePath has been tried on a number of other problems. For instance, using 28 experiments, GenePath found a correct genetic network for growth and development of *Dictyostelium* that includes 5 genes [17, 16]. From the data on 20 experiments, it successfully reconstructed the programmed cell death genetic network of *C. elegans* that involves four genes [11]. In its most elaborate test, 79 experiments involving 16 genes were presented to GenePath. Data were encoded to eliminate the possibility of help from an informed geneticist: genes where not named but rather given an ID number and the origin of the data was unknown. At the start, GenePath pointed out the inconsistencies in the data and identified a single gene that was a source of them. After that gene was excluded, GenePath developed a single solution in the form of a (linear) path. When this was presented to the owner of the data, it was confirmed to be correct and consistent with the one published [4].

Proposing genetic experiments is the newest addition to GenePath. We believe this mechanism will become crucial for GenePath to seamlessly support explorative genetic data analysis. Experiments in sporulation and aggregation, as described in this paper, show that experiment proposal may indeed point out to the experiments that are relevant and are needed to perform further exploration. Most notably, this has been shown in the experiment proposed by GenePath to relate *pkaR* and *regA*. Besides sole experiment proposal, this schema can be used also for what-if and explorative analysis, where genetic experiments are not necessarily performed in the laboratory but are rather hypothesized and studied for their effect on the regulatory mechanisms.

With respect to scalability of GenePath, the abductive part of GenePath should scale well: most GenePath's patterns are defined over a couple of genes and although the corresponding search space grows exponentially with the number of genes considered, geneticists would usually investigate regulatory networks that include at most a few

dozen genes. For both problems examined in this paper, GenePath constructed the resulting genetic networks within few seconds of CPU time (Pentium IV, 1200 MHz). The response time was equally fast even for our biggest real-life data set with 16 genes and 79 experiments mentioned above.

To test GenePath on data sets of the size beyond currently available ones in classical genetics, we have computer-generated several large genetic networks, constructed experiments from them and gave them to GenePath for reconstruction. The networks we have generated were constrained in the number of parallel paths. Experiments were generated so that we made sure that all necessary epistasis relations could be determined from the data. The results of these experiments are given in Table 4 and show that GenePath effectively handles data that contains hundreds of genes and experiments. Even for the largest case investigated (1000 genes and about 3000 experiments) the response time was reasonable. Further experiments show that the time to find relations from the data is neglectable in comparison with the time needed to construct genetic networks from abduced constraints. Notice that these experiments are rather unrealistic and largely exceed in size the currently biggest problems examined by contemporary classical genetics.

[Table 4 about here.]

Different from the abduction of relations and network generation, the experiment proposal part of GenePath is prone to scalability problems, as with the insufficient data there may be many pairs of genes with undefined relation and many corresponding sets of experiments that can be proposed. A way to limit this combinatorial explosion is expert's guidance: a geneticist can point out an interesting gene pair to explore, and only the least complex experiment sets will be considered. We do not regard this as a deficiency, as we view GenePath as an intelligent assistant and value its interactivity, rather than aim at providing a self-sufficient system for autonomous exploration of genetic data.

10 Related Work

GenePath uses a genetic logic similar to that described by Avery and Wasserman [2] for determining gene order by epistasis in regulatory networks. It uses abduction (see, for instance, [8]) as an inference mechanism, and principles of expert systems and artificial intelligence for combining expert and domain knowledge and data.

The best-known AI system intended for application in classical genetics is Mark Stefik's MOLGEN [18]. MOLGEN is an expert system for planning gene-cloning experiments in molecular genetics. Stefik [18] gives a detailed example of how MOLGEN reconstructed a solution to a rat-insulin gene cloning problem solved previously by Ullrich et al. [20]. However, due to the limitations of its knowledge base and possibly other difficulties, MOLGEN was never applied to find a solution to a previously unsolved genetic experimental problem.

In contrast to MOLGEN, GenePath is not intended to plan experiments, but to interpret experimental data and suggest new experiments that would, if carried out, provide most useful new information about the regulatory mechanism under study. Also, GenePath uses different AI techniques than MOLGEN. GenePath performs abductive inference, whereas the main AI mechanism in MOLGEN is hierarchical means-ends design of plans.

