A2-D6

Implementation of prototypes

Project title:	Reasoning on the Web with Rules and Semantics
Project acronym:	REWERSE
Project number:	IST-2004-506779
Project instrument:	EU FP6 Network of Excellence (NoE)
Project thematic priority:	Priority 2: Information Society Technologies (IST)
Document type:	D (deliverable)
Nature of document:	R (report)
Dissemination level:	PU (public)
Document number:	IST506779/Dresden/A2-D6/D/PU/b
Responsible editors:	Albert Burger, Gihan Dawelbait, François Fages, Patrick Lambrix, Sylvain Soliman, Sebastian Will
Reviewers:	Michael Schroeder
Contributing participants:	Dresden, Edinbrugh, Freiburg, Linköping, Paris
Contributing workpackages:	A2
Contractual date of deliverable:	28 February 2007
Actual submission date:	28 February 2007

Abstract
In this deliverable we present the first results and prototypes from the demonstrators previously defined in deliverable A2-D4.

Keyword List
Prototypes, bioinformatics, ontologies, rules, reasoning, sequence alignment

Project co-funded by the European Commission and the Swiss Federal Office for Education and Science within the Sixth Framework Programme.

Implementation of prototypes

Rolf Backofen^{Frei}, Albert Burger^{Edin}, Anke Busch^{Frei}, Gihan Dawelbait^{Dre}, François Fages^{Par}, Vaida Jakonien^{Lin}, Patrick Lambrix^{Lin}, Kenneth McLeod^{Edin}, Sylvain Soliman^{Par}, He Tan^{Lin}, Sebastian Will^{Frei},

^{Dre} Technische Universität Dresden, Germany, ^{Par} INRIA Rocquencourt Paris, France, ^{Lin} Linköping universitet, Sweden ^{Frei} Albert-Ludwigs-Universität, Freiburg, Germany ^{Edin} Heriot-Watt University Edinburgh, Scotland

28 February 2007

Abstract
In this deliverable we present the first results and prototypes from the demonstrators previously defined in deliverable A2-D4.

Keyword List
Prototypes, bioinformatics, ontologies, rules, reasoning, sequence alignment

Contents

1 Type Inference in Systems Biology (Paris) 3
1.1 Introduction . . . 3
1.2 Preliminaries on Abstract Interpretation, Type Checking and Type Inference . . 4
1.2.1 Concrete Domain of Reaction Models . . . 4
1.2.2 Abstract Domains, Abstractions and Galois Connections . . . 5
1.2.3 Type Checking and Type Inference by Abstract Interpretation . . . 5
1.3 A Type System for Protein Functions . . . 6
1.3.1 Abstract Domain of Protein Functions . . . 6
1.4 Evaluation Results . . . 6
1.4.1 MAPK model. . . . 6
1.4.2 Kohn's Map. . . . 7
1.5 A Type System for Activation and Inhibitory Influences . . . 7
1.5.1 Abstract Domain of Influences . . . 7
1.6 Evaluation Results . . . 8
1.6.1 MAPK model. . . . 8
1.6.2 p53-Mdm2 model. . . . 8
1.6.3 Kohn's Map. . . . 9
1.7 A Type System for Location Topologies . . . 9
1.7.1 Abstract Domain of Location Topologies . . . 9
1.8 Evaluation Results . . . 9
1.8.1 Models from biomodels.net . . . 9
1.8.2 P53/Mdm2. . . . 10
1.8.3 Delta and Notch Model. . . . 10
1.9 Conclusion . . . 11
1.10 Acknowledgement. . . . 11

2 Ontology Alignment - SAMBO and KitAMO (Linköping) 19
2.1 Ontology alignment framework . . . 19
2.2 SAMBO - an ontology alignment tool . . . 19
2.2.1 KitAMO - a tool for evaluating ontology alignment strategies . . . 23
2.3 Evaluating grouping algorithms - KitEGA . . . 24

3 Using argumentation in online distributed life science resources (Edinbrugh) 31
3.1 introduction . . . 31
3.2 Bioinformatics . . . 31
3.3 Argumentation . . . 32
3.3.1 Argumentation: suggested idea and usage . . . 33
3.3.2 The creation of arguments . . . 33
3.3.3 Application of schemes . . . 34
3.4 Conclusion . . . 34

4 Constraint-based Sequence Alignment by Cluster Tree Elimination (Freiburg) 37
4.1 Introduction . . . 37
4.2 A Constraint Model for Sequence Alignment . . . 38
4.3 Efficient Solving by Cluster Tree Elimination . . . 39
4.4 Extension of the Sequence Alignment Model . . . 40

4.5 Conclusion 44

Introduction

The aim of this deliverable (A2-D6) is to show a detailed representation of the underlying theory behind the prototypes of the tools partly presented in deliverable A2-D4. Further more, the prototypes' results of concrete bioinforamtics applications are shown.

- The **Paris** group has developed the Biocham system and a type system for it.

  Type checking and type inference are important concepts and methods of programming languages and software engineering. Type checking is a way to ensure some level of consistency, depending on the type system, in large programs and in complex assemblies of software components. Type inference provides powerful static analyses of pre-existing programs without types, and facilitates the use of type systems by freeing the user from entering type information. they investigate the application of these concepts to systems biology. More specifically, the paris group consider the Systems Biology Markup Language SBML and the Biochemical Abstract Machine BIOCHAM with their repositories of models of biochemical systems. They study three type systems: one for checking or inferring the functions of proteins in a reaction model, one for checking or inferring the activation and inhibition effects of proteins in a reaction model, and another one for checking or inferring the topology of compartments or locations. Through some examples, they show that the analysis of biochemical models by type inference provides accurate and useful information. Interestingly, such a mathematical formalization of the abstractions used in systems biology already provides some guidelines for the extensions of biochemical reaction rule languages.

- The **Linköping** group developed an ontology alignment system (SAMBO) and a system for evaluating alignment strategies (KitAMO).

  In recent years many biomedical ontologies (e.g [3, 8]) have been developed. They are a key technology for the Semantic Web [9, 4]. The benefits of using ontologies include reuse, sharing and portability of knowledge across platforms, and improved documentation, maintenance, and reliability. Ontologies lead to a better understanding of a field and to more effective and efficient handling of information in that field. Many of the currently developed ontologies contain overlapping information. For instance, OBO lists 18 different anatomy ontologies (February 2007), some of which are deprecated (e.g. Arabidopsis anatomy and Cereal anatomy) and have been replaced by a larger ontology (e.g Plant anatomy) when the large amount of overlap was realized.

  Often we would want to be able to use multiple ontologies and this requires knowledge of the relationships between the terms in the different ontologies. It has been realized that this is a major issue and some organizations have started to deal with it. For instance, in the area of anatomy SOFG (http://www.sofg.org/) has developed the SOFG Anatomy Entry List and an NCBO anatomy workshop was organized to start development of the Common Anatomy Reference Ontology (http://www.bioontology.org/wiki/index.php/-CARO:Main_Page). Also, systems for aligning ontologies, i.e. finding the inter-ontology relationships are being built.

- The **Edinbrugh** group used argumentaion to developed schemes to model the reasoning of an expert and use that reasoning to allow a system to evaluate the data presented to users. This will allow non-expert users to critically assess the wide array of data before further analysing it

- The **Freiburg** group developed a declarative approach that uses Dynamic programming and they show how it can be extended by formulating additional knowledge as constraints in the area of aligning DNA and protein sequences by using careful modeling and applying proper solving strategies.

1 Type Inference in Systems Biology (Paris)

François Fages, Sylvain Soliman

Type checking and type inference are important concepts and methods of programming languages and software engineering. Type checking is a way to ensure some level of consistency, depending on the type system, in large programs and in complex assemblies of software components. Type inference provides powerful static analyses of pre-existing programs without types, and facilitates the use of type systems by freeing the user from entering type information. In this paper, we investigate the application of these concepts to systems biology. More specifically, we consider the Systems Biology Markup Language SBML and the Biochemical Abstract Machine BIOCHAM with their repositories of models of biochemical systems. We study three type systems: one for checking or inferring the functions of proteins in a reaction model, one for checking or inferring the activation and inhibition effects of proteins in a reaction model, and another one for checking or inferring the topology of compartments or locations. We show that the framework of abstract interpretation elegantly applies to the formalization of these abstractions and to the implementation of linear time type checking as well as type inference algorithms. Through some examples, we show that the analysis of biochemical models by type inference provides accurate and useful information. Interestingly, such a mathematical formalization of the abstractions used in systems biology already provides some guidelines for the extensions of biochemical reaction rule languages.

1.1 Introduction

Type checking and type inference are important concepts and methods of programming languages and software engineering [1]. Type checking is a way to ensure some level of consistency, depending on the type system, in large programs and in complex assemblies of software components. Type inference provides powerful static analyzes of pre-existing programs without types, and facilitates the use of type systems by freeing the user from entering type information.

In this paper, we investigate the application of these concepts to systems biology. More specifically, we consider the Systems Biology Markup Language SBML [2] and the Biochemical Abstract Machine BIOCHAM [3].

In both of these languages, the biochemical models are described through a set of reaction rules.

We study three type systems:

1. one for checking or inferring the protein functions in a reaction model,

2. one for checking or inferring the activation and inhibition effects in a reaction model,

3. and another one for checking or inferring the topology of compartments or locations in reaction models with space considerations.

To this end, the formal framework of abstract interpretation will be used to provide type systems with a precise mathematical definition. Abstract interpretation is a theory of abstraction introduced by Cousot and Cousot in [4] as a framework for reasoning about programs, their semantics, and for designing static analysers, among which type inference systems [5].

Although not strictly necessary to the presentation of the type inference methods considered in this paper, we believe that that formal framework is very relevant to systems biology, as a formalism for providing a mathematical sense to modeling issues concerning multiple abstraction levels and their formal relationship.

We show that the framework of abstract interpretation elegantly applies to the formalization of the three abstractions considered in this paper and to the implementation of linear time type checking as well as type inference algorithms. Through examples of biochemical systems coming from the `biomodels.net` and BIOCHAM repositories of models, we show that the static analysis of reaction models by type inference provides both accurate and useful information. Interestingly, we show that these considerations also provide some guidelines concerning the extensions of biochemical reaction rule-based languages.

1.2 Preliminaries on Abstract Interpretation, Type Checking and Type Inference

1.2.1 Concrete Domain of Reaction Models

Following SBML and BIOCHAM conventions, a model of a biochemical system is a set of reaction rules of the form e `for` S `=>` S' where S is a set of molecules given with their stoichiometric coefficient, called a *solution*, S' is the transformed solution, and e is a kinetic expression involving the concentrations of molecules (which are not strictly required to appear in S). The set of molecules is noted $\mathcal{M}$. We will use the BIOCHAM operators `+` and `*` to denote solutions as `2*A + B`, as well as the syntax of catalyzed reactions `e for S =[C]=> S'` as an abbreviation for `e for S+C => S'+C`.

A set of reaction rules like $\{e_i$ `for` S_i `=>` $S_i'\}_{i=1,...,n}$ over molecular concentration variables $\{x_1, ..., x_m\}$, can be interpreted under different semantics. The traditional *differential semantics* interpret the rules by the following system of Ordinary Differential Equations (ODE):

$$dx_k/dt = \sum_{i=1}^{n} r_i(x_k) * e_i - \sum_{j=1}^{n} l_j(x_k) * e_j$$

where $r_i(x_k)$ (resp. l_i) is the stoichiometric coefficient of x_k in the right (resp. left) member of rule i. Thanks to its wide range of mathematical tools, this semantics is the most commonly used, when the data is available and the system of a reasonable size. The *stochastic semantics* interpret the kinetic expressions as transition probabilities (see for instance [6]), while the *boolean semantics* forget the kinetic expressions and interpret the rules as a non-deterministic (asynchronous) transition system over boolean states representing the absence or presence of molecules. In BIOCHAM these three semantics are implemented [7], while in the SBML exchange format, no particular semantics are defined.

For the simple analyzes considered in this paper, the concrete domain of reaction models will be the syntactic domain of formal reaction rules, with no other semantics than a data structure. A reaction model is thus a set of reaction rules, and the domain of reaction models is ordered by set inclusion, i.e. by the information ordering.

Definition The universe of reactions is the set of possible rules
$\mathcal{R} = \{e$ `for` S `=>` $S' \mid$ e is a kinetic expression,
and S and S' are solutions $\}$.

The concrete domain $\mathcal{D}_{\mathcal{R}}$ of reaction models is the power-set of reaction rules ordered by inclusion $\mathcal{D}_{\mathcal{R}} = (\mathcal{P}(\mathcal{R}), \subseteq)$.

1.2.2 Abstract Domains, Abstractions and Galois Connections

In the general setting of abstract interpretation, an abstract domain is a lattice $L(\sqsubseteq, \bot, \top, \sqcup, \sqcap)$ defined by the set L and the partial order $\sqsubseteq$, and where $\bot$, $\top$, $\sqcup$, $\sqcap$ denote the least element, the greatest element, the least upper bound and the greatest lower bound respectively.

As often the case in program analysis, the concrete domain and the abstract domains considered for analyzing biochemical models, are power-sets, that is set lattices $\mathcal{P}(\mathcal{S})(\subseteq, \emptyset, \mathcal{S}, \cup, \cap)$ ordered by inclusion, with the empty set as $\bot$ element, and the base set $\mathcal{S}$ (such as the universe of reaction rules here) as $\top$ element. An abstraction is formalized by a Galois connection as follows [4]:

Definition A Galois connection $C \rightarrow_{\alpha} A$ between two lattices C and A is defined by abstraction and concretization functions $\alpha : C \rightarrow A$ and $\gamma : A \rightarrow C$ that satisfy $\forall c \in C, \forall y \in A : x \sqsubseteq_C \gamma(y) \Leftrightarrow \alpha(x) \sqsubseteq_A y$.

For any Galois connection, we have the following properties:

1. $\gamma \circ \alpha$ is extensive (i.e. $x \sqsubseteq_C \gamma \circ \alpha(x)$) and represents the information lost by the abstractions;

2. α preserves $\sqcup$, and γ preserves $\sqcap$;

3. α is one-to-one *iff* γ is onto *iff* $\gamma \circ \alpha$ is the identity.

If $\gamma \circ \alpha$ is the identity, the abstraction α loses no information, and C and A are isomorphic from the information standpoint (although γ may not be one-to-one).

We will consider three abstract domains: one for protein functions, where molecules are abstracted into categories such as kinases and phosphatases, one for the influence graph, where the biochemical reaction rules are abstracted in activation and inhibition binary relations between molecules, and one for location topologies, where the reaction (and transport) rules are abstracted retaining only the neighborhood information between locations.

1.2.3 Type Checking and Type Inference by Abstract Interpretation

In this setting, a type system A for a concrete domain C is simply a Galois connection $C \rightarrow_{\alpha} A$. The *type inference* problem is, given a concrete element $x \in C$ (e.g. a reaction model) to compute $\alpha(x)$ (e.g. the protein functions that can be inferred from the reactions). The *type checking* problem is, given a concrete element $x \in C$ and a typing $y \in A$ (e.g. a set of protein functions), to determine whether $x \sqsubseteq_C \gamma(y)$ (i.e. whether the reactions provide less and compatible information on the protein functions) which is equivalent to $\alpha(x) \sqsubseteq_A y$ (i.e. whether the typing contains the inferred types).

The simple type systems considered in this paper will be implemented with type checking and type inference algorithms that basically browse the reactions, and check or collect the type information for each rule independently and in linear time.

1.3 A Type System for Protein Functions

To investigate the use of type inference in the domain of protein functions we first restrict ourselves to two simple functions: kinase and phosphatase. These correspond to the action of adding (resp. removing) a phosphate group to (resp. from) a compound.

For the sake of simplicity, we do not consider other categories such as protease (in degradation rules), acetylase and deacetylase (in modification rules), etc. This choice is in accordance with the BIOCHAM syntax which allows to mark the modified sites of a protein with the operator `~`, as in `P~{p,q}` without distinguishing however between a phosphorylation and an acetylation for instance. We thus consider BIOCHAM models containing compounds with different levels of phosphorylation or acetylation, without distinguishing the different forms of modification, and call them phosphorylation, by abuse of terminology.

The analysis of protein functions in a reaction model is interesting for several reasons. First, the kind of information (kinase activity) collected on proteins can be checked using online databases like GO, the Gene Ontology [8]. Second, in the context of the machine learning techniques implemented in BIOCHAM for completing or revising a model w.r.t. a temporal logic specification [7], the information that an enzyme acts as a kinase or as a phosphatase drastically reduce the search space for reaction additions, and help find more biologically plausible model revisions.

1.3.1 Abstract Domain of Protein Functions

Definition The abstract domain of protein functions $\mathcal{D}_{\mathcal{F}}$ is the domain of functions from molecules $\mathcal{M}$ to pairs of booleans, representing "has kinase function" (true/false) and "has phosphatase function" (true/false).

Definition $\alpha : \mathcal{D}_{\mathcal{R}} \to \mathcal{D}_{\mathcal{F}}$ is defined for each molecule as the disjunction of α on each single rule and each pair of rules:

α(`A =[B]=> C`) = where `C` is more phosphorylated than `A` (i.e. its set of active phosphorylation sites strictly includes that of `A`) is abstracted as *B has kinase function.*

α(`A =[B]=> C`) = where, on the contrary, `A` is more phosphorylated than `C`, we abstract that *B has phosphatase function.*

α(`A + B => A-B, A-B => C + B`) = where `C` is more phosphorylated than `A` is abstracted as *B has kinase function.*

α(`A + B => A-B, A-B => C + B`) = where, on the contrary, `A` is more phosphorylated than `C`, we abstract that *B has phosphatase function.*

1.4 Evaluation Results

1.4.1 MAPK model.

On a simple example of the MAPK cascade extracted from the SBML repository and originally based on [9], the type inference algorithm determines that `RAFK`, `RAF~{p1}` and `MEK~{p1,p2}` have a kinase function; `RAFPH`, `MEKPH` and `MAPKPH` have a phosphatase function; and the other compounds have no function inferred.

