VRIJE UNIVERSITEIT

Networks, modules and breeding schedules

Applications of Combinatorial Optimization to Computational Biology

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad Doctor
aan de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. F.A. van der Duyn Schouten,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de faculteit der Exacte Wetenschappen
op dinsdag 27 oktober 2015 om 13.45 uur
in de aula van de universiteit,
De Boelelaan 1105

door

Mohammed El-Kebir

geboren te Amsterdam

promotoren: prof.dr. J. Heringa
prof.dr. G.W. Klau

Contents

Chapter 1

Introduction

> Data, data everywhere but not a thought to think.
>
> Jesse H. Shera (1903-1982)

Technological advances have led to an unprecedented growth of biological data. The main challenge in the post-genomic era is to interpret and make sense out of these data and ultimately answer important biological questions ranging from determining the function and structure of proteins to elucidating the evolution of species and tumors [30, 71, 121]. Many of these questions can be formulated as *combinatorial optimization problems* where the goal is to find, given an objective function, an optimal object from a finite set of feasible objects [173]. Here, an optimal object has the minimum objective value in case of a minimization problem, or the maximum objective value in case of a maximization problem. Typically, the set of feasible objects grows exponentially in the size of the input—thus prohibiting an exhaustive enumeration. By carefully studying the combinatorial structure of the problem and patterns that are common to typical input data, one can often design an algorithm that performs well in practice. This thesis concerns several combinatorial optimization problems in computational biology for which we have developed algorithms that are of practical use. The approach that we have taken is depicted in Figure 1.1 and consists of the following steps.

1. *Formulating a combinatorial problem.* The first step is to phrase the biological question as a combinatorial optimization problem by defining the set of feasible solutions as well as an objective function that operates on this set. This is arguably the most difficult step as it is a very delicate process: There is a trade-off between including enough aspects of the question to arrive at a meaningful abstraction without getting lost in all the tiny details. Also, the biological optimum is often not well characterized and subject to interpretation [122].

   There are a lot of useful abstractions that are suitable blueprints for many biological problems, especially graph theoretical abstractions. For instance, the biological question of how similar two biological DNA sequences are, can be

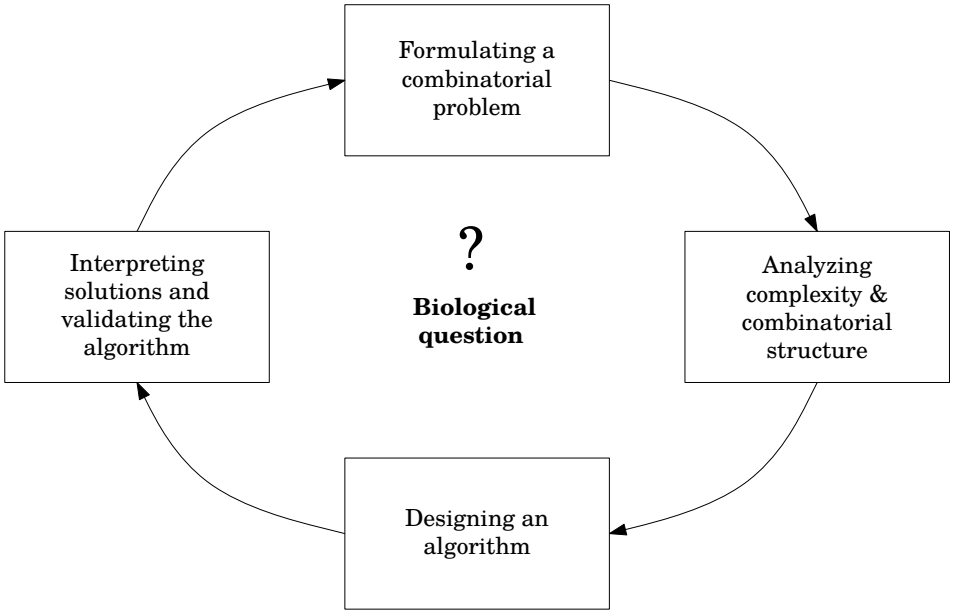


Figure 1.1: A scheme for solving biological problems using combinatorial optimization.

directly formulated as finding a longest path in a directed acyclic graph [92]. Typically, the problem statement will include a small twist that will make it slightly different from known combinatorial problems [160].