In terms of application, the problem of genetic network construction from data has recently gained substantial interest, but in modelling most of the related work focuses on the analysis of gene expression data [5]. For instance, Friedman at al. [6] use Bayesian networks to discover and Shrager et al. [15] use heuristic search to revise genetic networks, and Akutsu et al. [1] infer genetic networks in the form of Boolean or qualitative networks. Like GenePath, most contemporary systems infer networks which are directional and include both excitation and inhibition links.

Data considered by GenePath comes from classical genetics, *i.e.* each experiment observes the effect of the mutation on some biological process. The motivation for having a program that supports explorative research in this area is twofold. First, most of research in functional genomics is still based on classical phenotypes. And second, there

is a potential bridge between classical genomics and expression data analysis where a whole gene expression array would be considered as a mutant's phenotype [7]. It is beyond the scope of this paper to speculate on how gene relations could be inferred from expression array phenotypes, yet we have to note that even through such mechanisms the genetic networks would remain small due to the requirement for the construction and experimentation with single and double mutants (see also Introduction). Namely, epistasis, a cornerstone relation for construction of genetic networks, can by contemporary genetics be proved only through the use of mutants, and to gather a higher volume of relevant data would require solving the problem of constructing and analyzing a large number of mutants in parallel (see, for instance, [12, 21]). Such experiments are likely to be performed in other organisms in the near future [9, 19], so the need for automated methods for organizing genes in genetic pathways is evident.

11 Conclusion

GenePath has been a subject of experimental verification on real-life data and current problems in genetics for over two years, and is now approaching the point where it can assist geneticists, scholars and students in discovering and reasoning on genetic networks. In the paper, we have presented and discussed several original and essential GenePath's mechanisms: abduction of relations between genes and biological processes, construction of genetic networks and proposal of genetic experiments. While GenePath's implementation in Prolog already embeds all these components, we plan to accordingly enhance its Web-based interface (http://genepath.org), which at present supports experiment and data cataloging, abductive inference and network construction. With these, we believe that GenePath will mature into a seamless intelligent assistant for explorative genetic data analysis and reasoning on genetic networks.

Acknowledgment

This work was supported by a grant from Slovene Ministry of Education, Science and Sport (J2-3387-1539), by a grant from the National Institute of Child Health and Human Development (P01 HD39691-01) and by a travel grant from the National Academy of Sciences under the Collaboration in Basic Science and Engineering supported by Contract No. INT-0002341 from the National Science Foundation. G.S. is a recipient of the Basil O'Connor research award from the March of Dimes Birth Defects Foundation (5-FY99-735).

References

[1] T. Akutsu, S. Miyano, and S. Kuhara. Inferring qualitative relations in genetic networks and methabolic pathways. *Bioinformatics*, 16:727–734, 2000.

[2] L. Avery and S. Wasserman. Ordering gene function: the interpretation of epistasis in regulatory hierarchies. *Trends Genet.*, 8(9):312–6, 1992.

[3] I. Bratko. *Prolog Programming for Artificial Intelligence*. Addison-Wesley, Reading, MA, third edition, 2001.

[4] d. L. Riddle, M. M. Swanson, and P. S. Albert. Interacting genes in nematode dauer larva formation. *Nature*, (290):668–671, 1981.

[5] B. Dutilh. Analysis of data from microarray experiments, the state of the art in gene network reconstruction. Literature thesis, Theoretical biology and Bioinformatics, Utrecht University, 1999.

[6] N. Friedman, M. Linial., I. Nachman, and D. Pe'er. Using bayesian networks to analyze expression data. *J. Comp. Bio.*, 7:601–620, 2000.

[7] T. R. Hughes, M. J. Marton, A. R. Jones, C. J. Roberts, R. Stoughton, C. D. Armour, H. A. Bennett, E. Coffey, H. Dai, Y. D. He, M. J. Kidd, A. M. King, M. R. Meyer, D. Slade, P. Y. Lum, S. B. Stepaniants, D. D. Shoemaker, D. Gachotte, K. Chakraburtty, J. Simon, M. Bard, and S. H. Friend. Functional discovery via a compendium of expression profiles. *Cell*, 102(1):109–26, 2000.