If we wanted to type-check such a model, we would correctly check all phosphatases but would miss an example of the kinase function of `MAPK~{p1,p2}`, since its action is not visible in the above model.

1.4.2 Kohn's Map.

Kohn's map of the mammalian cell cycle control [10] has been transcribed in BIOCHAM to serve as a large benchmarking example of 500 species and 800 rules [11]. To check if this abstraction scales up we tried it on this model, and indeed obtain the answer in less than one second CPU time (on a PC 1,7GHz). Here is an excerpt of the output of the type inference:

```
cdk7-cycH is a kinase
Wee1 is a kinase
Myt1 is a kinase
cdc25C~{p1} is a phosphatase
cdc25C~{p1,p2} is a phosphatase
Chk1 is a kinase
C-TAK1 is a kinase
Raf1 is a kinase
cdc25A~{p1} is a phosphatase
cycA-cdk1~{p3} is a kinase
cycA-cdk2~{p2} is a kinase
cycE-cdk2~{p2} is a kinase
cdk2~{p2}-cycE~{p1} is a kinase
cycD-cdk46~{p3} is a kinase
cdk46~{p3}-cycD~{p1} is a kinase
cycA-cdk1~{p3} is a kinase
cycB-cdk1~{p3} is a kinase
cycA-cdk2~{p2} is a kinase
cycD-cdk46~{p3} is a kinase
cdk46~{p3}-cycD~{p1} is a kinase
Plk1 is a kinase
pCAF is a kinase
p300 is a kinase
HDAC1 is a phosphatase
```

It is worth noticing that in these results no compound is both a kinase and a phosphatase. The `cdc25 A` and `C` are the only phosphatases found in the whole map with `HDAC1`). The type inference also tells us that the cyclin-dependant kinases have a kinase function when in complex with a cyclin. Finally the acetylases `pCAF`, `p300` and the deacetylase `HDAC1` are detected but identified to kinases and phosphatases respectively, since the BIOCHAM syntax does not distinguish between phosphorylation and acetylation.

1.5 A Type System for Activation and Inhibitory Influences

1.5.1 Abstract Domain of Influences

Influence networks for activation and inhibition have been introduced for the analysis of gene expression in the setting of gene regulatory networks [12]. Such influence networks are in fact an abstraction of complex reaction networks, and can be applied as such to protein interaction networks. However the distinction between the influence network and the reaction network is crucial to the application of Thomas's conditions of multistationarity and oscillations [12, 13] to protein interaction network, and there has been some confusion between the two kinds of networks [14]. Here we precisely define influence networks as an abstraction of (or a type system for) reaction networks.

Definition The abstract domain of influences is the powerset of the binary relations of activation and inhibition between compounds $\mathcal{D}_{\mathcal{I}} = \mathcal{P}(\{A \text{ activates } B \mid A, B \in \mathcal{M}\} \cup \{A \text{ inhibits } B \mid A, B \in \mathcal{M}\})$.

The influence abstraction $\alpha : \mathcal{D}_{\mathcal{R}} \to \mathcal{D}_{\mathcal{I}}$ is the function

$$\begin{array}{lll} \alpha(x) = & \{A \text{ inhibits } B & \mid \exists(e_i \text{ for } S_i \Rightarrow S'_i) \in x, \\ & & l_i(A) > 0 \text{ and } r_i(B) - l_i(B) < 0\} \\ & \cup\{A \text{ activates } B & \mid \exists(e_i for S_i \Rightarrow S'_i) \in x, \\ & & l_i(A) > 0 \text{ and } r_i(B) - l_i(B) > 0\} \end{array}$$

In particular, we have the following influences for elementary reactions of complexation, modification, synthesis and degradation:

$$\begin{array}{rl} \alpha(\{\mathtt{A} + \mathtt{B} \Rightarrow \mathtt{C}\}) = \{ & \text{A inhibits B, A inhibits A, B inhibits A,} \\ & \text{B inhibits B, A activates C, B activates C}\} \\ \alpha(\{\mathtt{A} = [\mathtt{C}] \Rightarrow \mathtt{B}\}) = \{ & \text{C inhibits A, A inhibits A, A activates B, C activates B}\} \\ \alpha(\{\mathtt{A} = [\mathtt{B}] \Rightarrow \_\}) = \{ & \text{B inhibits A, A inhibits A}\} \\ \alpha(\{\_ = [\mathtt{B}] \Rightarrow \mathtt{A}\}) = \{ & \text{B activates A}\} \end{array}$$

The inhibition loops on the reactants are justified by the negative sign in the Jacobian matrix of the differential semantics of such reactions. It is worth noting however that they are often omitted in the influence graphs considered in the literature, as well as with some other influences, according to functionality, kinetic and non-linearity considerations.

1.6 Evaluation Results

1.6.1 MAPK model.

Let us first consider the MAPK signalling model of [9]. Fig. 1 depicts the reaction graph as a bipartite graph with round boxes for molecules and rectangular boxes for rules. Fig. 2 depicts the inferred influence graph, where activation (resp. inhibition) is materialized by plain (resp. dashed) arrows. The graph layouts of the figures have been computed in BIOCHAM by the Graphviz suite[1].

1.6.2 p53-Mdm2 model.

In the p53-Mdm2 model of [15], the protein $Mdm2$ is localized explicitly in two possible locations: the nucleus and in the cytoplasm, and transport rules are considered. Fig. 4 depicts the reaction graph of the model.

Fig. 3 depicts the inferred influence graph. Note that $Mdm2$ in the nucleus has both an activation and an inhibitory effect on $p53\ u$. This corresponds to different influences in different regions of the phase space.

Fig. 5 depicts the core influence graph considered for the logical analysis of this model [16]. In the core influence graph, some influence are neglected, as expected, however some inhibitions, such the inhibitory effect of $p53$ on $Mdm2$ in the nucleus, are considered while they do not appear in the inferred influence graph. The reason for these omissions is the way the reaction model is written. Some inhibitory effects are indeed expressed in the kinetic expression by subtraction of, or division by, the molecular concentration of some compounds that do not appear in the rule itself. Those inhibitions are thus missed by the type inference algorithm. An example of such a rule is the following one for the inhibition of $Mdm2$ by $p53$:

[1] `http://www.graphviz.org/`

```
macro(p53tot,[p53]+[p53~{u}]+[p53~{uu}]).
(kph*[Mdm2::c]/(Jph+p53tot),MA(kdeph)) for Mdm2::c <=> Mdm2~{p}::c.
```

Obviously, we cannot expect to infer such inhibitory effects from the kinetic expressions with all generality, however the model being written that way without fully decomposing all influences by reaction rules, a refinement of the abstraction function taking into account the kinetic expression is worth investigating. As an alternative, one could extend the syntax of reaction rules in order to indicate the inhibitors of the reaction, in a somewhat symmetric fashion to catalysts.

1.6.3 Kohn's Map.

On Kohn's map, the type inference of activation and inhibition influences takes less than one second CPU time (on a PC 1,7GHz) for the complete model, showing again the efficiency of the type inference algorithm.

1.7 A Type System for Location Topologies

To date, models of biochemical systems generally abstract from space considerations. Models taking into account cell compartments and transport phenomena are thus much less common. Nevertheless, with the advent of systems biology computational tools, more and more models are refined with space considerations and transport delays, e.g. [15]. In SBML [2] level 1 version 1, locations have been introduced as purely symbolic compartments without topology. We show in this section how the topology can be inferred from the reaction rules, and checked in different models.

1.7.1 Abstract Domain of Location Topologies

Definition Abstract domain of neighborhood relation $\mathcal{D}_{\mathcal{N}}$ is a relation on pairs of molecules $\mathcal{M} \times \mathcal{M}$.

Definition $\alpha : \mathcal{D}_{\mathcal{R}} \to \mathcal{D}_{\mathcal{N}}$ is defined by the union of its definition on single rules:
$\alpha(\texttt{E for } \texttt{A}_1 + \cdots + \texttt{A}_n \texttt{ => } \texttt{B}_1 + \cdots + \texttt{B}_m) =$ All A_i and all B_j are pairwise neighbors, and for all C_k such that `[`$\texttt{C}_k$`]` appears in `E`, C_k is a neighbor of all A_i and all B_j.

1.8 Evaluation Results

1.8.1 Models from biomodels.net

We have taken models from the literature through the `biomodels.net` database. Of the 50 models in the current version (dated January 2006) only 13 have more than one compartment, and only 7 of those use the *outside* attribute of SBML to provide more topological insight.

The neighboring relation is inferred in these models imported in BIOCHAM, and then checked consistent with the provided *outside* relation.

For instance for calcium oscillations, we tried both the Marhl et al. model of [17] and the Borghans et al. model of [18].

In the first case (model `BIOMD0000000039.xml`), three locations are defined: the cytosol, the endoplasmic reticulum and a mitochondria, from the reactions the inferred topology is that the cytosol is neighbor of the two other locations. This correspond exactly to the information obtained from the *outside* annotations (the cytosol being marked as the outside of the two other locations).

In the second case (models BIOMD0000000043.xml to `BIOMD0000000045.xml`) we focused on the last model (*two-pool*) since it is the only one with 4 different locations: the extracellular space, the cytosol and two internal vesiculae. The location inference produces a topology where the cytosol is neighbor of all other locations. Once again this is correct w.r.t. the outside information provided in the SBML file: both vesiculae have the cytosol as outside location and the cytosol itself has the extracellular space as outside location.

These considerations show that there is some mismatch between the SBML reaction models and the choice of expressing outside vs neighborhood properties of locations. In the perspective of type checking and type inference, neighborhood relations should be preferred as they can be checked, or inferred from the reaction model, whereas the outside relation contain more information that, while helpful for the modeler as meta-data, cannot be handled automatically without abstracting it first in neighbors properties.

1.8.2 P53/Mdm2.

The first example comes from [15]: a model of the p53/Mdm2 interaction with two locations where the transport between cytoplasm and nucleus is necessary to explain some time delays observed in the mutual repression of these proteins.

```
biocham: load_biocham('EXAMPLES/locations/p53Mdm2.bc').
...
biocham: show_neighborhood.
c and n are neighbors
```

In this precise case, the model as published does not systematically use the volume ratio in the kinetics. The transcription and type-checking of the model showed that if one wanted to keep the background degradation rate of $Mdm2$ (without DNA damage) independent of the location, one obtains different kinetics than those of the published model. In this case a formal transcription in BIOCHAM (or SBML) provided a supplementary model-validation step.

1.8.3 Delta and Notch Model.

The next example is adapted from [19]. The Delta and Notch proteins are crucial to the cell fate in several different organisms. A population of neighboring cells (here we chose a square grid) is represented through locations and the model allows to observe the salt-and-pepper coloring (corresponding to high Delta-low Notch/low Delta-high Notch) typical of the Delta-Notch lateral inhibition based differentiation. The signaling pathways are simplified to the extreme to take into account only the direct effect of Delta and Notch expression on the local and neighboring cells. This example would thus not provide a good basis for the abstraction of section 1.5.

Depending on the abstraction chosen we obtain figure 6 and 7. In the first case the abstraction used is not the one given in section 1.7.1 but

Definition $\alpha : \mathcal{D}_\mathcal{R} \to \mathcal{D}_\mathcal{N}$ is defined by the union of its definition on single rules:

α(`E for` $\mathtt{A}_1 + \cdots + \mathtt{A}_n$ `=>` $\mathtt{B}_1 + \cdots + \mathtt{B}_m$) = All A_i, all B_j, and all C_k such that `[`$\mathtt{C}_k$`]` appears in `E`, are pairwise neighbors.

This was indeed a reasonable candidate for an abstraction, but proved too coarse on some examples since co-modifiers are often put in the kinetic expression of a single rule for simplification purposes.

1.9 Conclusion

We have shown that the framework of abstract interpretation applies to the formalization of some abstractions commonly used in systems biology, and to the implementation of linear-time type checking as well as type inference algorithms.

In the three type systems studied in this paper, for protein functions, activation and inhibitory influences, and location topologies respectively, the analyses are based on static information gained directly from the syntax of reaction rules, without considering their formal semantics, nor their precise dynamics. It is worth noting that this situation also occurs in program analysis where the syntax of programs may capture a sufficient part of the semantics for many analyses. Here, it is remarkable that such simple analyses already provide useful information on biological models, independently from their dynamics for which different definitions are considered (discrete, continuous, stochastic, etc.) [7].

The formal definition of the influence graph as an abstraction of the reaction model eliminates some confusion that exists in the use of Thomas's conditions [12, 13] for the analysis of reaction models [14]. Such a formalization shows also that the influence graphs usually considered in the literature are further abstractions obtained by forgetting some influences, based on non-linearity considerations [20]. Some inhibitions may also be missing in the inferred influences when they are hidden in the kinetic expressions of the reactions and do not appear explicitly in the reactants. This suggests either to refine the abstraction function to take into account the kinetic expression when possible, or to extend the syntax of reactions in order to make explicit such inhibitory effects, in a symmetric fashion to catalysts for activations. In SBML there is actually an unique symmetrical notion of *Modifiers* which is not sufficient to infer the influence graph.

Similarly, the inference of protein functions and of location neighborhood have shown that the static analysis of reaction models by type inference provides both accurate and useful information. They also provide some guidelines for the extensions of biochemical reaction languages, like for instance in SBML considering neighborhood rather than outside properties, and introducing a syntax for the modification of compounds, and in BIOCHAM differentiating phosphorylation from other forms of modifications like acetylation.

1.10 Acknowledgement.

This work benefited from partial support of the Network of Excellence REWERSE of the European Union.

References

[1] Cardelli, L.: Typeful programming. In Neuhold, E.J., Paul, M., eds.: Formal Description of Programming Concepts. Springer-Verlag, Berlin (1991) 431–507

[2] Hucka, M., et al.: The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models. Bioinformatics **19** (2003) 524–531

[3] Fages, F., Soliman, S., Chabrier-Rivier, N.: Modelling and querying interaction networks in the biochemical abstract machine BIOCHAM. Journal of Biological Physics and Chemistry **4** (2004) 64–73

[4] Cousot, P., Cousot, R.: Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: POPL'77: Proceedings of the 6th ACM Symposium on Principles of Programming Languages, New York, ACM Press (1977) 238–252 Los Angeles.

[5] Cousot, P.: Types as abstract interpretation (invited paper). In: POPL'97: Proceedings of the 24th ACM Symposium on Principles of Programming Languages, New York, ACM Press (1997) 316–331 Paris.

[6] Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. Journal of Physical Chemistry **81** (1977) 2340–2361

[7] Calzone, L., Chabrier-Rivier, N., Fages, F., Soliman, S.: Machine learning biochemical networks from temporal logic properties. Transactions on Computational Systems Biology (2006) CMSB'05 Special Issue (to appear).

[8] Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Gene ontology: tool for the unification of biology. Nature Genetics **25** (2000) 25–29

[9] Levchenko, A., Bruck, J., Sternberg, P.W.: Scaffold proteins may biphasically affect the levels of mitogen-activated protein kinase signaling and reduce its threshold properties. PNAS **97** (2000) 5818–5823

[10] Kohn, K.W.: Molecular interaction map of the mammalian cell cycle control and DNA repair systems. Molecular Biology of the Cell **10** (1999) 2703–2734

[11] Chabrier-Rivier, N., Chiaverini, M., Danos, V., Fages, F., Schächter, V.: Modeling and querying biochemical interaction networks. Theoretical Computer Science **325** (2004) 25–44

[12] Thomas, R., Gathoye, A.M., Lambert, L.: A complex control circuit : regulation of immunity in temperate bacteriophages. European Journal of Biochemistry **71** (1976) 211–227

[13] Soulé, C.: Graphic requirements for multistationarity. ComplexUs **1** (2003) 123–133

[14] Markevich, N.I., Hoek, J.B., Kholodenko, B.N.: Signaling switches and bistability arising from multisite phosphorylation in protein kinase cascades. Journal of Cell Biology **164** (2005) 353–359

[15] Ciliberto, A., Novák, B., Tyson, J.J.: Steady states and oscillations in the p53/mdm2 network. Cell Cycle **4** (2005) 488–493

[16] Kaufman, M.: Private communication. (2006)

[17] Marhl, M., Haberichter, T., Brumen, M., Heinrich, R.: Complex calcium oscillations and the role of mitochondria and cytosolic proteins. BioSystems **57** (2000) 75–86

[18] Borghans, J., Dupont, G., Goldbeter, A.: Complex intracellular calcium oscillations: a theoretical exploration of possible mechanisms. Biophysical Chemistry **66** (1997) 25–41

[19] Ghosh, R., Tomlin, C.: Lateral inhibition through delta-notch signaling: A piecewise affine hybrid model. In Springer-Verlag, ed.: Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control, HSCC'01. Volume 2034 of Lecture Notes in Computer Science., Rome, Italy (2001) 232–246

[20] Thomas, R., Kaufman, M.: Multistationarity, the basis of cell differentiation and memory. Chaos **11** (2001) 170–195