2. *Analyzing complexity and combinatorial structure.* Having arrived at a problem statement, the next step is to uncover parts of the combinatorial structure of the problem. That is, what properties do optimal solutions have? Can they be broken up in smaller suboptimal pieces? To answer these questions, the first thing that needs to be assessed is the hardness of the problem: Is it even possible to design an algorithm whose running time scales polynomially in the size of the input? The answer to this question is 'no' if we can show the problem to be NP-hard (assuming P $\neq$ NP). This corresponds to showing that all instances of another known hard problem can be transformed, in polynomial time, into instances of our own problem. Most biological problems are NP-hard, as are the problems considered in this thesis.

   At this stage we also try to uncover common patterns in typical biological input instances. For instance, graphs corresponding to molecules are typically planar, have low treewidth and bounded degree. Such properties will aid in our endeavor at arriving at an algorithm that works well for practical problem instances.

3. *Designing an algorithm.* Using the knowledge accumulated in the previous stages, the goal of this stage is to develop an algorithm for the problem. We mea-

sure the performance of an algorithm on two different scales: (1) the asymptotic running time and (2) the quality of the returned solution in terms of its objective value with respect to all other feasible solutions. Asymptotic running time relates the size of the input to the number of operations performed by the algorithm: the running time of a quadratic algorithm scales quadratically with the input size—a doubling of the input size will result in a running time that is four times longer. The second scale is about the quality of the returned solutions. Exact algorithms are guaranteed to return optimal solutions, i.e. the objective value of all feasible solutions is equal to or worse than the objective value of the returned solution. Heuristic algorithms come with no guarantees as to the optimality of the returned solutions.

Ideally, the algorithm we develop has a running time that scales polynomially in the size of the input and is guaranteed to return an optimal solution. If, however, in the previous stage it turned out that the problem is NP-hard, finding such an exact polynomial-time algorithm is highly unlikely. That means that we either settle for an exponential time algorithm, or give up on designing an algorithm that is guaranteed to find an optimal solution.

An exponential-time algorithm is not bad news per se. It may actually work well for most practical problem instances, especially when it exploits the properties identified in the previous stage. As mentioned before, an example of such a property is bounded degree in a graph. If we can theoretically show that the worst-case running time of the algorithm is polynomial in the input size but exponential in some parameter describing the input then the problem in question is fixed-parameter tractable. Combinatorial optimization techniques that we have used in this thesis include dynamic programming, (mixed) integer linear programming and Lagrangian relaxation. Dynamic programming is applicable if the problem manifests optimal substructure: An optimal solution can be constructed efficiently from optimal solutions of its subproblem. In mixed integer linear programming the set of feasible solutions is described by a set of linear inequalities on a space of both fractional and integer variables. Commercial solvers such as CPLEX achieve good performance in practice [106]. The starting point of Lagrangian relaxation is also an integer linear programming formulation. Instead of trying to solve this formulation, certain inequalities are relaxed such that the set of feasible solutions of the original problem is a subset of the set of feasible solutions of the relaxed problem [91]. The relaxed problem is easier to solve than the original problem. Violations of the relaxed inequalities are penalized with Lagrangian multipliers in the objective function. In case the original problem is a maximization (minimization) problem, the goal is to find a subset of multipliers that minimizes (maximizes) the relaxed objective function.

4. *Interpreting solutions and validating the algorithm.* The final step is to run the algorithm on biological input instances and to interpret the returned solutions with the aim of answering the original biological question. Information visualization techniques may help in the interpretation of the identified solutions.

The biological quality of the returned solutions can be assessed in several ways. One way is to re-evaluate the found solutions in terms of a different objective function that captures other aspects of the biological problem. For instance, for network alignment we use Gene Ontology terms [15] to assess how biologically similar pairs of aligned proteins are. For certain biological problems, benchmark instances have been compiled. These are instances that come with biologically verified solutions against which the solutions of the designed algorithm can be compared. When available, we can compare against other methods for the same biological problem in the benchmark.