[8] A. C. Kakas, R. A. Kowalski, and F. Toni. *Handbook of logic in Artificial Intelligence and Logic Programming*, volume 5, chapter The role of abduction in logic programming, pages 235–324. Oxford University Press, 1998.

[9] A. Kuspa, R. Sucgang, and G. Shaulsky. The promise of a protist: the Dictyostelium genome project. *Functional and Integrative Genomics*, 1:279–293, 2001.

[10] W. F. Loomis. Role of PKA in the timing of developmental events in Dictyostelium cells. *Microbiol. Mol. Biol. Rev.*, 62(3):684–94, 1998.

[11] M. M. Metzstein, G. M. Stanfield, and H. R. Horvitz. Genetics of programmed cell death in *C. elegans*: past, present and future. *Trends in Genetics*, 14(10):410–416, 1998.

[12] P. Ross-Macdonald, P. S. Coelho, T. Roemer, S. Agarwal, A. Kumar, R. Jansen, K. H. Cheung, A. Sheehan, D. Symoniatis, L. Umansky, M. Heidtman, F. K. Nelson, H. Iwasaki, K. Hager, M. Gerstein, P. Miller, G. S. Roeder, and M. Snyder. Large-scale analysis of the yeast genome by transposon tagging and gene disruption. *Nature*, 402:413–8, 1999.

[13] G. Shaulsky, D. Fuller, and W. F. Loomis. cAMP-phosphodiesterase controls PKA-dependent differentiation. *Development*, 125(4):691–9, 1998.

[14] G. Shaulsky, A. Kuspa, and W. F. Loomis. A multidrug resistance transporter/serine protease gene is required for prestalk specialization in Dictyostelium. *Genes Devel.*, 9(9):1111–22, 1995.

[15] J. Shrager, P. Langley, and A. Pohorille. Guiding revision of regulatory models with expression data. In *Proc. Pacific Symposium on Biocomputing*, pages 486–497, 2002.

[16] G. M. Souza, A. M. da Silva, and A. Kuspa. Starvation promotes Dictyostelium development by relieving PufA inhibition of PKA translation through the YakA kinase pathway. *Development*, 126(14):3263–74, 1999.

[17] G. M. Souza, S. J. Lu, and A. Kuspa. YakA, a protein kinase required for the transition from growth to development in Dictyostelium. *Development*, 125(12):2291–2302, 1998.

[18] M. Stefik. Planning with constraints (MOLGEN: Part1). *Artificial Intelligence*, 16:111–140, 1981.

[19] R. Sucgang, G. Shaulsky, and A. Kuspa. Toward the functional analysis of the Dictyostelium discoideum genome. *J Eukaryot Microbiol*, 47:334–9, 2000.

[20] A. Ullrich, J. Shine, J. Chirgwin, R. Pictet, E. Tischer, W. J. Rutter, and H. M. Goodman. Rat insulin genes: construction of plasmids containing the coding sequences. *Science*, 196:1313–1319, 1977.

[21] E. A. Winzeler, D. D. Shoemaker, A. Astromoff, H. Liang, K. Anderson, B. Andre, R. Bangham, R. Benito, J. D. Boeke, H. Bussey, A. M. Chu, C. Connelly, K. Davis, F. Dietrich, S. W. Dow, M. El Bakkoury, F. Foury, S. H. Friend, E. Gentalen, G. Giaever, J. H. Hegemann, T. Jones, M. Laub, H. Liao, and R. W. Davis. Functional characterization of the S. cerevisiae genome by gene deletion and parallel analysis. *Science*, 285:901–6, 1999.

[22] C. Zimmer. The slime alternative. *Discover*, pages 86–93, May 1998.

[23] B. Zupan, J. Demsar, I. Bratko, P. Juvan, J. Halter, A. Kuspa, and G. Shaulsky. Genepath: a system for automated construction of genetic networks from mutant data. *Bioinformatics*, in print.