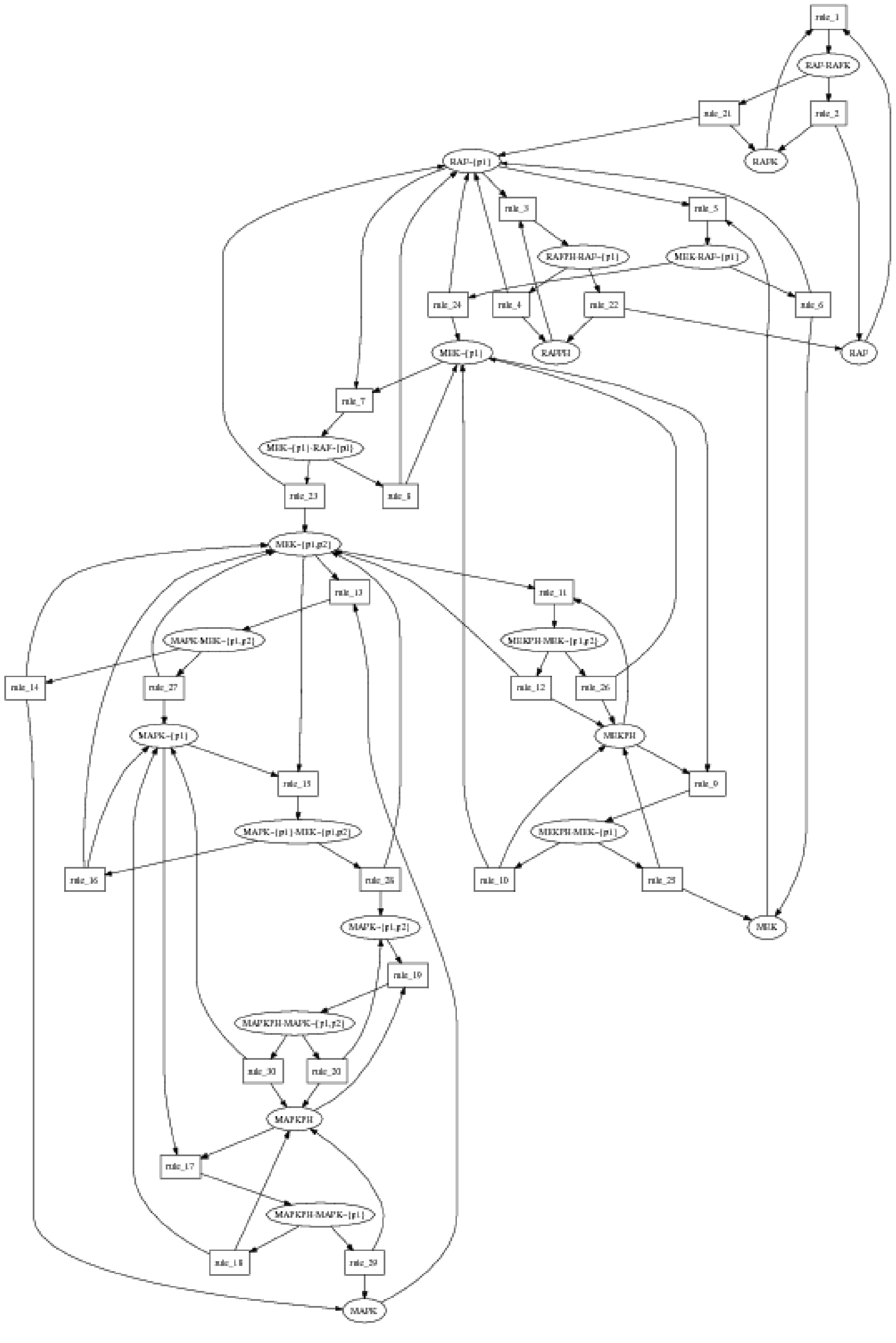


Figure 1: Reaction graph of the MAPK model

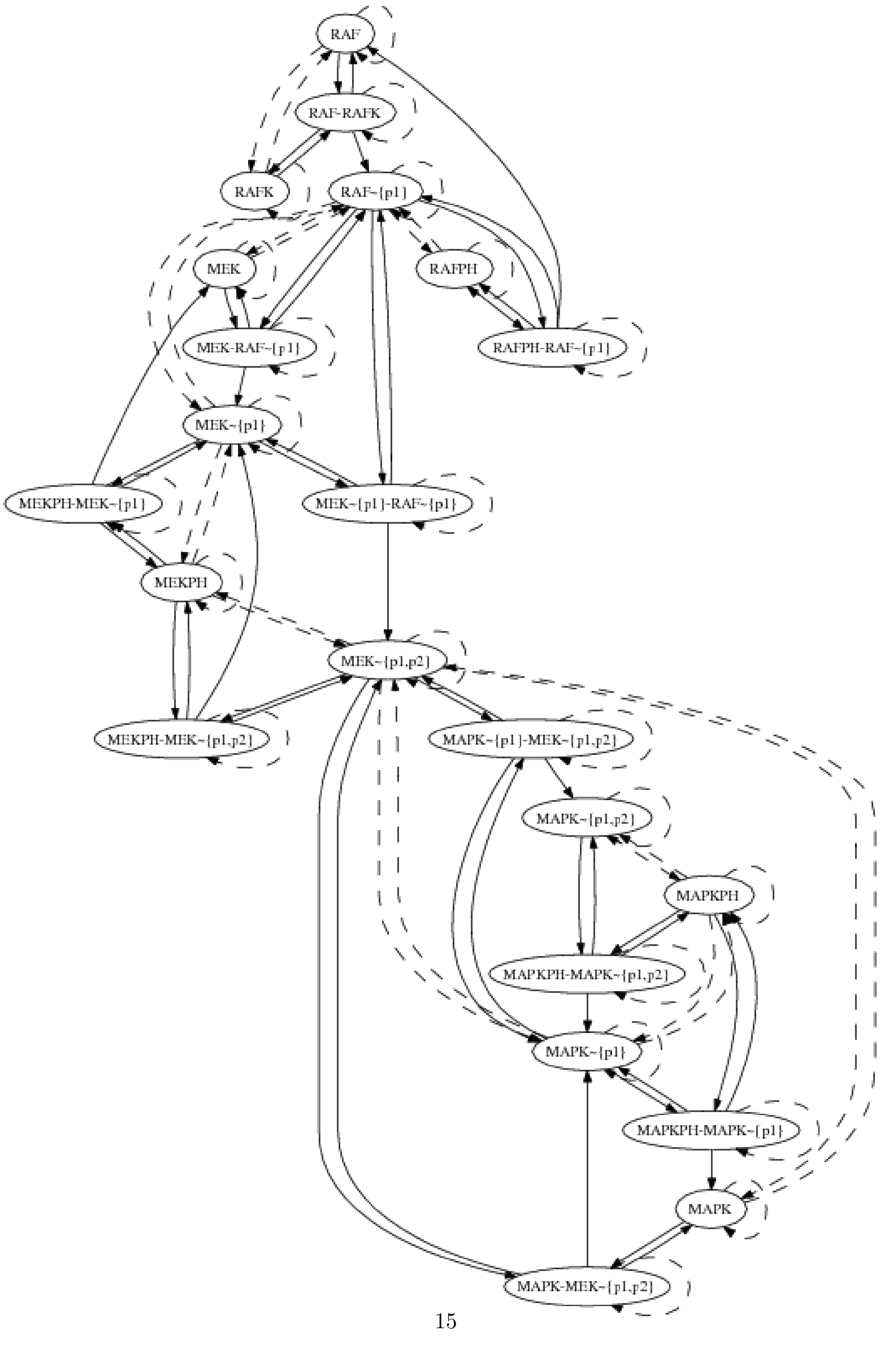


Figure 2: Inferred influence graph of the MAPK model

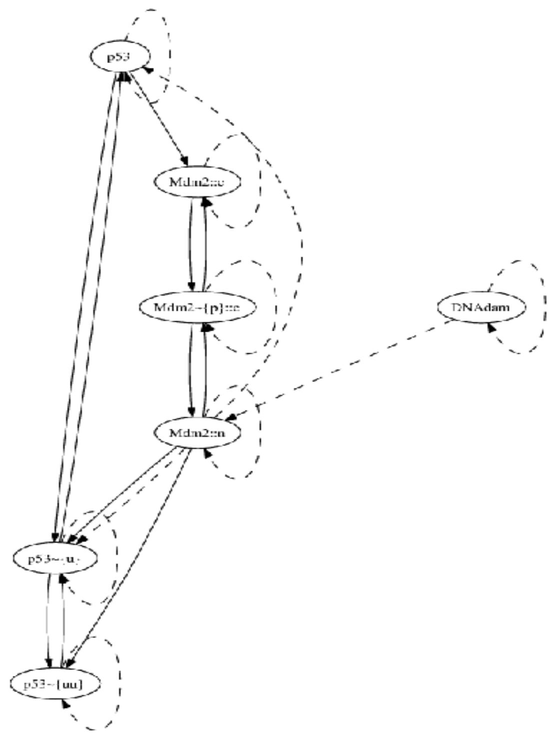


Figure 3: Inferred influence graph of the p53-Mdm2 model

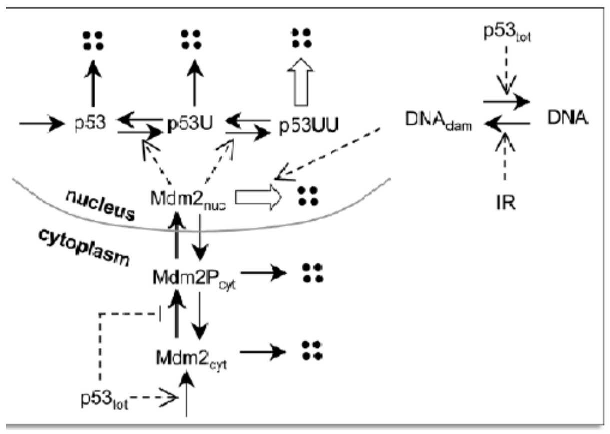


Figure 4: Original reaction graph considered in [15] for the p53-Mdm2 model.

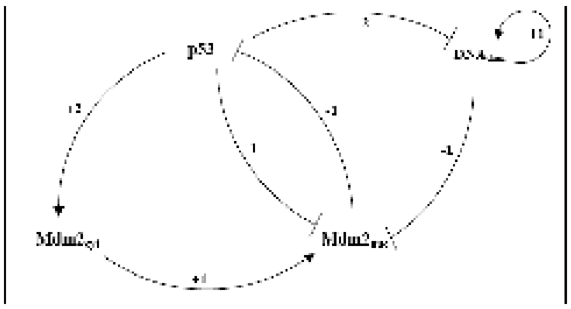


Figure 5: Core influence graph.

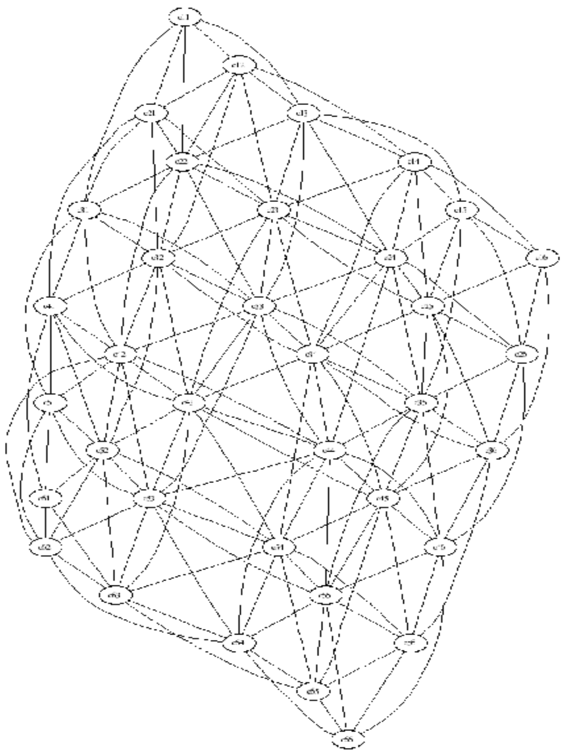


Figure 6: Delta-Notch square cell grid inferred in a 6x6 model, with modifiers, reactants and products as pairwise neighbors

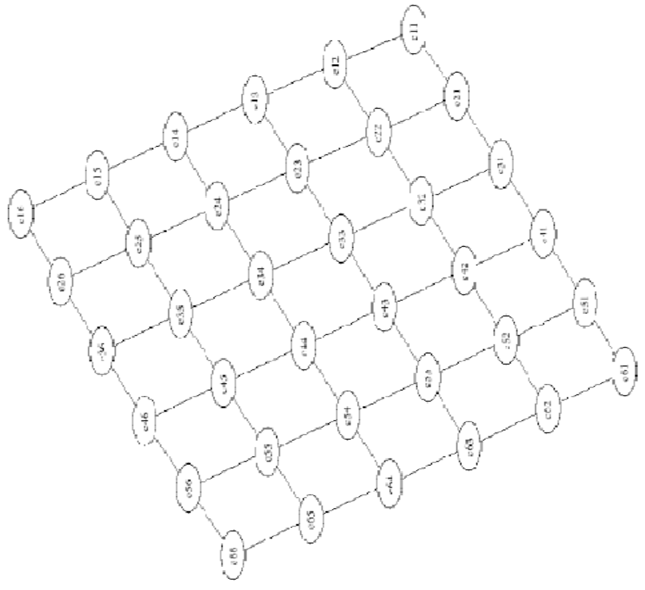


Figure 7: Delta-Notch square cell grid inferred in a 6x6 model, without modifier-modifier neighborhood

2 Ontology Alignment - SAMBO and KitAMO (Linköping)

Patrick Lambrix, He Tan, Vaida Jakonienė

In recent years many biomedical ontologies (e.g [3, 8]) have been developed. They are a key technology for the Semantic Web [9, 4]. The benefits of using ontologies include reuse, sharing and portability of knowledge across platforms, and improved documentation, maintenance, and reliability. Ontologies lead to a better understanding of a field and to more effective and efficient handling of information in that field. Many of the currently developed ontologies contain overlapping information. For instance, OBO lists 18 different anatomy ontologies (February 2007), some of which are deprecated (e.g. Arabidopsis anatomy and Cereal anatomy) and have been replaced by a larger ontology (e.g Plant anatomy) when the large amount of overlap was realized.

Often we would want to be able to use multiple ontologies and this requires knowledge of the relationships between the terms in the different ontologies.

In this section we briefly describe an ontology alignment system (SAMBO) and a system for evaluating alignment strategies (KitAMO) developed at Linköpings universitet. For more information we refer to [5, 6, 7].

2.1 Ontology alignment framework

Many of the current systems are based on the computation of similarity values between terms in the source ontologies, and can be seen as instantiations of the framework defined in [5]. This framework is shown in figure 8. It consists of two parts. The first part (*I* in figure 8) computes alignment suggestions. The second part (*II*) interacts with the user to decide on the final alignments. Some systems may not have the second part. An alignment algorithm receives as input two source ontologies. The algorithm can include several matchers. The matchers can implement strategies based on linguistic matching, structure-based strategies, constraint-based approaches, instance-based strategies, strategies that use auxiliary information or a combination of these. Each matcher utilizes knowledge from one or multiple sources. The matchers calculate similarities between the terms from the different source ontologies. Alignment suggestions are then determined by combining and filtering the results generated by one or more matchers. By using different matchers and combining and filtering the results in different ways we obtain different alignment strategies. The suggestions are then presented to the user who accepts or rejects them. The acceptance and rejection of a suggestion may influence further suggestions. Further, a conflict checker is used to avoid conflicts introduced by the alignment relationships. The output of the alignment algorithm is a set of alignment relationships between terms from the source ontologies.

2.2 SAMBO - an ontology alignment tool

SAMBO[2] [5, 7] is developed according to the framework described above. The current implementation supports ontologies in OWL format and deals with alignment of relations and

[2]http://www.ida.liu.se/∼iislab/projects/SAMBO/

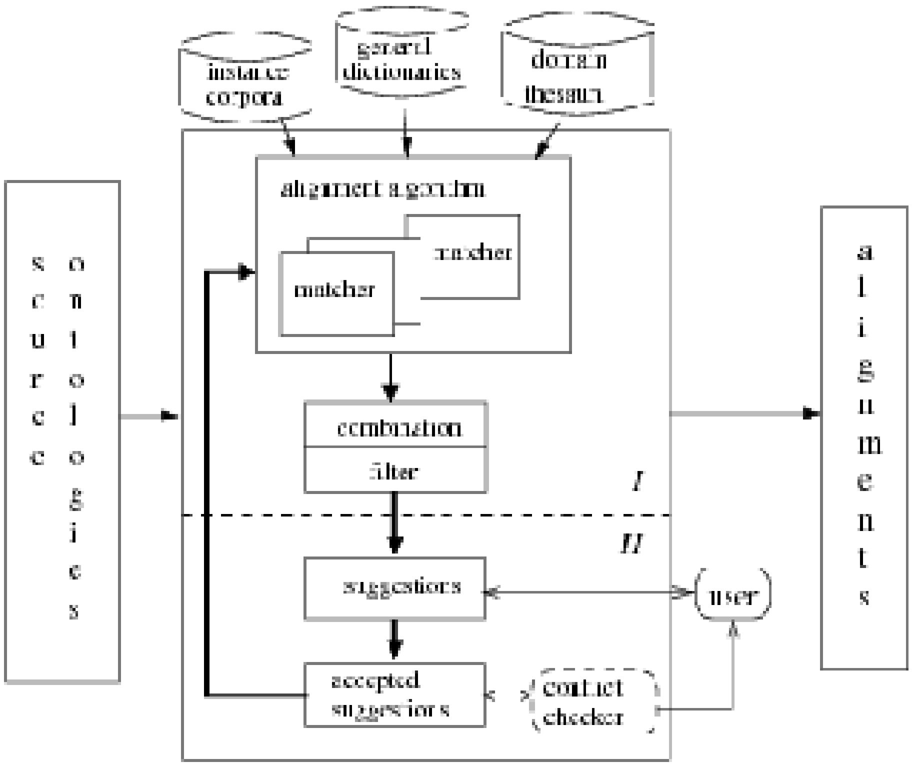
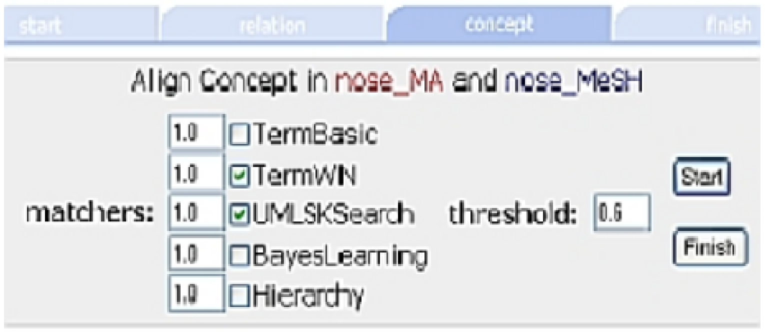


Figure 8: A general alignment strategy [5].

concepts. In the suggestion mode several kinds of matchers can be used and combined. The implemented matchers are a terminological matcher (TermBasic), the terminological matcher using WordNet (TermWN), a structure-based matcher (Hierarchy), a matcher (UMLSKSearch) using domain knowledge in the form of the Unified Medical Language System and an instance-based matcher (BayesLearning).