There are two causes of low biological quality of returned solutions: (1) either the algorithm returns suboptimal solutions (with respect to the original objective function) or (2) the problem statement does not capture the biological problem adequately. We can rule out (1) if the algorithm in question is an exact algorithm. This is the main advantage that exact algorithms have over heuristic algorithms. To overcome cause (2), other aspects of the biological problem need to be included, which leads back to the first step of the cycle.

1.1 Outline

This thesis is split up in three parts: Networks, modules and breeding schedules. In Part I, we start by considering a biological problem rooted in comparative network analysis in Chapters 2 and 3. Here, the goal is to identify commonalities between biological networks from different strains or species, or derived from different conditions. We solve this problem using Lagrangian relaxation. In Chapter 4 we focus on the prediction of protein-protein interactions using the notion of coevolution: Evidence of coevolution of the protein families of two proteins may indicate an evolutionary preserved interaction between the two proteins. Interestingly, the same combinatorial problem formulation of Chapters 2 and 3 applies to this different biological problem.

The problems we consider in Part II concern the extraction of smaller connected subnetworks from a larger network. In Chapter 5, we consider the maximum-weight connected subgraph problem, which is a combinatorial formulation of the active module problem: Given differential expression data and a protein-protein interaction network, find a connected subnetwork that is significantly differentially expressed. For solving this problem, we use integer linear programming by applying a branch-and-cut scheme. To interpret identified active modules, we introduce a set-based visualization technique in Chapter 6. Chapter 7 generalizes the active module problem across species. We introduce the charge group partitioning in problem Chapter 8. This problem occurs in the automated parameterization of molecular compounds for use in molecular dynamics simulations. We exploit properties of practical input data, including bounded treewidth, and develop a dynamic programming based method.

Part III and Chapter 9 introduce the crossing schedule optimization problem, which, given a set of parental genotypes, asks for an efficient way of crossing these and their offspring with the goal of arriving at a specified desired genotype. After formally stating the problem and analyzing its complexity and combinatorial structure, we introduce a mixed integer linear programming formulation for solving it.

Part I

Networks

Chapter 2

Sparse global network alignment

Adapted from:

M. El-Kebir, J. Heringa, and G. W. Klau. Lagrangian relaxation applied to sparse global network alignment. In *Pattern Recognition in Bioinformatics, PRIB 2011, Delft, The Netherlands, November 2–4, 2011*, pages 225–236, 2011

Abstract

Data on molecular interactions is increasing at a tremendous pace, while the development of solid methods for analyzing this network data is lagging behind. This holds in particular for the field of comparative network analysis, where one wants to identify commonalities between biological networks. Since biological functionality primarily operates at the network level, there is a clear need for topology-aware comparison methods. In this paper we present a method for global network alignment that is fast and robust, and can flexibly deal with various scoring schemes taking both node-to-node correspondences as well as network topologies into account. It is based on an integer linear programming formulation, generalizing the well-studied quadratic assignment problem. We obtain strong upper and lower bounds for the problem by improving a Lagrangian relaxation approach and introduce the software tool NATALIE 2.0, a publicly available implementation of our method. In an extensive computational study on protein interaction networks for six different species, we find that our new method outperforms alternative state-of-the-art methods with respect to quality and running time.

2.1 Introduction

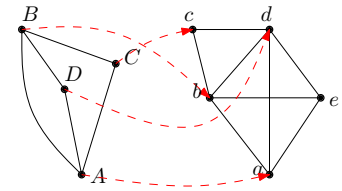
In the last decade, data on molecular interactions has increased at a tremendous pace. For instance, the STRING database [193], which contains protein protein interaction (PPI) data, grew from 261,033 proteins in 89 organisms in 2003 to 5,214,234 proteins in 1,133 organisms in May 2011, more than doubling the number of proteins in the database every two years. The same trends can be observed for other types of biological networks, including metabolic, gene-regulatory, signal transduction and