List of Figures

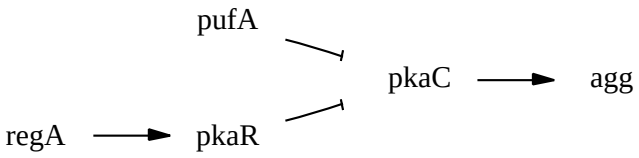


Figure 1: An example of genetic network.

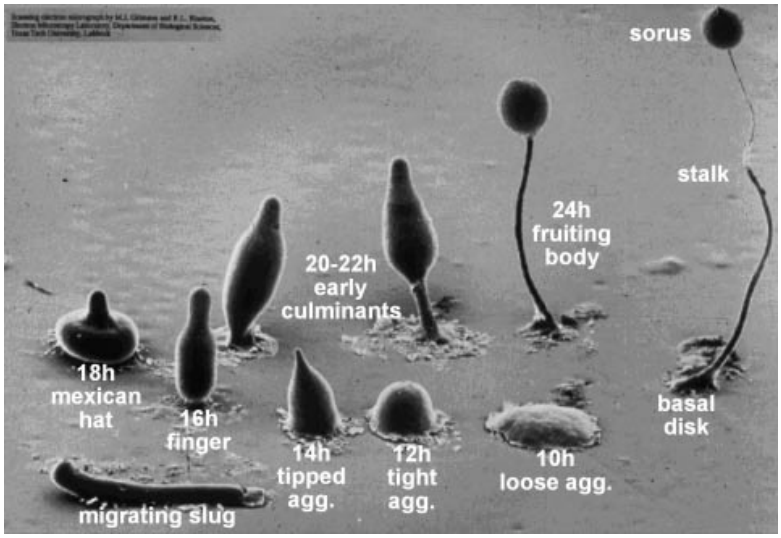


Figure 2: Development of *Dictyostelium* (R. Blanton, M. Grimson, Texas Tech. Univ.)