Figure 9 shows how different matchers can be chosen and weights can be assigned to these matchers. Filtering is performed using a threshold value. The pairs of terms with a similarity value equal to or above this value are shown to the user as alignment suggestions. An example alignment suggestion is given in figure 10. The system displays information (definition/identifier, synonyms, relations) about the source ontology terms in the suggestion. For each alignment suggestion the user can decide whether the terms are equivalent, whether there is an is-a relation between the terms, or whether the suggestion should be rejected. If the user decides that the terms are equivalent, a new name for the term can be given as well. Upon an action of the user, the suggestion list is updated. If the user rejects a suggestion where two different terms have the same name, she is required to rename at least one of the terms. At each point in time during the alignment process the user can view the ontologies represented in trees with the information on which actions have been performed, and she can check how many suggestions still need to be processed. Figure 11 shows the remaining suggestions for a particular alignment process. A similar list can be obtained to view the previously accepted alignment suggestions. In addition to the suggestion mode, the system also has a manual mode in which the user can view the ontologies and manually align terms (figure 12). The source ontologies are illustrated using is-a and part-of hierarchies (*i* and *p* icons, respectively). The user can choose terms from the ontologies and then specify an alignment operation. Previously aligned terms are identified by different icons. For instance, the *M* icons in front of 'nasal_cavity' in the two ontologies in figure 12 show that these were aligned using an equivalence relationship. There is also a search functionality to find specific terms more easily in the hierarchy. The suggestion and manual modes can be interleaved. The suggestion mode can also be repeated several times, and take into account the previously performed operations.



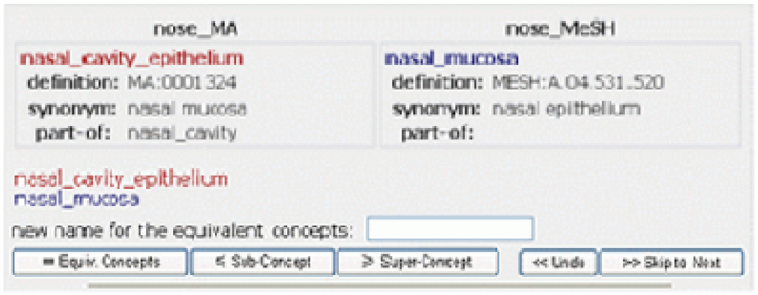
Align Concept in nose_MA and nose_MeSH

matchers:
1.0 ☐ TermBasic
1.0 ☑ TermWN
1.0 ☑ UMLSKSearch
1.0 ☐ BayesLearning
1.0 ☐ Hierarchy

threshold: 0.6

Figure 9: SAMBO - Combination and filtering [5].

After the user accomplishes the alignment process, the system receives the final alignment list and can be asked to create a new ontology. The system merges the terms in the alignment list, computes the consequences, makes the additional changes that follow from the operations, and finally copies the other terms to the new ontology. Furthermore, SAMBO uses a DIG

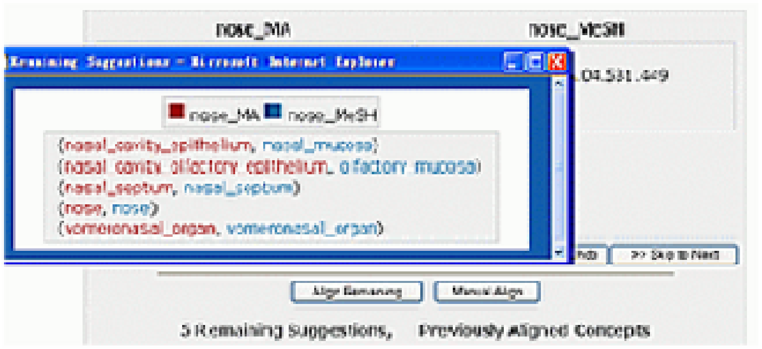


nasal_cavity_epithelium
definition: MA:0001324
synonym: nasal mucosa
part-of: nasal_cavity

nasal_mucosa
definition: MESH:A.04.531.520
synonym: nasal epithelium
part-of:

nasal_cavity_epithelium
nasal_mucosa
new name for the equivalent concepts:

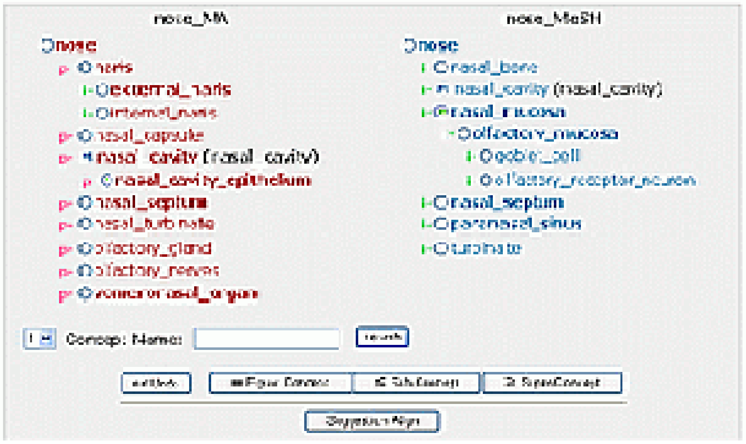
Figure 10: SAMBO - Alignment suggestion [5].



(nasal_cavity_epithelium, nasal_mucosa)
(nasal_cavity_olfactory_epithelium, olfactory_mucosa)
(nasal_septum, nasal_septum)
(nose, nose)
(vomeronasal_organ, vomeronasal_organ)

5 Remaining Suggestions, Previously Aligned Concepts

Figure 11: SAMBO - Information about the remaining suggestions [5].



nose
naris
external_naris
internal_naris
nasal_capsule
nasal_cavity (nasal_cavity)
nasal_cavity_epithelium
nasal_septum
nasal_turbinate
olfactory_gland
olfactory_nerves
vomeronasal_organ

nose
nasal_bone
nasal_cavity (nasal_cavity)
nasal_mucosa
olfactory_mucosa
goblet_cell
olfactory_receptor_neuron
nasal_septum
paranasal_sinus
turbinate

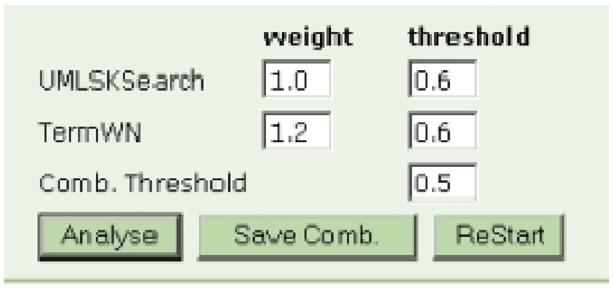
Figure 12: SAMBO - Manual mode [5].

description logic reasoner to provide a number of reasoning services. The user can ask the system whether the new ontology is consistent and can ask for information about unsatisfiable concepts and cycles in the ontology.

2.2.1 KitAMO - a tool for evaluating ontology alignment strategies

Currently, we do not have much knowledge about how well the different alignment strategies perform for different kinds of ontologies. Comparative evaluations of ontology alignment systems and algorithms have been performed by only some groups. It is realized that the study of the properties, and the evaluation and comparison of the alignment strategies and their combinations, would give us valuable insight in how the strategies could be used in the best way. We describe here a tool that allows us to do this. KitAMO[3] [6, 7] is a tool that provides an integrated system for comparative evaluation and analysis of alignment strategies and their combinations.

The current implementation of KitAMO focuses on the evaluation of matchers and implements a weighted sum as combination strategy and filtering based on a threshold value. The matchers are added to KitAMO as plug-ins.



	weight	threshold
UMLSKSearch	1.0	0.6
TermWN	1.2	0.6
Comb. Threshold		0.5

Figure 13: KitAMO - The weights and thresholds assignment [6].

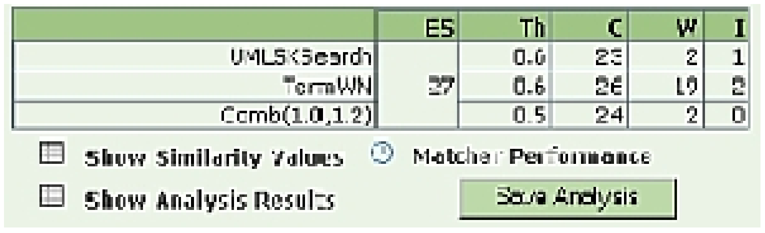
The user starts the evaluation process by choosing an evaluation case. Then the user decides which matchers should be used in the evaluation from the list of matcher plug-ins configured in KitAMO. The selected matchers calculate similarity values between the terms in the chosen evaluation case, and the results are written to a database. For the combination each matcher can be assigned a weight (weight in figure 13). The similarity values generated by the combination, i.e. the weighted sum, can also be saved to the database by the user. For the filter the user can assign threshold values for individual matchers and the combination (threshold in figure 13).

Assuming we have chosen the *ear* case (a predefined test case using MeSH and the Adult Mouse Anatomical Dictionary) and the matchers TermWN and UMLSKSearch, we receive the results as in figure 14. It shows the number of expected alignments (ES), the thresholds (Th), the number of correct suggestions (C), the number of wrong suggestions (W) and the number of redundant (or inferred) suggestions (I). We can save the analysis results and then experiment with other combinations and thresholds. For instance, after experimenting with thresholds 0.4, 0.5, 0.6, 0.7 and 0.8 for the two individual matchers, and different weights for the combination for the threshold 0.5, we get the results shown in figure 15. The results can be sorted according

[3] http://www.ida.liu.se/∼iislab/projects/KitAMO/

to the different colums. This allows us to analyze and compare the different matchers and their combinations. To examine the matchers in more detail we can use the similarity table as in figure 16. Also the performance can be compared (figure 17).

For a practical example of the use of KitAMO and the kinds of analysis that can be performed, we refer to [6].



	ES	Th	C	W	I
UMLSKSearch		0.6	23	2	1
TermWN	27	0.6	26	19	2
Comb(1.0,1.2)		0.5	24	2	0

Figure 14: KitAMO - The analysis result [6].

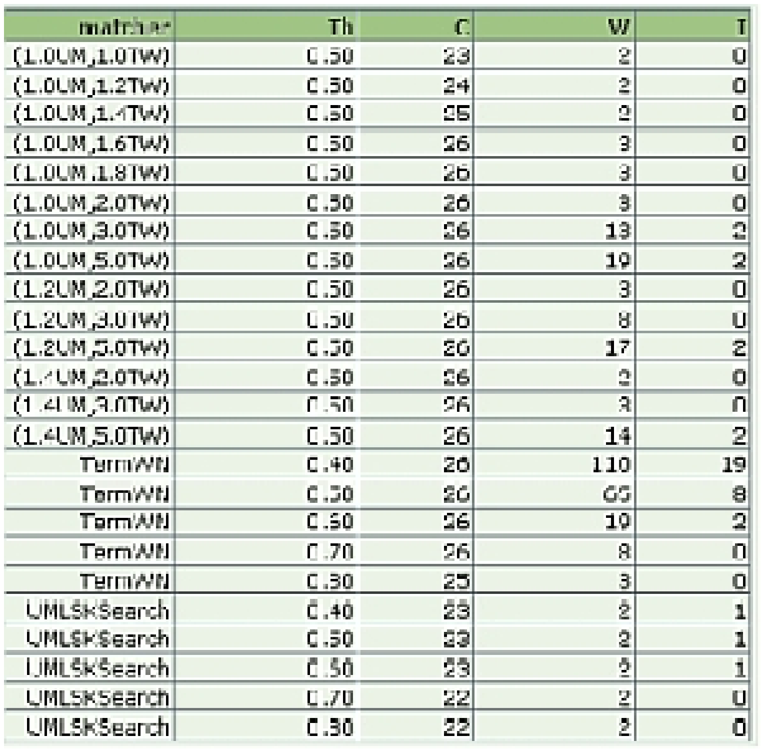
2.3 Evaluating grouping algorithms - KitEGA

During the last decade an enormous amount of biological data has been generated and techniques and tools to analyze this data have been developed. Many of these tools use some form of grouping and are used in, for instance, data integration, data cleaning, prediction of protein functionality, and correlation of genes based on microarray data. A number of aspects influence the quality of the grouping results: the data sources, the grouping attributes and the algorithms implementing the grouping procedure. Many methods exist, but it is often not clear which methods perform best for which grouping tasks. The study of the properties, and the evaluation and the comparison of the different aspects that influence the quality of the grouping results, would give us valuable insight in how the grouping procedures could be used in the best way. It would also lead to recommendations on how to improve the current procedures and develop new procedures. To be able to perform such studies and evaluations we need environments that support us in comparing and evaluating different grouping strategies for different grouping tasks on different data sets. In this section we present such an environment developed at Linköpings universitet, KitEGA[4]. It is based on the method described in [2].

KitEGA receives as an input a set of components (plug-ins) that define the grouping procedures that we want to evaluate. The user starts the evaluation process by choosing a number of parameters specifying a test case (figure 18). In the current implementation she selects a data source, a grouping rule (defining when two entities are similar), a grouping method (defining how to group entities into groups), evaluation methods and a classification source (defining the given classes of entities). The content of the user interface in figure 18 is generated dynamically based on the configuration file specifying the plug-ins that were made available to the system. Further, the user specifies attributes and the maximum size of the data values for the attributes which should be presented in the results (figure 19).

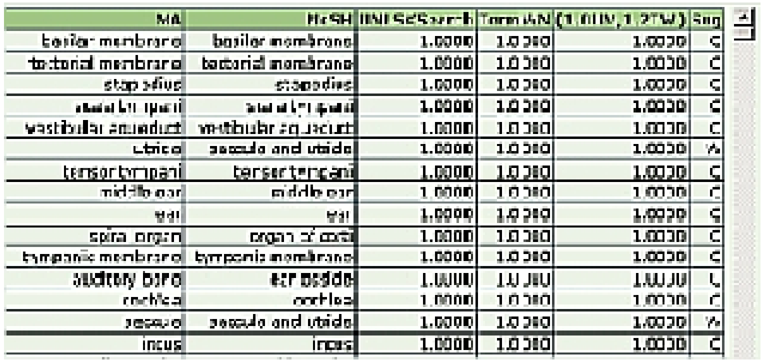
KitEGA will then run the test case and present the results to the user. Figure 20 shows the main form presenting grouping and evaluation results for a selected test case. The form shows the data entries included in each group together with information about the class they belong to

[4]http://www.ida.liu.se/∼iislab/projects/KitEGA/



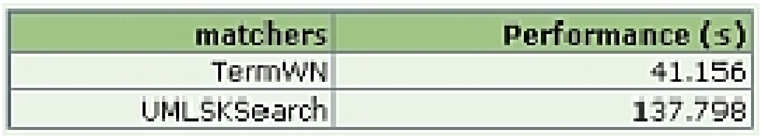
matcher	Th	C	W	T
(1.0UM,1.0TW)	0.50	23	2	0
(1.0UM,1.2TW)	0.50	24	2	0
(1.0UM,1.4TW)	0.50	25	2	0
(1.0UM,1.6TW)	0.50	26	3	0
(1.0UM,1.8TW)	0.50	26	3	0
(1.0UM,2.0TW)	0.50	26	3	0
(1.0UM,3.0TW)	0.50	26	13	2
(1.0UM,5.0TW)	0.50	26	19	2
(1.2UM,2.0TW)	0.50	26	3	0
(1.2UM,3.0TW)	0.50	26	8	0
(1.2UM,5.0TW)	0.50	26	17	2
(1.4UM,2.0TW)	0.50	26	2	0
(1.4UM,3.0TW)	0.50	26	3	0
(1.4UM,5.0TW)	0.50	26	14	2
TermWN	0.40	26	110	19
TermWN	0.50	26	65	8
TermWN	0.60	26	19	2
TermWN	0.70	26	8	0
TermWN	0.80	25	3	0
UMLSKSearch	0.40	23	2	1
UMLSKSearch	0.50	23	2	1
UMLSKSearch	0.60	23	2	1
UMLSKSearch	0.70	22	2	0
UMLSKSearch	0.80	22	2	0

Figure 15: KitAMO - The analysis results for the ear case [6].