metagenomic networks, where the latter can incorporate the excretion and uptake of organic compounds through, for example, a microbial community [119, 177]. In addition to the plethora of experimentally derived network data for many species, also the structure and behavior of molecular networks have become intensively studied over the last few years [7], leading to the observation of many conserved features at the network level. However, the development of solid methods for analyzing network data is lagging behind, particularly in the field of comparative network analysis. Here, one wants to identify commonalities between biological networks from different strains or species, or derived form different conditions. Based on the assumption that evolutionary conservation implies functional significance, comparative approaches may help (i) improve the accuracy of data, (ii) generate, investigate, and validate hypotheses, and (iii) transfer functional annotations. Until recently, the most common way of comparing two networks has been to solely consider node-to-node correspondences, for example by finding homologous relationships between nodes (e.g. proteins in PPI networks) of either network, while the topology of the two networks has not been taken into account. Since biological functionality primarily operates at the network level, there is a clear need for topology-aware comparison methods. In this paper we present a network alignment method that is fast and robust, and can flexibly deal with various scoring schemes taking both node-to-node correspondences as well as network topologies into account.

Previous work. Network alignment establishes node correspondences based on both node-to-node similarities and conserved topological information. Similar to sequence alignment, *local* network alignment aims at identifying one or more shared subnetworks, whereas *global* network alignment addresses the overall comparison of the complete input networks.

Over the last years a number of methods have been proposed for both global and local network alignment, for example PathBlast [123], NetworkBlast [178], MaWISh [130], Graemlin [75], IsoRank [183], Graal [133], and SubMAP [18]. PathBlast heuristically computes high-scoring similar paths in two PPI networks. Detecting protein complexes has been addressed with NetworkBlast by Sharan et al. [178], where the authors introduce a probabilistic model and propose a heuristic greedy approach to search for shared complexes. Koyutürk et al. [130] use a more elaborate scoring scheme based on an evolutionary model to compute local pairwise alignments of PPI networks. The IsoRank algorithm by Singh et al. [183] approaches the global alignment problem by preferably matching nodes which have a similar neighborhood, which is elegantly solved as an eigenvalue problem. Kuchaiev et al. [133] take a similar approach. Their method Graal matches nodes that share a similar distribution of so-called graphlets, which are small connected non-isomorphic induced subgraphs.

In this paper we focus on pairwise global network alignment, where an alignment is scored by summing up individual scores of aligned node and interaction pairs. Among the above mentioned methods, IsoRank and Graal use a scoring model that can be expressed in this manner.



$$c(v_1, v_2) = \begin{cases} 1, & \text{if } v_1 = A \text{ and } v_2 = a, \\ 1, & \text{if } v_1 = B \text{ and } v_2 = b, \\ 1, & \text{if } v_1 = C \text{ and } v_2 = c, \\ 1, & \text{if } v_1 = D \text{ and } v_2 = d, \\ 0, & \text{otherwise.} \end{cases}$$

$$w(v_1, v_2, w_1, w_2) = \begin{cases} 10, & \text{if } (v_1, w_1) \in E_1 \\ & \text{and } (v_2, w_2) \in E_2, \\ 0, & \text{otherwise.} \end{cases}$$

Figure 2.1: Example of a network alignment. With the given scoring function, the alignment has a score of $4 + 40 = 44$.

Contribution. We present an algorithm for global network alignment based on an integer linear programming (ILP) formulation, generalizing the well-studied quadratic assignment problem (QAP). We improve upon an existing Lagrangian relaxation approach presented in previous work [126] to obtain strong upper and lower bounds for the problem. We exploit the closeness to QAP and generalize a dual descent method for updating the Lagrangian multipliers to the generalized problem. We have implemented the revised algorithm from scratch as the software tool NATALIE 2.0. In an extensive computational study on protein interaction networks for six different species, we compare NATALIE 2.0 to GRAAL and ISORANK, evaluating the number of conserved edges as well as functional coherence of the modules in terms of GO annotation. We find that NATALIE 2.0 outperforms the alternative methods with respect to quality and running time. Our software tool NATALIE 2.0 as well as all data sets used in this study are publicly available at http://planet-lisa.net.