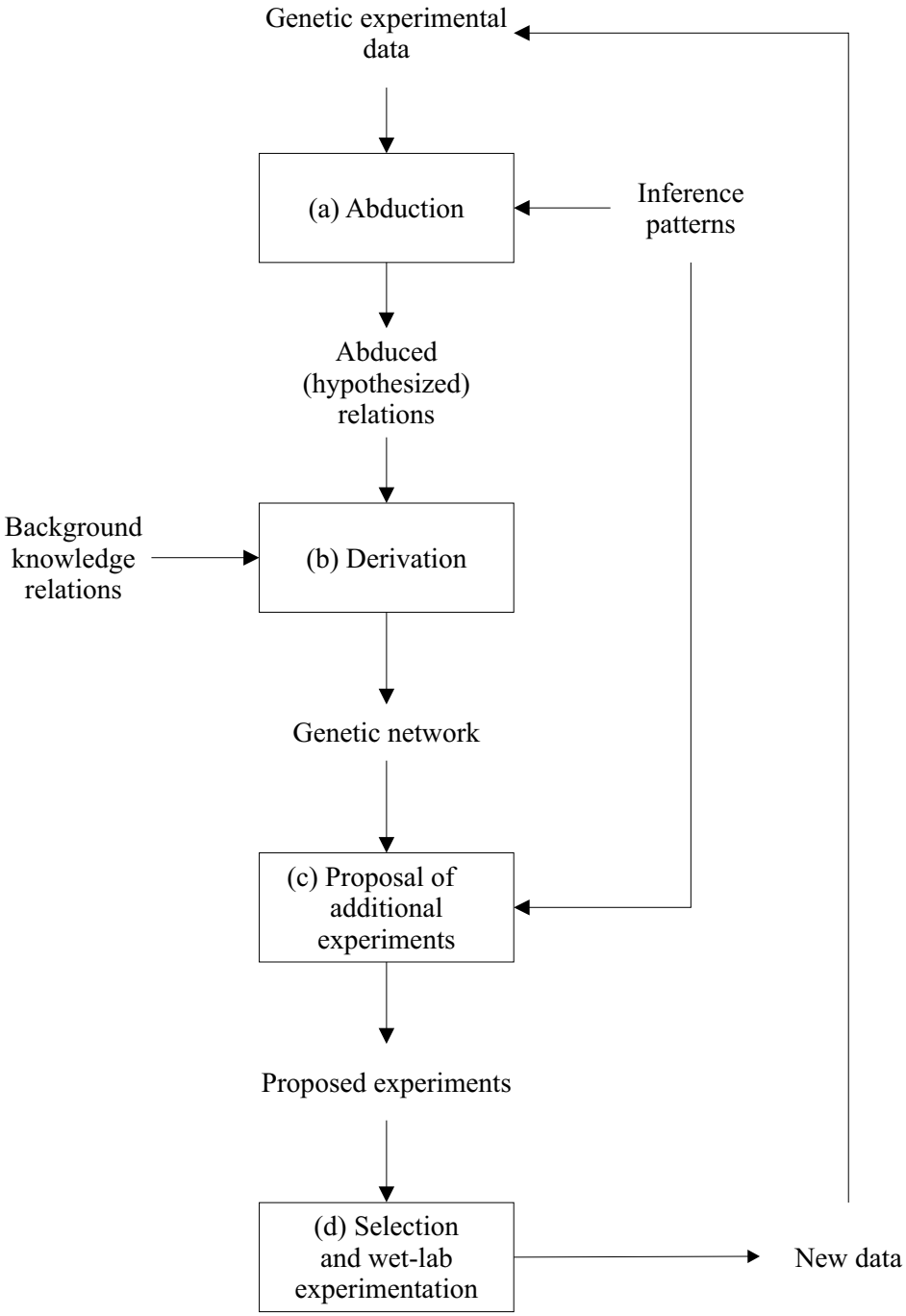


Figure 3: Overview of GenePath.

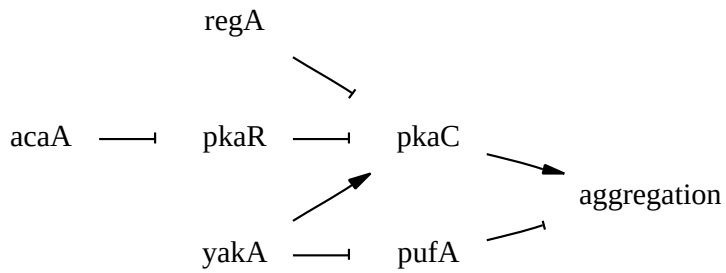


Figure 4: Genetic network for aggregation of *Dictyostelium* derived through constraint satisfaction

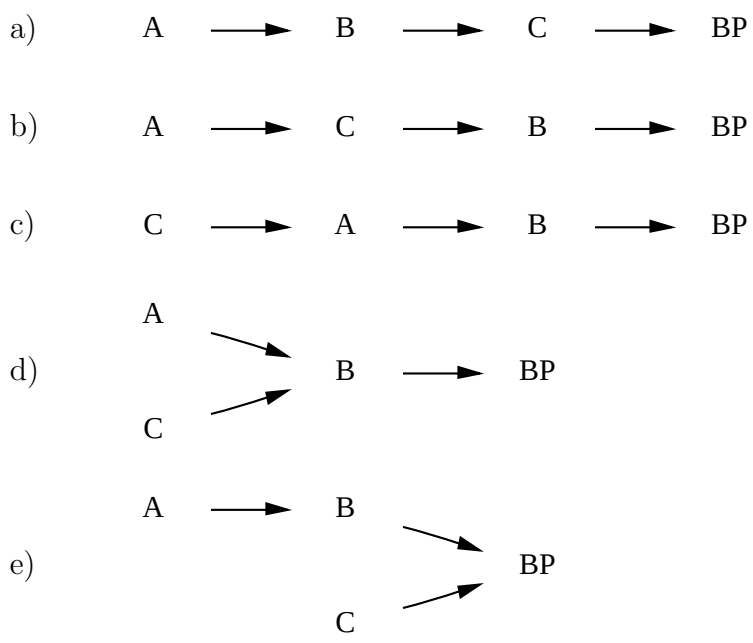


Figure 5: Five different genetic networks, all consistent with constraints that require epistasis of gene B over A and a positive influence of genes A, B and C to biological process BP.