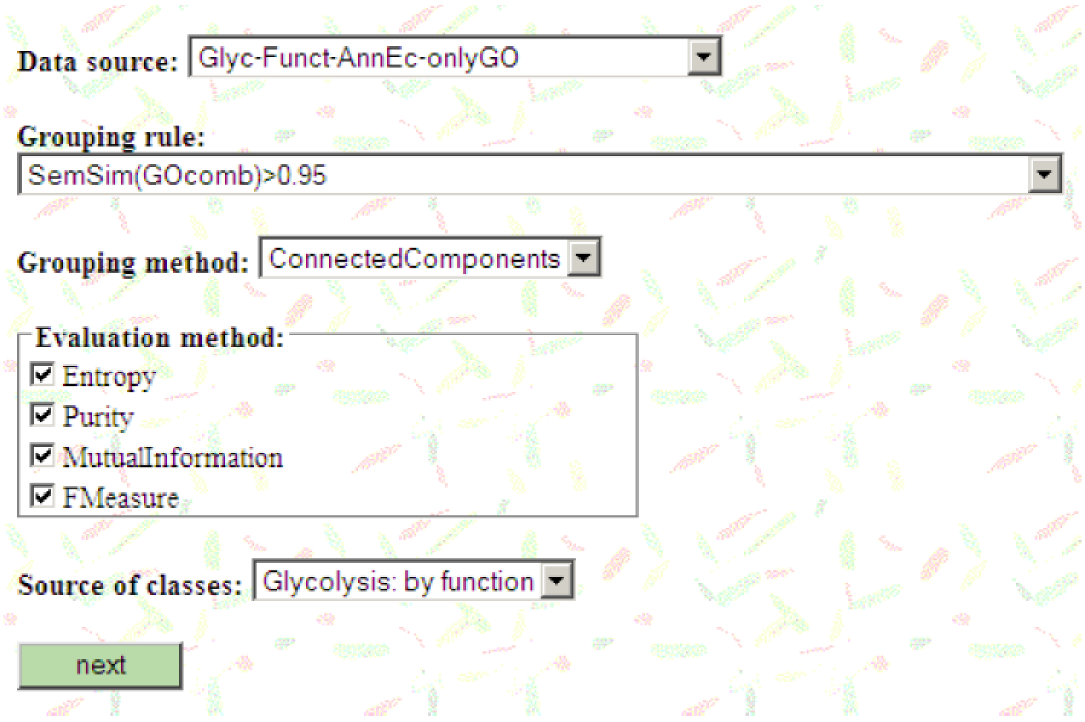
MA	MeSH	UMLSKSearch	TermWN	(1.0UM,1.2TW)	Sug
basilar membrane	basilar membrane	1.0000	1.0000	1.0000	C
tectorial membrane	tectorial membrane	1.0000	1.0000	1.0000	C
stapedius	stapedius	1.0000	1.0000	1.0000	C
scala tympani	scala tympani	1.0000	1.0000	1.0000	C
vestibular aqueduct	vestibular aqueduct	1.0000	1.0000	1.0000	C
utricle	saccule and utricle	1.0000	1.0000	1.0000	W
tensor tympani	tensor tympani	1.0000	1.0000	1.0000	C
middle ear	middle ear	1.0000	1.0000	1.0000	C
ear	ear	1.0000	1.0000	1.0000	C
spiral organ	organ of corti	1.0000	1.0000	1.0000	C
tympanic membrane	tympanic membrane	1.0000	1.0000	1.0000	C
auditory bone	ear ossicle	1.0000	1.0000	1.0000	C
cochlea	cochlea	1.0000	1.0000	1.0000	C
saccule	saccule and utricle	1.0000	1.0000	1.0000	W
incus	incus	1.0000	1.0000	1.0000	C

Figure 16: KitAMO - The similarity table [6].



matchers	Performance (s)
TermWN	41.156
UMLSKSearch	137.798

Figure 17: KitAMO - The performance table [6].



Data source: Glyc-Funct-AnnEc-onlyGO

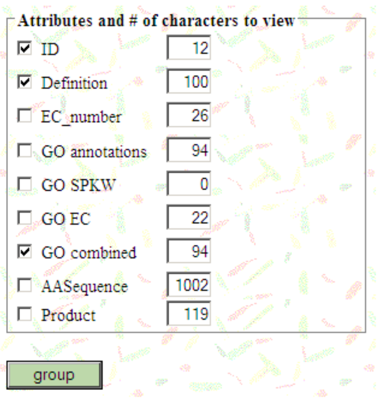
Grouping rule: SemSim(GOcomb)>0.95

Grouping method: ConnectedComponents

Evaluation method:
- ☑ Entropy
- ☑ Purity
- ☑ MutualInformation
- ☑ FMeasure

Source of classes: Glycolysis: by function

Figure 18: KitEGA - Specification of a test case.


Attributes and # of characters to view

☑ ID	12
☑ Definition	100
☐ EC_number	26
☐ GO annotations	94
☐ GO SPKW	0
☐ GO EC	22
☑ GO combined	94
☐ AASequence	1002
☐ Product	119

Figure 19: KitEGA - Customization of result representation for a test case.

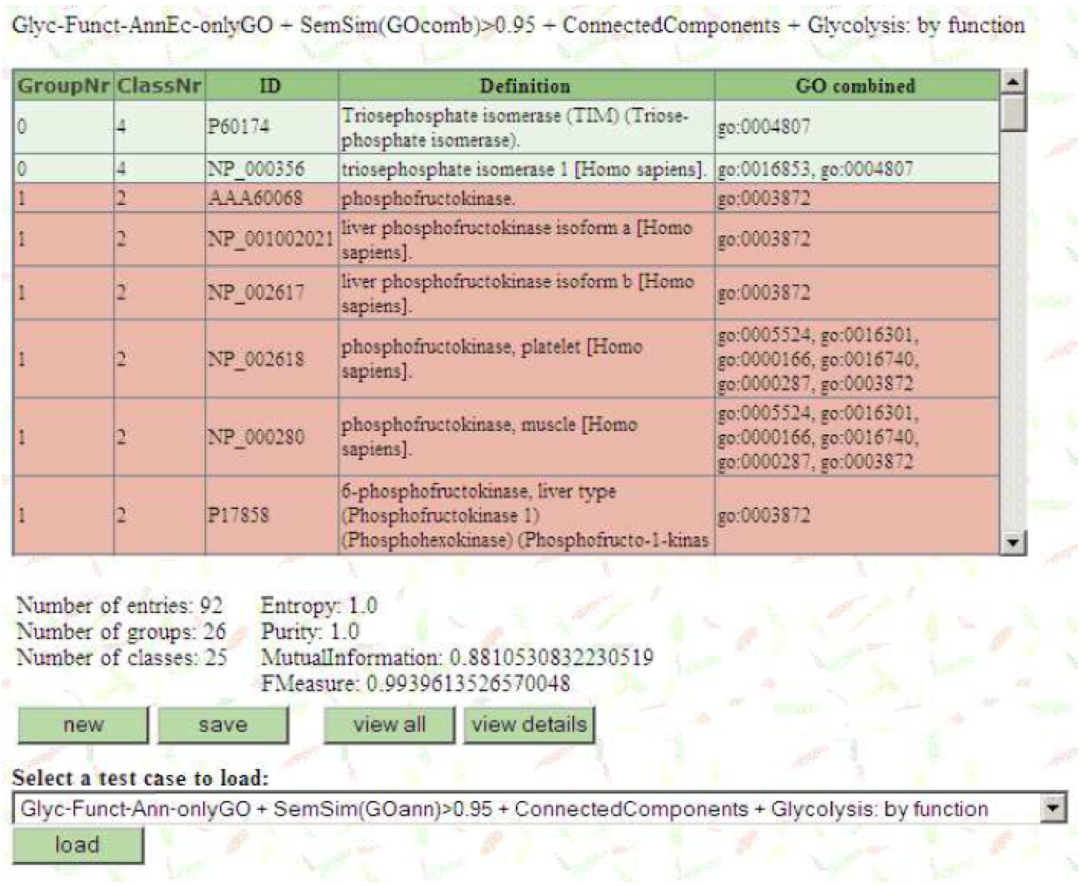
according to the previously selected classification. Further, the form presents the values of the computed evaluation measures together with some additional information about the test case, e.g. the total number of data entries in the data source and the total number of classes in the classification. The form provides support for starting new test cases, for saving test cases and their results and for loading previously saved test cases. In addition to the basic grouping result, the current KitEGA implementation supports several other forms giving different views on the data. To give a deeper insight into the grouping results, for a selected test case, KitEGA shows how the data entries in the generated groups are distributed among the classes in the selected classification (figure 21). In the figure a row represents a group while a column represents a class. The numbers in parentheses represent the total number of data entries in a group or a class, respectively. A cell stores data on the number of data entries that are true positives (they belong to the group and the class), false positives (they belong to the group, but not to the class) and false negatives (they belong to the class, but not to the group), respectively. Further, the system allows for a given group and class to view detailed information on the true positives, false positives and false negatives (e.g. figure 22). True positives are color-coded. To support comparative analysis of grouping procedures, the system also presents a form gathering evaluation results from different saved test cases (figure 23).

As a feasibility study we have used the test cases proposed in [2] and re-evaluated these using KitEGA [1]. We compared procedures for grouping different data sources by function or isozymes and discussed how to find the best procedure. The grouping procedures were also further analyzed to find the influence of different data sources, thresholds and grouping approaches.

References

[1] Jakonienė V, Lambrix P, 'A tool for similarity-based grouping of biological data', forthcoming.

[2] Jakonienė V, Rundqvist D, Lambrix P, 'A method for similarity-based grouping of biological data', *Proceedings of the 3rd International Workshop on Data Integration in the Life Sciences - DILS06*, LNBI 4075, pp 136-151, 2006.


Glyc-Funct-AnnEc-onlyGO + SemSim(GOcomb)>0.95 + ConnectedComponents + Glycolysis: by function

GroupNr	ClassNr	ID	Definition	GO combined
0	4	P60174	Triosephosphate isomerase (TIM) (Triose-phosphate isomerase).	go:0004807
0	4	NP_000356	triosephosphate isomerase 1 [Homo sapiens].	go:0016853, go:0004807
1	2	AAA60068	phosphofructokinase.	go:0003872
1	2	NP_001002021	liver phosphofructokinase isoform a [Homo sapiens].	go:0003872
1	2	NP_002617	liver phosphofructokinase isoform b [Homo sapiens].	go:0003872
1	2	NP_002618	phosphofructokinase, platelet [Homo sapiens].	go:0005524, go:0016301, go:0000166, go:0016740, go:0000287, go:0003872
1	2	NP_000280	phosphofructokinase, muscle [Homo sapiens].	go:0005524, go:0016301, go:0000166, go:0016740, go:0000287, go:0003872
1	2	P17858	6-phosphofructokinase, liver type (Phosphofructokinase 1) (Phosphohexokinase) (Phosphofructo-1-kinas	go:0003872

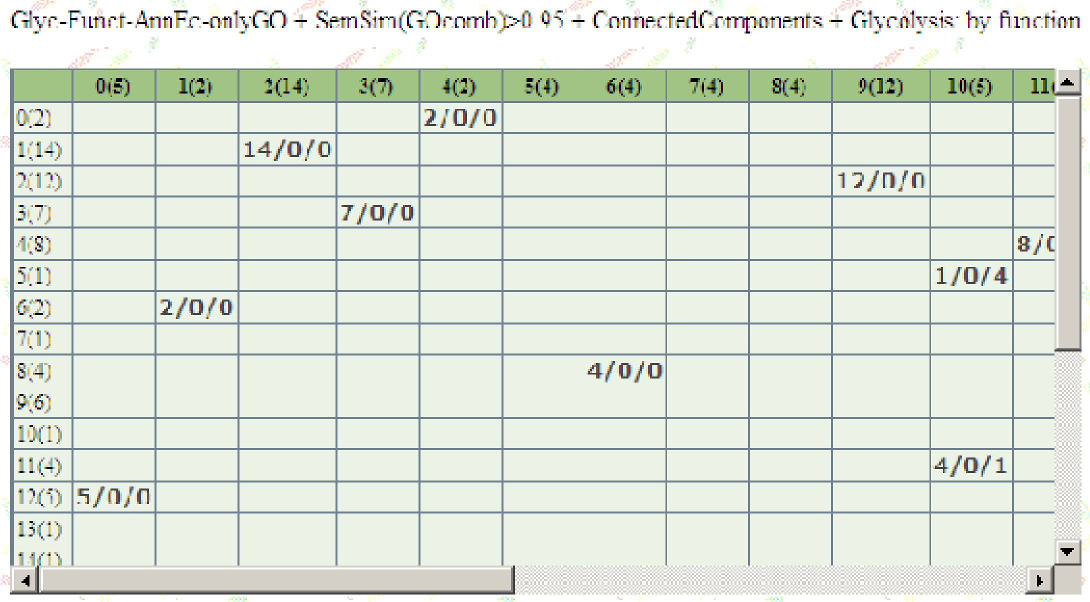
Number of entries: 92
Number of groups: 26
Number of classes: 25

Entropy: 1.0
Purity: 1.0
MutualInformation: 0.8810530832230519
FMeasure: 0.9939613526570048

Select a test case to load:

Glyc-Funct-Ann-onlyGO + SemSim(GOann)>0.95 + ConnectedComponents + Glycolysis: by function

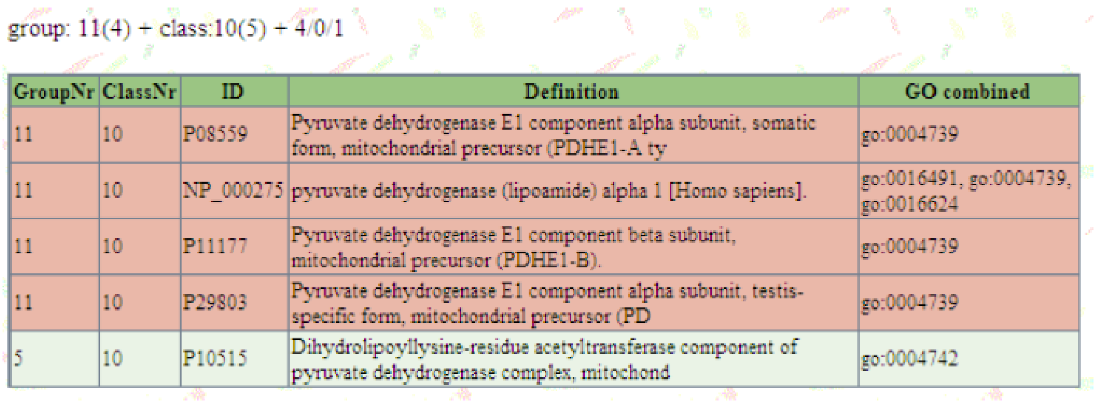
Figure 20: KitEGA - Grouping and evaluation results of a test case.



Glyc-Funct-AnnEc-onlyGO + SemSim(GOcomb)>0.95 + ConnectedComponents + Glycolysis: by function

	0(5)	1(2)	2(14)	3(7)	4(2)	5(4)	6(4)	7(4)	8(4)	9(12)	10(5)	11(
0(2)					2/0/0							
1(14)			14/0/0									
2(12)										12/0/0		
3(7)				7/0/0								
4(8)												8/
5(1)											1/0/4	
6(2)		2/0/0										
7(1)												
8(4)							4/0/0					
9(6)												
10(1)												
11(4)											4/0/1	
12(5)	5/0/0											
13(1)												
14(1)												

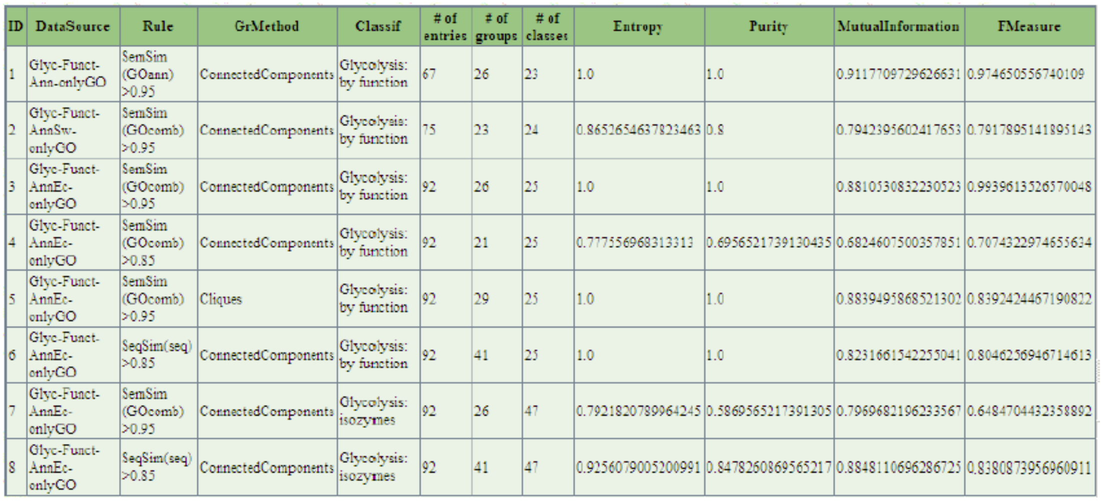
Figure 21: KitEGA - Detailed comparision of groups and classes. (Rows represent groups. Columns represent classes. Numbers in parentheses represent the total number of data entries in a group or a class. A cell stores data on true positives/false positives/false negatives.)



group: 11(4) + class:10(5) + 4/0/1

GroupNr	ClassNr	ID	Definition	GO combined
11	10	P08559	Pyruvate dehydrogenase E1 component alpha subunit, somatic form, mitochondrial precursor (PDHE1-A ty	go:0004739
11	10	NP_000275	pyruvate dehydrogenase (lipoamide) alpha 1 [Homo sapiens].	go:0016491, go:0004739, go:0016624
11	10	P11177	Pyruvate dehydrogenase E1 component beta subunit, mitochondrial precursor (PDHE1-B).	go:0004739
11	10	P29803	Pyruvate dehydrogenase E1 component alpha subunit, testis-specific form, mitochondrial precursor (PD	go:0004739
5	10	P10515	Dihydrolipoyllysine-residue acetyltransferase component of pyruvate dehydrogenase complex, mitochond	go:0004742

Figure 22: KitEGA - Details on true positives/false positives/false negatives for a given group and class.



ID	DataSource	Rule	GrMethod	Classif	# of entries	# of groups	# of classes	Entropy	Purity	MutualInformation	FMeasure
1	Glyc-Funct-Ann-onlyGO	SemSim (GOann) >0.95	ConnectedComponents	Glycolysis: by function	67	26	23	1.0	1.0	0.911770972962663l	0.974650556740109
2	Glyc-Funct-AnnSw-onlyGO	SemSim (GOcomb) >0.95	ConnectedComponents	Glycolysis: by function	75	23	24	0.8652654637823463	0.8	0.7942395602417653	0.7917895141895143
3	Glyc-Funct-AnnEc-onlyGO	SemSim (GOcomb) >0.95	ConnectedComponents	Glycolysis: by function	92	26	25	1.0	1.0	0.8810530832230523	0.9939613526570048
4	Glyc-Funct-AnnEc-onlyGO	SemSim (GOcomb) >0.85	ConnectedComponents	Glycolysis: by function	92	21	25	0.777556968313313	0.6956521739130435	0.6824607500357851	0.7074322974655634
5	Glyc-Funct-AnnEc-onlyGO	SemSim (GOcomb) >0.95	Cliques	Glycolysis: by function	92	29	25	1.0	1.0	0.8839495868521302	0.8392424467190822
6	Glyc-Funct-AnnEc-onlyGO	SeqSim(seq) >0.85	ConnectedComponents	Glycolysis: by function	92	41	25	1.0	1.0	0.8231661542255041	0.8046256946714613
7	Glyc-Funct-AnnEc-onlyGO	SemSim (GOcomb) >0.95	ConnectedComponents	Glycolysis: isozymes	92	26	47	0.7921820789964245	0.5869565217391305	0.7969682196233567	0.6484704432358892
8	Glyc-Funct-AnnEc-onlyGO	SeqSim(seq) >0.85	ConnectedComponents	Glycolysis: isozymes	92	41	47	0.9256079005200991	0.8478260869565217	0.8848110696286725	0.8380873956960911

Figure 23: KitEGA - Evaluation result for the saved test cases.

[3] Lambrix P, ‘Ontologies in Bioinformatics and Systems Biology’, chapter 8 in Dubitzky, Azuaje, (eds), *Artificial Intelligence Methods and Tools for Systems Biology*, pp 129-146, Springer, 2004.

[4] Lambrix P, ‘Towards a Semantic Web for Bioinformatics using Ontology-based Annotation’, *Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, pp 3-7, 2005. Invited talk.

[5] Lambrix P, Tan H, ‘SAMBO - A System for Aligning and Merging Biomedical Ontologies’, *Journal of Web Semantics, Special issue on Semantic Web for the Life Sciences*, 4(3):196-206, 2006.

[6] Lambrix P, Tan H, ‘A Tool for Evaluating Ontology Alignment Strategies’, *Journal on Data Semantics*, LNCS 4380, VIII:182-202, 2007.

[7] Lambrix P, Tan H, ‘Ontology alignment and merging’, chapter in Burger, Davidson, Baldock, (eds), *Anatomy Ontologies for Bioinformatics: Principles and Practice*, Springer, 2007. To appear.

[8] Lambrix P, Tan H, Jakonienė V, Strömbäck L, ‘Biological Ontologies’, chapter 4 in Baker, Cheung, (eds), *Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences*, pp 85-99, Springer, 2007.

[9] REWERSE, EU Network of Excellence on Reasoning on the Web with Rules and Semantics, Deliverables of the A2 Working Group. http://www.rewerse.net/

3 Using argumentation to tackle inconsistency and incompleteness in online distributed life science resources (Edinbrugh)

Kenneth McLeod, Albert Burger

3.1 introduction

The bioinformatics community has access to distributed resources, which are increasing rapidly, both in terms of quantity and size. Regardless of this growth, the data sets in these resources are often necessarily incomplete. There will always exist some time gap between the introduction of new experimental techniques and their exhaustive application, e.g. few gene expression databases cover entire genomes. Inconsistency between the resources is a further problem that must be overcome in order to make effective use of the data available. With these issues in mind, we suggest that bioinformatics provides a suitable domain for the application of non-monotonic reasoning, focusing in particular on argumentation. We propose the use of schemes to model the reasoning of an expert and use that reasoning to allow a system to evaluate the data presented to users. This will allow non-expert users to critically assess the wide array of data before further analysing it. Section 2 discusses the nature of distributed online resources available to life science researchers. Section 3 describes argumentation. Section 4 talks about the work we have done so far, before concluding in Section 5.

3.2 Bioinformatics

The 2006 annual review of databases in the sub domain of molecular biology [1] showed that 90 new databases were created, and 68 existing databases were significantly changed in 2005. This growth in online distributed databases illustrates the importance of such resources to the life science community. In addition to publishing data, the Internet provides access to tools that analyse that data, thus providing the mechanism to create new information. For example, GoPubMed [4] annotates literature made available through PubMed (www.pubmedcentral.nih.gov) with terms from the Gene Ontology (www.geneontology.org), an ontology published to provide a controlled vocabulary for describing genes and gene products in any organism. It is likely that the development of the semantic web and grid technologies will further encourage the use of services that use data published through the Internet. Increasing numbers of tools and repositories will have web/grid services alongside the more traditional data sharing techniques like FTP or Perl programmatic interfaces. The web services will be used as part of automated workflows. Projects like myGrid [5] have increased the ease of designing and running workflows, and consequently, the frequency with which this is done. Regardless of whether the online data resources are accessed by an html interface, web service, Perl or FTP, the user must take into account that these resources are not perfect. They feature inconsistent and incomplete information. For example, in the field of gene expression, which discovers the genes active (expressed) in the anatomical structures of organisms, few databases hold information on every gene in every structure of an organism. In order to reduce the gaps in knowledge, scientists continually research new methods and technologies to help them work faster and more accurately. If the latest technology is more sensitive than the previous one, it may detect genes in structures where they were not known to exist. At this point, there will be a need to repeat some of the experiments conducted with the old technology, and the gap in knowledge will have temporarily increased. In addition to incomplete information the researchers must also consider conflicting information. Two research groups may conduct similar experiments, but obtain different results and conclusions. This may be due to experimental error or simply a slight variation in experimental conditions. Despite the variations, both results will be published and entered into (possibly the same) online databases. Therefore the distributed online databases contradict each other, and themselves. For example, in the context of gene expression databases, such as GXD (www.informatics.jax.org), data is collected from various sources. The contributing studies may be based on different experimental setups, for example the use of different probes for the identification of genes. Such variation can lead to conflicting information; a recent version of GXD contains some 1300 instances of particular genes being reported as expressed and not expressed in the same mouse embryo tissue. For example, a GXD query on which genes are not expressed in the mouse brain will report, amongst others, a gene called Tenascin C, since one experiment reported this to be the case. Unless the user also executes a query that asks which genes are expressed in the brain, (s)he will not pick up that there are 14 experiments stored in GXD that report Tenascin C to be expressed in the brain. Clearly, any workflow that is solely based on the first query would be very suspect in its results. This is not an issue unique to GXD - we simply used it to illustrate the point - but one that needs to be considered when using other resources as well, including the mouse gene expression database EMAGE at the MRC (genex.hgu.mrc.ac.uk). In fact, resources such as GXD and EMAGE that cover the same biological domain (mouse gene expression in this case) provide a good basis for distributed argumentation systems in bioinformatics. In conclusion, there are a large and growing number of distributed resources available to the life science community, but there is a requirement to evaluate the output from a life science resource before using it, and a need for appropriate help to be given to the user when doing this evaluation.

3.3 Argumentation

We believe that an argumentation-based approach [3] could provide a solution to the problems described in Section 2. Before we describe our proposed solution, let us first briefly discuss argumentation. An argument [9] is a reason to believe something is true, it is used in dialogue to support or attack a conclusion. Arguments can also attack and defeat each other. Once defeated an argument can be reinstated if the argument that defeats it, is in turn defeated itself. When presented with the arguments for/against a conclusion, the user can evaluate the evidence and make a decision as to whether or not to believe it. As time passes, new information becomes available, and so new arguments can be created. These new arguments may defeat existing arguments, thus reinstating other arguments. When presented to the user, these changes may alter their perspective, and so alter their opinion of the conclusion. The reinstatement of arguments is the technique used to handle non-monotonicity, or defeasibility. This is the idea of assuming a conclusion is true until new evidence shows it is not. In computing terms, argumentation is often used to make computers argue, or to help them assist humans when arguing. By argue, we do not mean an uncontrolled dispute, but instead we are referring to controlled styles of dialogue similar to those used in legal debates. Argumentation has been successfully used in medical informatics, where it was tried as a method of overcoming the weaknesses of traditional mathematical and logical decision-making techniques [6]. Argumentation is used to help a doctor make a decision by providing a list of options and the arguments for and against each option (e.g. [7]. Often a recommendation is made, but the practitioner makes the final decision. Because it presents results in a manner humans find natural, argumentation has additionally been used to explain decisions to patients that were made by other means [11]. The attributes that make argumentation suitable for use in medical informatics also make it applicable to bioinformatics, where it has been employed to help a user evaluate the output of a single resource [8]. Considering the nature of bioinformatics resources, as discussed above, there is a clear intuitive argument to apply this technology not just to a single tool, but also to a range of distributed online tools and databases.

3.3.1 Argumentation: suggested idea and usage

Our proposal is to create a system that can analyse the output of any online resource that it has arguments for. Such a system can work with calls to a single resource, or calls to several resources featured in a workflow. For each query, the system will follow the same reasoning as a human expert in evaluating the resource's output, asking the same questions, and looking at similar metrics. The system will also compare data from different but related resources in order to provide arguments for/against each possible answer to the query. The process of creating arguments against a particular result would identify the most controversial results, allowing them to be brought to the user's attention. This would be of great assistance when looking for possible errors.

3.3.2 The creation of arguments

Our arguments capture the reasoning of human experts. The form of documentation we chose was argument schemes. On the one hand arguments are presented to users, but few in the life sciences have a background in formal logic or mathematics, so something more natural is required. On the other hand, our arguments have to be used in a real argumentation system, most of which have a logical basis. Thus pure natural language is inappropriate as is an overly theoretic solution. Argument schemes present the obvious compromise. A scheme is a natural language template for an argument that consists of two parts. The first is an inference rule comprising of a group of premises and a conclusion. The second part is a group of critical questions that allow an argument (i.e. instance of the scheme) to be challenged. Our schemes are based on the Expert Scheme created by [10]. This scheme models the reasoning that takes place when an expert witness is called in a legal case. The natural language inference suggests that when an expert voices an opinion, we should trust it because they are experts. The critical questions document the most common lines of attack that the opposition would use, for example asking the expert to provide evidence to support their opinion. For us, the notion of an expert is replaced by the notion of an online resource, such as GXD. We keep the basic presumption that the data held in the resource is accurate, and so we should believe it. Interviewing biologists, and asking them to explain their reasoning processes created our critical questions. As we produce more schemes, we insert them into a hierarchy. The top of our hierarchy is a scheme applicable to any resource with general questions like: are you using the latest data set for the resource? Underneath that we have schemes for groups of resources like gene expression databases. The critical questions are now less general, for example: Does the gene expression result have an associated image of the result? Schemes from that level are then specialised to make them appropriate for individual resources in that field, so the EMAGE mouse gene expression database has a scheme featuring the question: Did the EMAGE editor feel confident enough in the result to award it 3 stars? This makes reference to the scoring system that EMAGE editors use to indicate how confident they are in the result. This star system is not used throughout the domain, so an alternative resource like GXD would not have this question in their scheme. Sometimes resources have more than one scheme associated with them, e.g. the NCBI's BLAST (http://130.14.29.110/BLAST/index.shtml). BLAST is an algorithm, implemented in a number of tools, which allow researchers to compare gene/protein sequences to try and find similar genes/proteins, often in different organisms. The results from BLAST do not actually state whether or not BLAST believes proteins to be similar, but instead provides the user with a number of metrics they can use to make the decision. Most researchers will group the results into one of three categories: similar proteins, no similarity between proteins, and proteins that might be similar. One scheme can determine which group each result is entered into. However, some researchers will wish to examine the results in the third group to determine if similarity does exist. A second scheme is required for this. In addition to extra schemes for a resource, there are links between schemes of different resources. For example, the second BLAST scheme has a question that asks what the output of Pfam (www.sanger.ac.uk/Software/Pfam) shows. Pfam is a separate online biological resource and so has its own scheme. It is possible to turn the schemes into inference rules and thus use them in most logic-based argumentation systems.

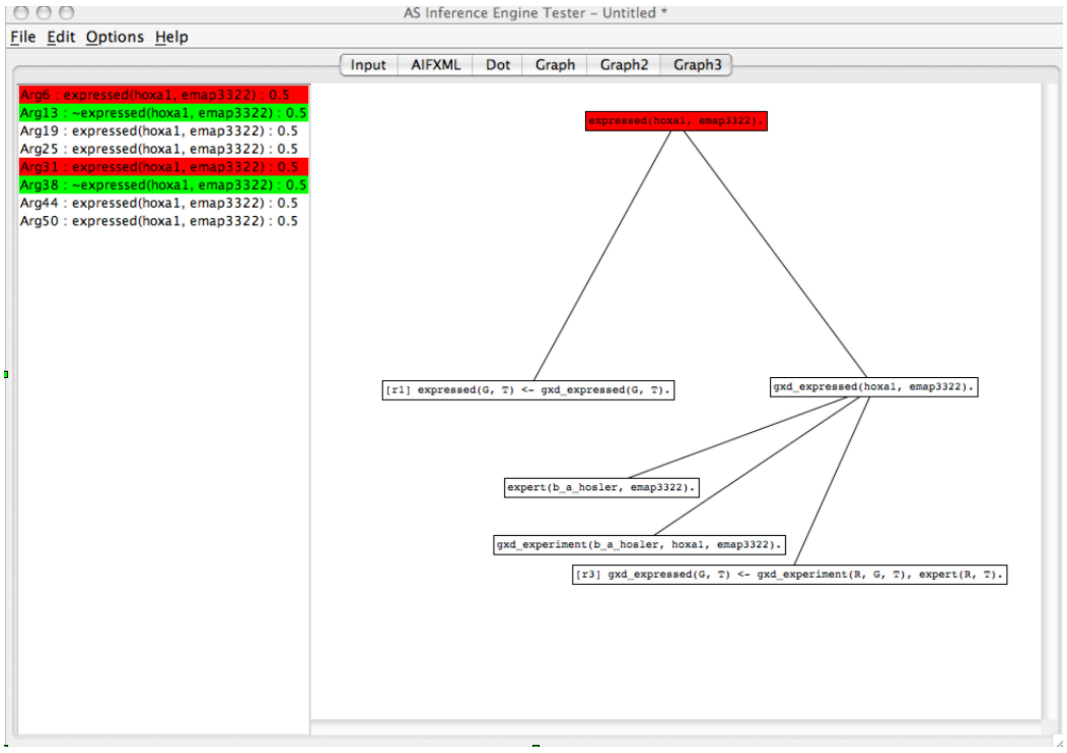
3.3.3 Application of schemes

As mentioned above in Section 4.1, schemes can be turned into inference rules for use in most argumentation systems. One argumentation engine that makes such a requirement is the engine produced by the ASPIC project. The Argumentation Services Platform with Integrated Components (ASPIC - www.argumentation.org) is an E.U. funded project that set out to develop re-usable components for argument-based interactions such as agent negotiation. So far, amongst other things, this project has produced a prototype argumentation engine in Java, based on theory created for and published by the project members [2]. This argumentation engine performs the tasks of: creating arguments; deciding which arguments attack and defeat each other; and then calculates the set of justified (correct) arguments.

In order for the argumentation engine to function, the user needs to supply a knowledgebase containing inference rules and facts, and with a query. The query asks the engine to check if something is true. It will be true, if an argument for it is justified. This means that the engine must create the arguments for and against the user's query, before balancing them out and deciding if the query is justified. The user can alter a number of parameters to change the behaviour of the engine, including changing the engine from skeptical to credulous. We are currently experimenting with this argumentation engine and our schemes. By adding real data from online resources such as GXD or EMAGE we can build arguments for and against each experimental result contained in the resource. For example, Figure24 shows the development of an argument for the gene Hoxa1 being expressed in the Mouse tissue EMAP:3222 (Mouse Embryo Stage 19). The system also balances the arguments to determine that the gene is not expressed in that particular tissue. The argumentation engine and its user interface are still in the prototype phase, so a full evaluation of this system is still to be completed.

3.4 Conclusion

The online resources available to the life science community contain incomplete, inconsistent and incorrect information. Therefore the data provided by these resources cannot be taken at



```
Arg6 : expressed(hoxa1, emap3322) : 0.5
Arg13 : ~expressed(hoxa1, emap3322) : 0.5
Arg19 : expressed(hoxa1, emap3322) : 0.5
Arg25 : expressed(hoxa1, emap3322) : 0.5
Arg31 : expressed(hoxa1, emap3322) : 0.5
Arg38 : ~expressed(hoxa1, emap3322) : 0.5
Arg44 : expressed(hoxa1, emap3322) : 0.5
Arg50 : expressed(hoxa1, emap3322) : 0.5
```

```
expressed(hoxa1, emap3322).
[r1] expressed(G, T) <- gxd_expressed(G, T).
gxd_expressed(hoxa1, emap3322).
expert(b_a_hosler, emap3322).
gxd_experiment(b_a_hosler, hoxa1, emap3322).
[r3] gxd_expressed(G, T) <- gxd_experiment(R, G, T), expert(R, T).
```

Figure 24: Screen shot of ASPIC Argumentation Engine

face value. This is important not just when the results are used directly, but also when they are combined into an automated workflow, where the incorrect data from one resource may mean that all subsequent resource queries are misleading. We have suggested that one form of non-monotonic reasoning may provide a suitable solution to this problem. For each resource query, argumentation could create arguments for/against every possible result, thus enabling the user to make an informed decision as to which result is correct. Our initial experiments in this area, using the ASPIC argumentation engine, suggest that this is potentially a hugely beneficial area of research in bioinformatics. It also suggests that the use of so-called argumentation schemes will be the most acceptable form of argumentation to biologists, as it not only provides a reasonable logic-based foundation, but also is easily understood by scientists without a detailed knowledge of logic theory. However, many questions remain to be investigated, such as a more formal assessment of usability and issues of scalability in the context of worldwide resources on the Internet.