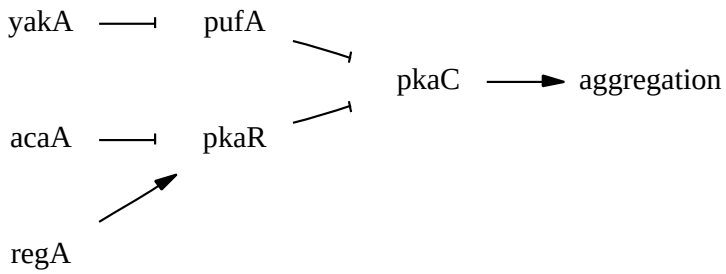
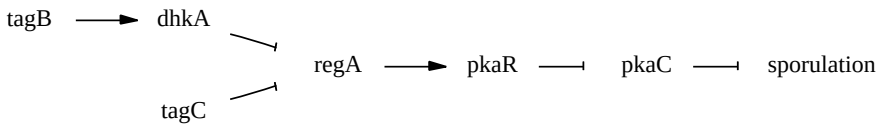
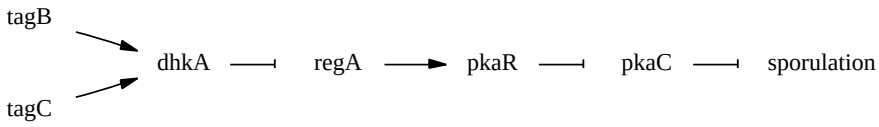


Figure 6: Genetic network for aggregation of *Dictyostelium* based on analysis of data and proposal of new genetic experiments



(a)



(b)

Figure 7: Two genetic networks for sporulation of *Dictyostelium*, one as derived from the data from Table 2 (a), and another one using additional experiments that determine the relation between *tagC* and *dhkA* (b).

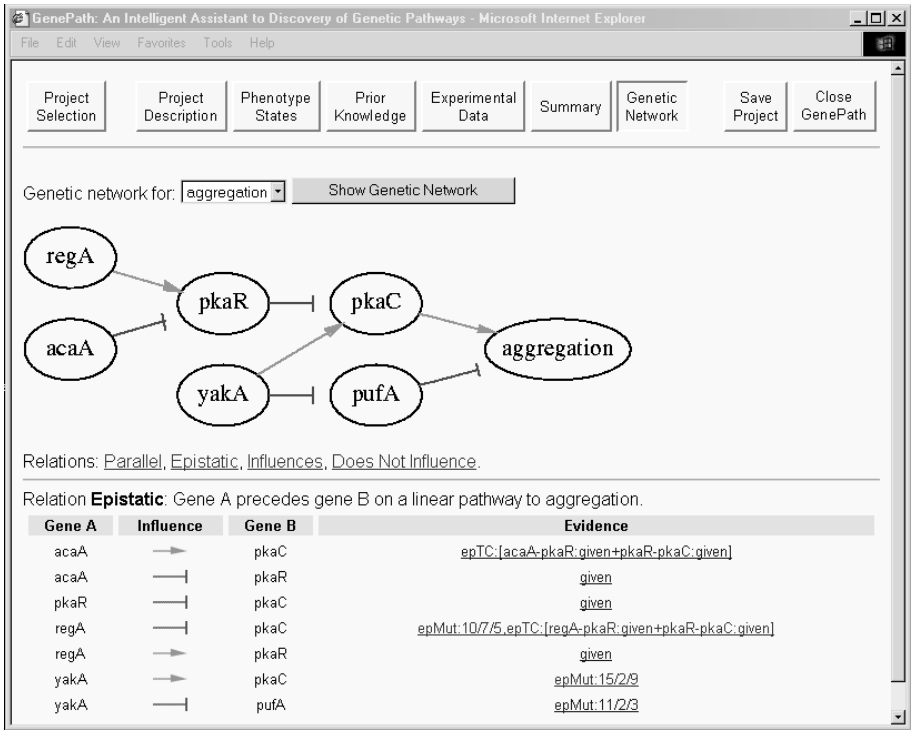


Figure 8: A web-based interface showing the results of analysis.