References

[1] Bateman et al.: Editorial Nucleic Acids Research **34**

[2] Caminada M. W. A., Amgoud L.: An Axiomatic Account of Formal Argumentation Proceedings of the 20th National Conference on Artificial Intelligence,Pittsburgh, USA, pp. 608-613

[3] Carbogim D. V. et al: Argument-based application to knowledge engineering Knowledge Engineering Review **15** 2002 119–149

[4] Doms A., Schroeder M.: GoPubMed: Exploring PubMed with the GeneOntology Nucleic Acids Research,**33** 783–786

[5] Goble C. et al: The myGrid project: services, architecture and demonstrator Proceeding of UK e-Science All Hands Meeting, Nottingham, UK, 959–603

[6] Glasspool D. W., Fox J.: Knowledge, argument and meta-cognition in routine decision-making The routines of decision making, Lawrence Erlbaum, New Jersey, USA, 343–358

[7] Hurt C. et al Computerised advice on drug dosage decisions in childhood leukemia: a method and a safety strategy Artificial Intelligence in Medicine, Protaras, Cyprus, 158–162

[8] Jefferys B. R. et al: Capturing expert knowledge with argumentation: a case study in bioinformatic Bioinformatics, **22** 8 923–933

[9] Pollock J.: Defeasible reasoning with variable degrees of justification Artificial Intelligence, **13** 1-2 233–282

[10] Walton D.: Appeal to expert opinion : arguments from authority

[11] Williams M. and Williamson J.: Combining argumentation and bayesian nets for breast cancer prognosis Journal of Logic, Language and Information, 15 1-2 155–178

4 Efficient Constraint-based Sequence Alignment by Cluster Tree Elimination(Freiburg)

Sebastian Will, Anke Busch and Rolf Backofen

Aligning DNA and protein sequences has become a standard method in molecular biology. Often, it is desirable to include partial prior knowledge and conditions in an alignment. The most common and successful technique for efficient alignment algorithms is dynamic programming (DP). However, a weakness of DP is that one cannot include additional constraints without specifically tailoring a new DP algorithm. Here, we discuss a declarative approach that is based on constraint techniques and show how it can be extended by formulating additional knowledge as constraints. We take special care to obtain the efficiency of DP for sequence alignment. This is achieved by careful modeling and applying proper solving strategies.

4.1 Introduction

Modern molecular biology is not possible without tools for the comparison of the macromolecules DNA, RNA, and proteins.

It is most desirable to be able to specify additional restrictions for such similarity search whenever prior knowledge on the analyzed molecules is available. For example, consider the case of a biologist, who knows that certain regions in her sequences share a common local motif. Based on this knowledge, the rest of the sequences should be compared. Then, we need to optimize similarity under the additional constraint that parts of such regions should be matched to each other. Another striking example is the enhancement of RNA or protein comparison by employing knowledge on the structure of the macromolecules [10, 3, 1, 5].

However in general, similarity searching tools on the web do not allow to take such prior knowledge into account automatically. The reason for this deficiency is of algorithmic nature. Only for certain special constraints, alignment algorithms have been discussed. In particular, there are approaches that incorporate anchor constraints [7] and precedence constraints [8]. We will later discuss how such constraints fit into our newly introduced framework as simple cases. Aligning sequences and (to some extent) sequences with additional structural information is commonly and most successfully performed by *dynamic programming (DP)* [9, 11, 3]. There is no straightforward and general way to extend a DP algorithm in order to take additional knowledge into account.

To overcome this, declarative formulations of the alignment problem have been proposed. Due to their use of constraints, such approaches can be extended to incorporate prior knowledge. For this aim, such knowledge is formulated as constraints and added to the model for unconstrained alignment. One such previous approach [6] is based on *integer linear programming (ILP)*. Since in ILP one can only use boolean variables, the ILP model of [6] for aligning two sequences of length n and m introduces $O(nm)$ variables for modeling the alignment edges. Due to the resulting complexity, one needs to introduce artificial restrictions on the possible alignment edges for solving the problem in practice. Furthermore, the solving strategy for ILP does not achieve the efficiency of DP for the unconstrained case. Another declarative approach [12] is based on constraint programming. The approach introduces quadratically many variables and constraints and remodels the given DP algorithm. As a consequence, only a rather restricted class of side constraints can be handled efficiently.

Here, we introduce a new constraint-based approach. The main challenge that we face with our approach is to compete with the very good efficiency of DP in the standard case and allow extension by introducing new constraints.

We achieve the desired efficiency and adaption to additional constraints by modeling the alignment problem as a constraint optimization problem in the sense of [2, 4] and then applying a special solution strategy, which is known as *cluster tree elimination (CTE)* [4].

4.2 A Constraint Model for Sequence Alignment

We develop a constraint model for sequence alignment of two sequences $a = a_1 \dots a_n$ and $b = b_1 \dots b_m$ that are both words of the alphabet Σ. To be more precise, we define an *alignment* $\mathcal{A}$ *of* a *and* b as an ordered matching of positions in a and b, i.e. as a subset of $\{1, \dots, n\} \times \{1, \dots, m\}$ such that for all $(i, j), (i', j') \in \mathcal{A}$:

1. $i = i'$ if and only if $j = j'$ and
2. $i < i'$ implies $j < j'$.

We call i and j matched by $\mathcal{A}$ if and only if $(i, j) \in \mathcal{A}$.

The *score of an alignment* $\mathcal{A}$, which we want to maximize, depends on the *similarity function on positions* $\sigma : \{1, \dots, n\} \times \{1, \dots, m\} \to \mathbb{R}$ and *gap cost* $\gamma \in \mathbb{R}$. It is defined as

$$\text{score}(\mathcal{A}) = (n + m - 2|\mathcal{A}|)\gamma + \sum_{(i,j)\in\mathcal{A}} \sigma(i, j). \tag{1}$$

The classical DP algorithm for sequence alignment is specified via the recursion equation $D_{i\,j} = \max\{D_{i-1\,j-1} + \sigma(i, j), D_{i-1\,j} + \gamma, D_{i\,j-1} + \gamma\}$ with initialization $D_{0\,0} = 0$, $D_{i\,0} = i\gamma$, and $D_{0\,j} = j\gamma$ for $1 \leq i \leq n$ and $1 \leq j \leq m$ and solves the problem in $O(nm)$ time.

Here, we model alignment as a constraint optimization problem in the framework that is described in a more general form in [4]. There, one defines variables with finite domains and functions on these variables. In our special case, the solution of the problem is a valuation of the variables that maximizes the sum of the function values. Note that hard constraints c can be encoded in this framework by functions that yield $-\infty$ if the constraint is violated and 0 otherwise. Tacitly, our arithmetic is extended canonically in order to handle sums and maximizations involving infinity.

In our model, we represent alignments of a and b by finite domain variables X_i for $1 \leq i \leq n$ with domains $\text{dom}(X_i) = \{0, \dots, m\}$. Furthermore for technical reasons, we introduce the fixed variables $X_0 = 0$ and $X_{n+1} = m + 1$ and extend σ by defining $\sigma(n + 1, m + 1) = 0$. A given alignment $\mathcal{A}$ is uniquely encoded by a valuation $(X_0 = x_0, \dots, X_{n+1} = x_{n+1})$ of variables $X_0, \dots, X_{n+1}$ where 1.) $x_i = j$ if $(i, j) \in \mathcal{A}$ and 2.) $x_i = x_{i-1}$, for every i that is not matched in $\mathcal{A}$. Note that i and j are matched if and only if $x_i = j$ and $x_i > x_{i-1}$. For example, the valuation $\vec{x} = (0, 1, 2, 5, 6, 6, 6, 7, 8)$ of $X_0, \dots, X_8$ corresponds to the alignment {(1,1),(2,2),(3,5),(4,6),(7,7)}, which can be represented alternatively by

$$\begin{array}{l} a_1\, a_2 - - a_3\, a_4\, a_5\, a_6\, a_7 \\ b_1\, b_2\, b_3\, b_4\, b_5\, b_6 - - b_7 \end{array}.$$

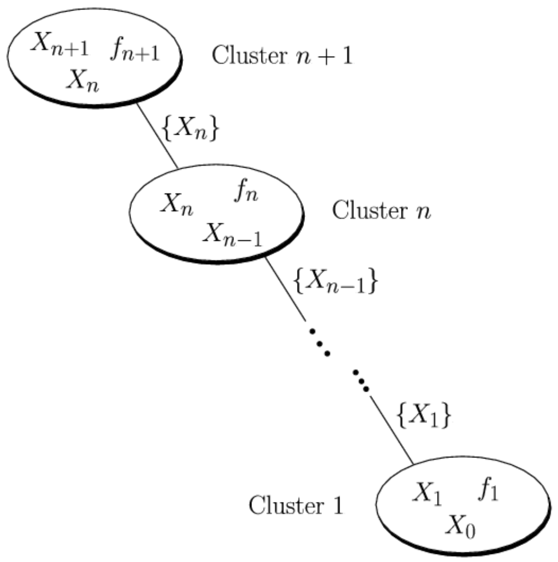


Figure 25: CTD of pure sequence alignment.

The only hard constraints on the variables X_i are $X_{i-1} \leq X_i$ for $1 \leq i \leq n+1$. They are modeled by functions

$$leq_i : \mathrm{dom}(X_{i-1}) \times \mathrm{dom}(X_i) \to \{-\infty, 0\}.$$

The scoring scheme is encoded via functions $f_i(X_{i-1}, X_i)$ for $1 \leq i \leq n+1$ that are defined by

$$f_i(j', j) = \begin{cases} \sigma(i,j) + (j - j' - 1)\gamma & \text{if } j' < j \\ \gamma & \text{otherwise.} \end{cases}$$

Note that we correctly model alignments and their scores. Firstly, a valuation $(X_0 = x_0, \dots, X_{n+1} = x_{n+1})$ represents an alignment $\mathcal{A}$ of a and b if and only if $\sum_{1 \leq i \leq n+1} f_i(x_{i-1}, x_i) + leq_i(x_{i-1}, x_i)$ is not $-\infty$. Secondly in this case, $\sum_{1 \leq i \leq n+1} f_i(x_{i-1}, x_i)$ equals the score of $\mathcal{A}$ (see Eq. 1).

4.3 Efficient Solving by Cluster Tree Elimination

Here, we sketch CTE and show its application to the our model. We demonstrate how direct application of CTE yields an $O(nm^2)$ algorithm. Then, by introducing modifications to the standard CTE approach, we improve the complexity to $O(nm)$ time.

For applying CTE, we first need a *cluster tree decomposition (CTD)* [4]. In such a decomposition, we distribute variables and functions to vertices (*clusters*) of a tree, such that 1.) each function occurs in exactly one cluster, 2.) if a function occurs in a cluster, then all variables of the function are assigned to the cluster as well, and 3.) for each variable the set of clusters that contain this variable induces a connected subtree.

Due to the definition, clusters that share variables are connected by edges. The shared variables are called *separator variables*. Figure 25 shows a cluster tree decomposition of our alignment model where edges are labeled by separator variables. We call the cluster consisting of X_{i-1}, X_i, f_i, and leq_i the cluster i. Note that in this figure (and the following ones) we omit the functions leq_i in our presentation.

CTE solves a constraint optimization problem by repeatedly exchanging messages between the clusters. The messages are functions that combine the functions of the cluster and marginalize them to the separator variables. Each message becomes a new function of the receiving cluster. From cluster i to cluster $i+1$, CTE sends a function g_i of the separator variable X_i. Beginning with cluster 1 it proceeds until cluster $n+1$ receives its message g_n. When sending a message from cluster i, this cluster is already augmented by a function g_{i-1}. Finally, it can be shown that $\max_{1 \leq j \leq m} (g_n(j) + f_{n+1}(j, m+1))$, which is the marginalization of the functions in cluster $n+1$ to the empty set of variables, is the maximal alignment score.

It remains to show how the messages g_i are computed. Due to the CTE algorithm, the message g_i is defined for $0 \leq j \leq m$ as

$$g_i(j) = \max_{0 \leq j' \leq m} \left(g_{i-1}(j') + f_i(j', j) + leq_i(j', j)\right). \tag{2}$$

Clearly, the standard approach takes $O(m^2)$ time for computing the function g_i. Since $O(n)$ messages are sent until the final alignment score can be computed, this results in an $O(nm^2)$ algorithm. Thereby, we have shown that the direct application of CTE to our constraint model yields a polynomial algorithm for sequence alignment.

Improving complexity. The complexity can be improved further if we employ the internal structure of the functions g_{i-1}, f_i, and leq_i. For this reason, we rewrite Equation 2 by the semantics of leq_i and expand the definition of f_i.

$$g_i(j) = \max_{0 \leq j' \leq j} \left(g_{i-1}(j') + \begin{cases} \sigma(i,j) + (j - j' - 1)\gamma & \text{if } j' < j \\ \gamma & \text{otherwise} \end{cases} \right).$$

Now, we can resolve the case distinction of f_i and move the constant $\sigma(i,j)$ out of the maximization. Then,

$$g_i(j) = \max \begin{cases} \sigma(i,j) + \max\limits_{0 \leq j' < j} \left(g_{i-1}(j') + (j - j' - 1)\gamma\right) \\ g_{i-1}(j) + \gamma. \end{cases}$$

A helper function $g^{\mathrm{m}}(j) = \max_{0 \leq j' < j} (g_{i-1}(j') + (j - j' - 1)\gamma)$ can be defined recursively and then computed in $O(m)$ time by DP as

$$g^{\mathrm{m}}(0) = -\infty,\ g^{\mathrm{m}}(1) = g_{i-1}(0), \text{ and for } j > 1\ g^{\mathrm{m}}(j) = \max \begin{cases} g^{\mathrm{m}}(j-1) + \gamma \\ g_{i-1}(j-1). \end{cases}$$

In consequence, the total computation of g_i is done in $O(m)$. This results in an $O(nm)$ time algorithm for the computation of the alignment score.[5]

4.4 Extension of the Sequence Alignment Model

Recently discussed constrained alignment approaches handled constraints like anchor constraints and precedence constraints. Such constraints can be encoded in our model straightforwardly and are handled by restricting the domains of variables, which even increases the

[5]$O(m)$ space can be achieved by further modifications to CTE.

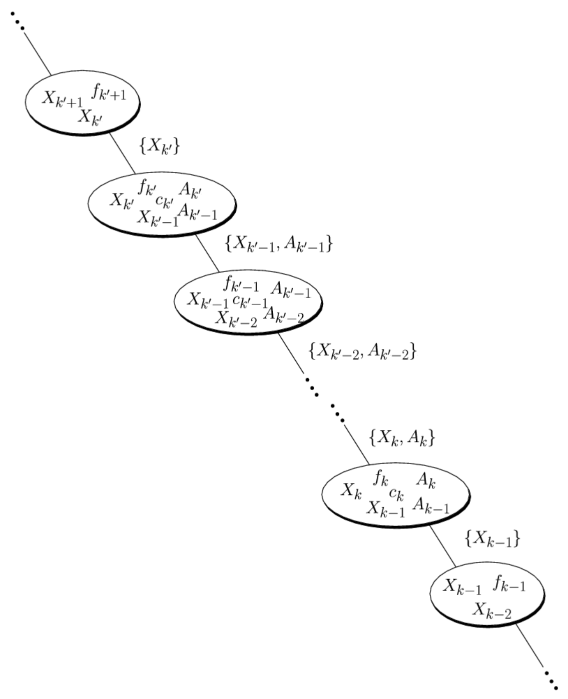

$X_{k'+1}$ $f_{k'+1}$ $X_{k'}$
$\{X_{k'}\}$
$f_{k'}$ $A_{k'}$ $X_{k'}$ $c_{k'}$ $X_{k'-1}$ $A_{k'-1}$
$\{X_{k'-1}, A_{k'-1}\}$
$f_{k'-1}$ $A_{k'-1}$ $X_{k'-1}$ $c_{k'-1}$ $X_{k'-2}$ $A_{k'-2}$
$\{X_{k'-2}, A_{k'-2}\}$
$\{X_k, A_k\}$
f_k A_k X_k c_k X_{k-1} A_{k-1}
$\{X_{k-1}\}$
X_{k-1} f_{k-1} X_{k-2}

Figure 26: CTD of an alignment with segment constraints.

efficiency of our algorithm. An *anchor constraints*, as discussed in [7], tells that position i in the first sequence can only be aligned to position j in the second sequence and furthermore, positions strictly left (resp. right) of i are aligned to positions strictly left (resp. right) of j. These conditions are expressed in our model by the constraints

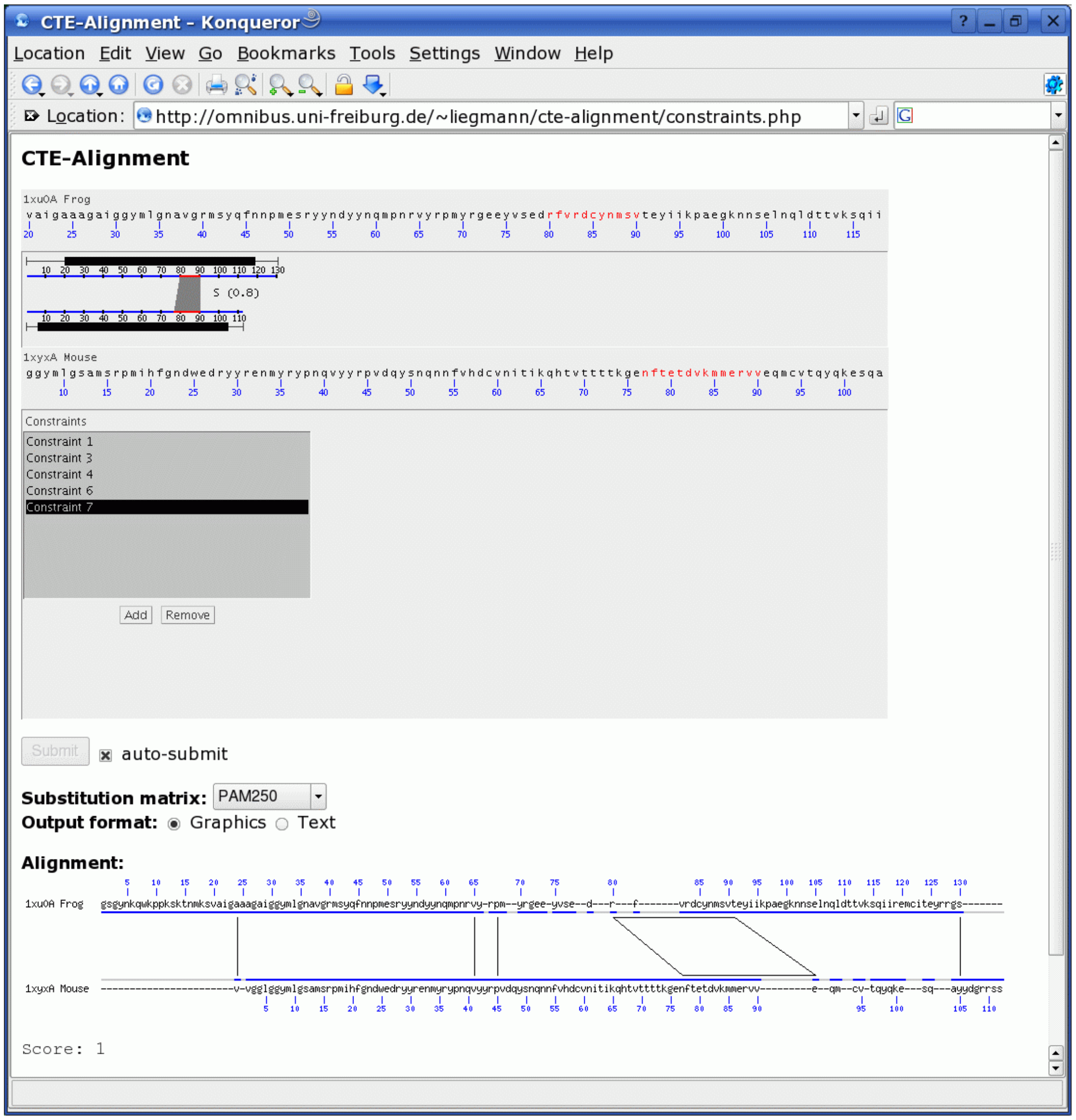
$$X_{i-1} < j,\ X_{i+1} > j, \text{ and } X_i = j \vee X_i = X_{i-1};$$

the latter implies $X_i \leq j$. The constraints are directly propagated to the domains of X_i and X_{i-1} and do not increase the complexity of our constraint problem. Via the less than constraints the new domain information is further propagated to the domains of all variables.

A *precedence constraint*, handled in [8], tells that in the alignment position i of the first sequence is left (resp. right) of position j of the second sequence. In our model, corresponding conditions are encoded as $X_i \geq j$ (resp. $X_i \leq j$). A further example for a trivial extension of the model is a condition like "position k is aligned to l or l'" (constraints: $X_k \in \{l, l'\}$ and $X_{k-1} < X_k$).

In this section, we discuss two more challenging extensions by example. Namely, the incorporation of prior knowledge on aligned segments and the extension to sequence structure alignment.

Aligned Segments As example of constraining the alignment between segments in a and b, we consider the constraint that at least x% of the positions $\{k, \ldots, k'\}$ in a have to be matched with positions $\{l, \ldots, l'\}$ in b. For extending our model by this constraint, we add variables



CTE-Alignment - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: http://omnibus.uni-freiburg.de/~liegmann/cte-alignment/constraints.php
S (0.8)
Add
Remove
Submit

CTE-Alignment

```
1xu0A Frog
vaigaaagaiggymlgnavgrmsyqfnnpmesryyndyynqmpnrvyrpmyrgeeyvsedrfvrdcynmsvteyiikpaegknnselnqldttvksqii

1xyxA Mouse
ggymlgsamsrpmihfgndwedryyrenmyrypnqvyyrpvdqysnqnnfvhdcvnitikqhtvttttkgenftetdvkmmervveqmcvtqyqkesqa
```

Constraints
- Constraint 1
- Constraint 3
- Constraint 4
- Constraint 6
- Constraint 7

[✓] auto-submit

Substitution matrix: PAM250
Output format: [✓ Graphics] Text

Alignment:

```
1xu0A Frog   gsgynkqwkppksktnmksvaigaaagaiggymlgnavgrmsyqfnnpmesryyndyynqmpnrvy-rpm--yrgee-yvse--d---r---f-------vrdcynmsvteyiikpaegknnselnqldttvksqiiremciteyrrgs-------

1xyxA Mouse  ----------------------v-vgglggymlgsamsrpmihfgndwedryyrenmyrypnqvyyrpvdqysnqnnfvhdcvnitikqhtvttttkgenftetdvkmmervv---------e--qm--cv-tqyqke---sq---ayydgrrss
```

Score: 1

Figure 27: Screen shot of the CTE Server

$A_{k-1}, \dots, A_{k'}$ and for each $k \leq i \leq k'$ the function $c_i(A_{i-1}, A_i, X_{i-1}, X_i)$ that encodes the hard constraint

$$A_i = A_{i-1} + \begin{cases} 1 & \text{if } X_{i-1} < X_i \text{ and } l \leq X_i \leq l' \\ 0 & \text{otherwise.} \end{cases}$$

We fix $A_{k-1} = 0$. Since the variables A_i count the proper matches in the prefix segment {k,...,i}, we can finally express the constraint by restricting the domains of A_i to $\{\max(0, \lceil \frac{x}{100}(k' - k + 1) \rceil - (k' - i)), \dots, i - k + 1\}$.

Figure 26 shows the cut-out of the CTD that is affected by the extension of the model. CTE works essentially as in the standard case. For $k \leq i \leq k'$, CTE sends messages g_i depending on the separator variables that each can be computed in $O(m\bar{k})$ time where $\bar{k} = k' - k + 1$. Thus, the total complexity is $O(m(n - \bar{k}) + m\bar{k}^2)$. Note that, as assumed in this result, one can transfer the complexity improvement of the previous section to this case of constrained alignment. It suffices to look at the message g_i from the cluster that contains a variable A_i (and thus contains the variables X_i, X_{i-1}, and A_{i-1} by construction). The message g_i, which depends on values for X_i and A_i, is given (already using the semantic of leq_i and c_i) as

$$g_i(j, a) = \max_{0 \leq j \leq j} \begin{cases} g_{i-1}(j', a - 1) + f_i(j', j) & \text{if } j' < j \text{ and } l \leq j \leq l' \\ g_{i-1}(j', a) + f_i(j', j) & \text{otherwise.} \end{cases}$$

One transforms further to

$$g_i(j, a) = \max \begin{cases} \sigma(i, j) + g^{\mathrm{m}}(j, a) \\ \gamma + g_{i-1}(j) \end{cases}$$

where we define

$$g^{\mathrm{m}}(j, a) = \max_{0 \leq j' < j} ((j - j' - 1)\gamma + g_{i-1}(j', a'))$$

where $a' = a$ if $l \leq j \leq l'$ and $a' = a - 1$ otherwise. Finally, g^{m} can be defined recursively as in the previous section as

$$g^{\mathrm{m}}(0, a) = -\infty, \; g^{\mathrm{m}}(1, a) = g(1, a'), \text{ and}$$

$$\text{for } j > 1 \; g^{\mathrm{m}}(j, a) = \max \begin{cases} g^{\mathrm{m}}(j - 1, a'') + \gamma \\ g_{i-1}(j - 1, a'') \end{cases}$$

where $a' = a$ if $l \leq 1 \leq l'$ and $a' = a - 1$ otherwise. Furthermore, $a'' = a$ if $l \leq j \leq l'$ and $a'' = a - 1$ otherwise.

We have demonstrated, that for this class of constraints the efficiency can be improved in the same way as in the case of unconstrained alignment. Intuitively, the additional constraints do not interfere with the nature of our score that enables the recursive decomposition.

Sequence Structure Alignment Here as additional input, we have two structures $P_a \subset \{1, \dots, n\} \times \{1, \dots, n\}$ and $P_b \subset \{1, \dots, m\} \times \{1, \dots, m\}$ and a function $\omega : \{1, \dots, n\} \times \{1, \dots, n\} \times \{1, \dots, m\} \times \{1, \dots, m\} \to \mathbb{R}$. A pair $(i_l, i_r) \in P_a$ (resp. $(j_l, j_r) \in P_b$) expresses a dependency, e.g. base pairing in RNA, between the positions i_l and i_r (resp. j_l and j_r). The function ω yields a score for aligning pairs of dependent positions.

The score of an alignment $\mathcal{A}$ is now defined in extension of Eq. 1 as

$$\text{score}(\mathcal{A}) + \sum_{\substack{(i_l,i_r)\in P_a,(j_l,j_r)\in P_b,\\(i_l,j_l)\in\mathcal{A},(i_r,j_r)\in\mathcal{A}}} w(i_l,i_r;j_l,j_r).$$

Our alignment model can be extended by adding for each $(i_l, i_r) \in P_a$ functions $h_{i_l i_r}(X_{i_l-1}, X_{i_l}, X_{i_r-1}, X_{i_r})$ that are defined as

$$h_{i_l i_r}(j'_l, j_l, j'_r, j_r) = \begin{cases} \omega(i_l, i_r; j_l, j_r) & \text{if } j'_l < j_l,\ j'_r < j_r, \text{ and } (j_l, j_r) \in P_b \\ 0 & \text{otherwise.} \end{cases}$$

Figure 28 provides an example for $P_a = \{(k_l, k_r), (l_l, l_r)\}$ and arbitrary P_b, which demonstrates the general construction principle of such a CTD. Due to the base pair (k_l, k_r) (and analogously for (l_l, l_r)), the decomposition contains a node consisting of the variables X_{k_l}, X_{k_r} and their predecessors X_{k_l-1}, X_{k_r-1}, since these variables depend on each other via the function $h_{k_l k_r}$. This node is parent of two sub-trees. In its left sub-tree, we handle the alignment for positions between k_l and k_r and in the right sub-tree the alignment for the positions less than k_l. Due to the conditions for a CTD, the variable X_{k_l} has to be shared with nodes of the left sub-tree, since it is constrained to variables in the leftmost leave.

In this tree structure, CTE begins with the leave vertices and proceeds to the root. From each cluster, it sends a message to its parent cluster. The final alignment score is obtained from the root node.

4.5 Conclusion

We present the first declarative approach to sequence alignment that is equally efficient as the commonly used method of dynamic programming. However, due to the declarative nature of the presented algorithm, it is extensible by additional constraints. This extensibility subsumes and goes beyond earlier constrained alignment approaches. Especially, we have shown how certain prior knowledge and structure information can be incorporated into the alignment model. By applying cluster tree elimination to the resulting extended alignment problem, we solve it efficiently. Finally, we have demonstrated for the alignment problem how CTE could profit from intelligent reasoning on the constraint model. Thereby, we hint at possible improvements of a current constraint solving strategy. A screen shot of the server is shown in 27.

References

[1] Rolf Backofen and Sebastian Will. Local sequence-structure motifs in RNA. *Journal of Bioinformatics and Computational Biology (JBCB)*, 2(4):681–698, 2004.

[2] Stefano Bistarelli, Ugo Montanari, and Francesca Rossi. Semiring-based constraint satisfaction and optimization. *Journal of the ACM*, 44(2):201–236, 1997.

[3] Tao Jiang, Guohui Lin, Bin Ma, and Kaizhong Zhang. A general edit distance between RNA structures. *Journal of Computational Biology*, 9(2):371–88, 2002.

[4] Kalev Kask, Rina Dechter, Javier Larrosa, and Avi Dechter. Unifying cluster-tree decompositions for reasoning in graphical models. *Artificial Intelligence*, 2005. forthcoming.

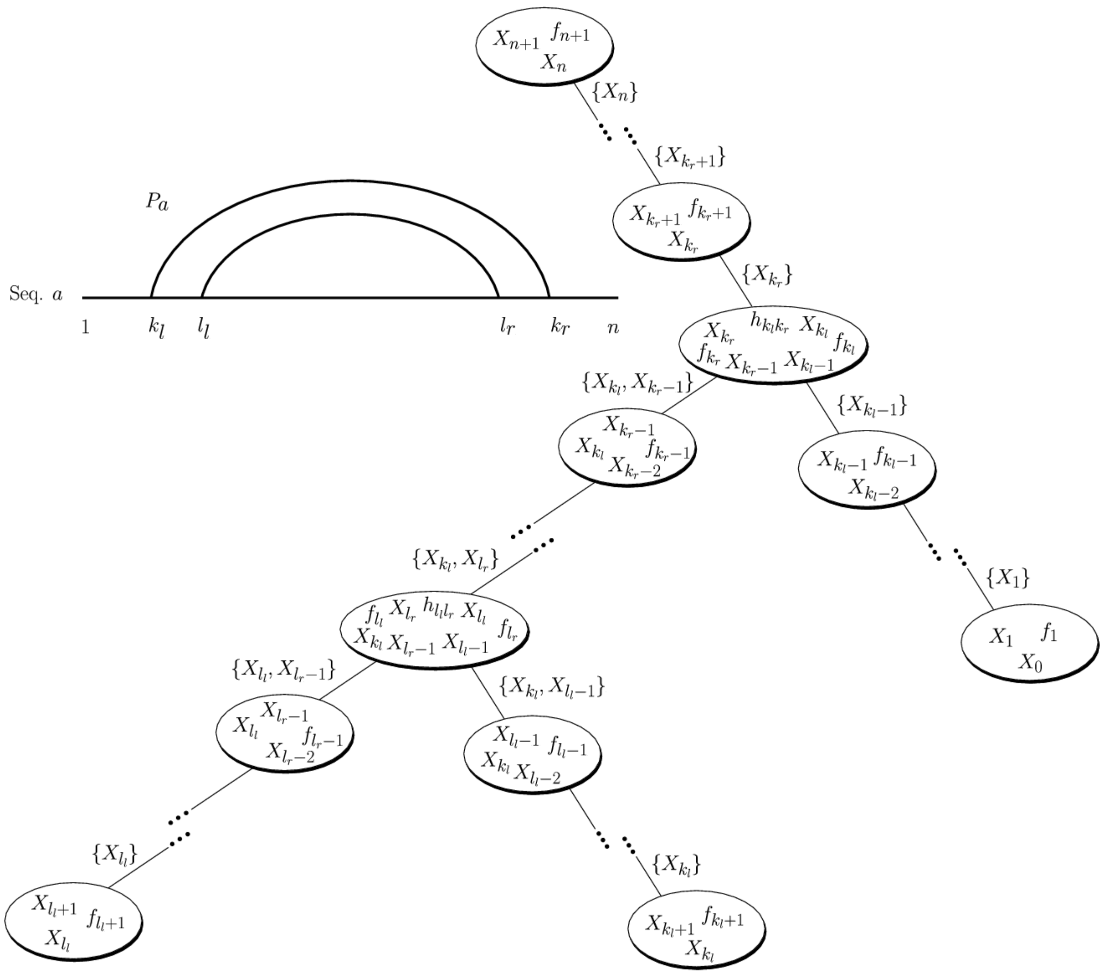

P_a
Seq. a
1
k_l
l_l
l_r
k_r
n
X_{n+1} f_{n+1} X_n
$\{X_n\}$
$\{X_{k_r+1}\}$
X_{k_r+1} f_{k_r+1} X_{k_r}
$\{X_{k_r}\}$
X_{k_r} $h_{k_lk_r}$ X_{k_l} f_{k_l} f_{k_r} X_{k_r-1} X_{k_l-1}
$\{X_{k_l}, X_{k_r-1}\}$
$\{X_{k_l-1}\}$
X_{k_r-1} X_{k_l} f_{k_r-1} X_{k_r-2}
X_{k_l-1} f_{k_l-1} X_{k_l-2}
$\{X_{k_l}, X_{l_r}\}$
$\{X_1\}$
f_{l_l} X_{l_r} $h_{l_ll_r}$ X_{l_l} f_{l_r} X_{k_l} X_{l_r-1} X_{l_l-1}
X_1 f_1 X_0
$\{X_{l_l}, X_{l_r-1}\}$
$\{X_{k_l}, X_{l_l-1}\}$
X_{l_r-1} X_{l_l} f_{l_r-1} X_{l_r-2}
X_{l_l-1} f_{l_l-1} X_{k_l} X_{l_l-2}
$\{X_{l_l}\}$
$\{X_{k_l}\}$
X_{l_l+1} f_{l_l+1} X_{l_l}
X_{k_l+1} f_{k_l+1} X_{k_l}

Figure 28: Example sequence structure alignment CTD (see text for details).

[5] Giuseppe Lancia, Robert Carr, Brian Walenz, and Sorin Istrail. 101 optimal PDB structure alignments: a branch-and-cut algorithm for the maximum contact map overlap problem. In *Proc. of the Fifth Annual International Conferences on Compututational Molecular Biology (RECOMB01)*. ACM Press, 2001.

[6] Hans-Peter Lenhof, Knuth Reinert, and Martin Vingron. A polyhedral approach to rna sequence structure alignment. In *Proc. of the Second Annual International Conferences on Compututational Molecular Biology (RECOMB98)*, volume 5, pages 517–30. ACM Press, 1998.

[7] Burkhard Morgenstern, Nadine Werner, Sonja J. Prohaska, Rasmus Steinkamp, Isabella Schneider, Amarendran R. Subramanian, Peter F. Stadler, and Jan Weyer-Menkhoff. Multiple sequence alignment with user-defined constraints at GOBICS. *Bioinformatics*, 21(7):1271–1273, 2005.

[8] Gene Myers, Sanford Selznick, Zheng Zhang, and Webb Miller. Progressive multiple alignment with constraints. In *Proceedings of the first annual international conference on Computational molecular biology (RECOMB 1997)*, pages 220–225, 1997.

[9] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–53, 1970.

[10] David Sankoff. Simultaneous solution of the RNA folding, alignment and protosequence problems. *SIAM J. Appl. Math.*, 45(5):810–825, 1985.

[11] T.F. Smith and M.S. Waterman. Comparison of biosequences. *Adv. appl. Math.*, 2:482–489, 1981.

[12] Roland H. C. Yap. Parametric sequence alignment with constraints. *Constraints*, 6(2/3):157–172, 2001.