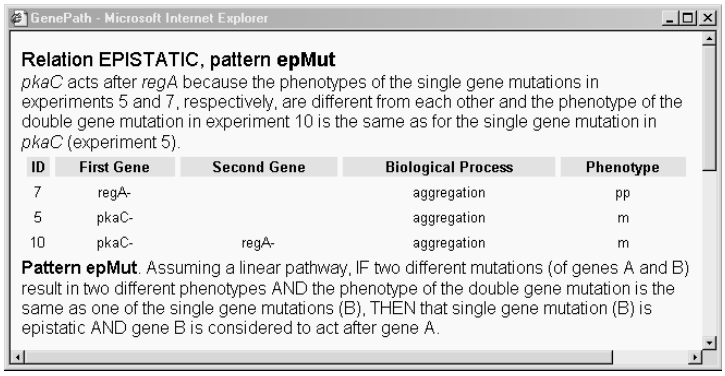


Figure 9: Explanation of the abduced relation $regA \dashv pkaC$.

List of Tables

Table 1: Genetic data set: aggregation of *Dictyostelium*

#	Genotype	aggregation
1	wild-type	+
2	*yakA*-	-
3	*pufA*-	++
4	*pkaR*-	++
5	*pkaC*-	-
6	*acaA*-	-
7	*regA*-	++
8	*acaA* +	++
9	*pkaC* +	++
10	*pkaC*-, *regA*-	-
11	*yakA*-, *pufA*-	++
12	*yakA*-, *pkaR*-	±
13	*yakA*-, *pkaC*-	-
14	*pkaC*-, *yakA* +	-
15	*yakA*-, *pkaC* +	++

Table 2: Experimental data on *Dictyostelium* sporulation

#	Genotype	Sporulation
1	wild-type	normal
2	*dhkA*-	slow
3	*dhkA*-, *pkaC* +	rapid
4	*dhkA*-, *pkaR*-	rapid
5	*dhkA*-, *regA*-	rapid
6	*pkaR*-	rapid
7	*pkaR* +	slow
8	*regA*-	rapid
9	*regA*-, *pkaR* +	slow
10	*tagB*-	no
11	*tagB*-, *dhkA* +	normal
12	*tagB*-, *pkaR*-	rapid
13	*tagB*-, *regA*-	rapid
14	*tagC*-	no
15	*pkaC* +	rapid
16	*pkaC*-	no
17	*tagC*-, *pkaR*-	rapid
18	*tagC*-, *regA*-	rapid
19	*dhkA* +	normal

Table 3: Epistatic relations for sporulation of *Dictyostelium*

Epistatic Relation	Evidence
$dhkA \rightarrow pkaC$	epMut, 2, 15, 3
$dhkA \dashv pkaR$	epMut, 2, 6, 4
$dhkA \dashv regA$	epMut, 2, 8, 5
$regA \rightarrow pkaR$	epMut, 7, 8, 9
$regA \dashv pkaC$	epTC, [$regA \rightarrow pkaR$ and $pkaR \dashv pkaC$]
$tagB \rightarrow pkaR$	epMut, 10, 6, 12
$tagB \dashv regA$	epMut, 10, 8, 13
$tagB \rightarrow pkaC$	epTC, [$tagB \rightarrow dhkA$ and $dhkA \dashv pkaC$]
$tagB \rightarrow dhkA$	epMut, 10, 19, 11
$tagC \dashv regA$	epMut, 14, 8, 18
$tagC \dashv pkaR$	epMut, 14, 6, 17
$tagC \rightarrow pkaC$	epTC, [$tagC \rightarrow pkaR$ and $pkaR \dashv pkaC$]

Table 4: GenePath's response time on a set of computer-generated data sets. (#Genes = number of genes in experiments, #Exp = number of experiments (about 1/3 of these are double mutants), #Par = number of parallel paths in the network, $T_{relations}[s]$ = CPU time in seconds for abduction of relations, $T_{network}[s]$ = CPU time in seconds for construction of genetic network).

#Genes	#Exp	#Par	$T_{relations}[s]$	$T_{network}[s]$
1000	2998	50	43	2262
500	1259	25	8	308
500	1205	0	8	353
250	650	25	3	50
250	633	10	3	37
250	615	0	2	46
100	264	10	1	4
100	261	5	<1	4
100	256	0	<1	2