2.2 Preliminaries

Given two simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, an *alignment* $a : V_1 \rightharpoonup V_2$ is a *partial injective function* from V_1 to V_2. As such we have that an alignment relates every node in V_1 to at most one node in V_2 and that conversely every node in V_2 has at most one counterpart in V_1. An alignment is assigned a real-valued *score* using an additive scoring function s defined as follows:

$$s(a) = \sum_{v \in V_1} c(v, a(v)) + \sum_{\substack{v,w \in V_1 \\ v<w}} w(v, a(v), w, a(w)) \tag{2.1}$$

where $c : V_1 \times V_2 \to \mathbb{R}$ is the score of aligning a pair of nodes in V_1 and V_2 respectively. On the other hand, $w : V_1 \times V_2 \times V_1 \times V_2 \to \mathbb{R}$ allows for scoring topological similarity. The problem of global pairwise network alignment (GNA) is to find the highest scoring alignment a^*, i.e. $a^* = \arg\max s(a)$. Figure 2.1 shows an example.

NP-hardness of GNA follows by a simple reduction from the decision problem CLIQUE, which asks whether there is a clique of cardinality at least k in a given simple graph $G = (V, E)$ [120]. The corresponding GNA instance concerns the alignment of the complete graph of k vertices $K_k = (V_k, E_k)$ with G using the scoring function

$s(a) = |\{(v,w) \in E_k \mid (a(v),a(w)) \in E\}|$. Since an alignment is injective, there is a clique of cardinality at least k if and only if the cost of the optimal alignment is $\binom{k}{2}$. The close relationship of GNA with the quadratic assignment problem is more easily observed when formulating GNA as a mathematical program. Throughout the remainder of the text we use dummy variables $i,j \in \{1,\dots,|V_1|\}$ and $k,l \in \{1,\dots,|V_2|\}$ to denotes nodes in V_1 and V_2, respectively. Let C be a $|V_1| \times |V_2|$ matrix such that $c_{ik} = c(i,k)$ and let W be a $(|V_1| \times |V_2|) \times (|V_1| \times |V_2|)$ matrix whose entries w_{ikjl} correspond to interaction scores $w(i,k,j,l)$. Now we can formulate GNA as

$$\max_x \quad \sum_{i,k} c_{ik}x_{ik} + \sum_{\substack{i,j \\ i<j}} \sum_{\substack{k,l \\ k\neq l}} w_{ikjl}x_{ik}x_{jl} \tag{IQP}$$

$$\text{s.t.} \quad \sum_l x_{jl} \leq 1 \qquad \forall j \tag{2.2}$$

$$\sum_j x_{jl} \leq 1 \qquad \forall l \tag{2.3}$$

$$x_{ik} \in \{0,1\} \qquad \forall i,k \tag{2.4}$$

where the decision variable x_{ik} indicates whether the i-th node in V_1 is aligned with the k-th node in V_2. The above formulation shares many similarities with Lawler's formulation [137] of the QAP. However, instead of finding an assignment we are interested in finding a matching, which is reflected in constraints (2.2) and (2.3) being inequalities rather than equalities. As can be seen in (2.1) we only consider the upper triangle of W rather than the entire matrix. An analogous way of looking at this, is to consider W to be symmetric. This is usually not the case for QAP instances. In addition, due to the fact that biological input graphs are typically sparse, we have that W is sparse as well. These differences allow us to come up with an effective method of solving the problem as we will see in the following.

2.3 Method

The relaxation presented here follows the same lines as the one given by Adams and Johnson for the QAP [1]. We start by linearizing (IQP) by introducing binary variables y_{ikjl} defined as $y_{ikjl} := x_{ik}x_{jl}$ and constraints $y_{ikjl} \leq x_{jl}$ and $y_{ikjl} \leq x_{ik}$ for all $i \leq j$ and $k \neq l$. If we assume that all entries in W are positive, we do not need to enforce that $y_{ikjl} \geq x_{ik} + x_{jl} - 1$. In Section 2.5 we will discuss this assumption. Rather than using the aforementioned constraints, we make use of a stronger set of constraints which we obtain by multiplying constraints (2.2) and (2.3) by x_{ik}:

$$\sum_{\substack{l \\ l\neq k}} y_{ikjl} = \sum_{\substack{l \\ l\neq k}} x_{ik}x_{jl} \leq \sum_l x_{ik}x_{jl} \leq x_{ik}, \quad \forall i,j,k,\ i<j \tag{2.5}$$

$$\sum_{\substack{j \\ j>i}} y_{ikjl} = \sum_{\substack{j \\ j>i}} x_{ik}x_{jl} \leq \sum_j x_{ik}x_{jl} \leq x_{ik}, \quad \forall i,k,l,\ k\neq l \tag{2.6}$$

We proceed by splitting the variable y_{ikjl} (where $i<j$ and $k \neq l$). In other words, we extend the objective function such that the counterpart of y_{ikjl} becomes y_{jlik}. This

is accomplished by rewriting the dummy constraint in (2.6) to $j \neq i$. In addition, we split the weights: $w_{ikjl} = w_{jlik} = (w'_{ikjl}/2)$ where w'_{ikjl} denotes the original weight. Furthermore, we require that the counterparts of the split decision variables assume the same value, which amounts to

$$\max_{x,y} \quad \sum_{i,k} c_{ik}x_{ik} + \sum_{\substack{i,j \\ i<j}} \sum_{\substack{k,l \\ k\neq l}} w_{ikjl}y_{ikjl} + \sum_{\substack{i,j \\ i>j}} \sum_{\substack{k,l \\ k\neq l}} w_{ikjl}y_{ikjl} \tag{ILP}$$

$$\text{s.t.} \quad \sum_{l} x_{jl} \leq 1 \qquad \forall j \tag{2.7}$$

$$\sum_{j} x_{jl} \leq 1 \qquad \forall l \tag{2.8}$$

$$\sum_{\substack{l \\ l\neq k}} y_{ikjl} \leq x_{ik} \qquad \forall i,j,k,\ i \neq j \tag{2.9}$$

$$\sum_{\substack{j \\ j\neq i}} y_{ikjl} \leq x_{ik} \qquad \forall i,k,l,\ k \neq l \tag{2.10}$$

$$y_{ikjl} = y_{jlik} \qquad \forall i,j,k,l,\ i < j, k \neq l \tag{2.11}$$

$$y_{ikjl} \in \{0,1\} \qquad \forall i,j,k,l,\ i \neq j, k \neq l \tag{2.12}$$

$$x_{ik} \in \{0,1\} \qquad \forall i,k \tag{2.13}$$

We can solve the continuous relaxation of (ILP) via its Lagrangian dual by dualizing the linking constraints (2.11) with multiplier λ:

$$\min_{\lambda} \quad Z_{\text{LD}}(\lambda)\ , \tag{LD}$$

where $Z_{\text{LD}}(\lambda)$ equals

$$\max_{x,y} \quad \sum_{i,k} c_{ik}x_{ik} + \sum_{\substack{i,j \\ i<j}} \sum_{\substack{k,l \\ k\neq l}} (w_{ikjl} + \lambda_{ikjl})y_{ikjl} + \sum_{\substack{i,j \\ i>j}} \sum_{\substack{k,l \\ k\neq l}} (w_{ikjl} - \lambda_{jlik})y_{ikjl}$$

$$\text{s.t.} \quad \text{(2.7), (2.8), (2.9), (2.10), (2.12) and (2.13)}$$

Now that the linking constraints have been dualized, one can observe that the remaining constraints decompose the variables into $|V_1||V_2|$ disjoint groups, where variables across groups are not linked by any constraint, and where each group contains a variable x_{ik} and variables y_{ikjl} for $j \neq i$ and $l \neq k$. Hence, we have

$$Z_{\text{LD}}(\lambda) = \max_{x} \quad \sum_{i,k} [c_{ik} + v_{ik}(\lambda)]x_{ik} \tag{LD$_\lambda$}$$

$$\text{s.t.} \quad \sum_{l} x_{jl} \leq 1 \qquad \forall j \tag{2.14}$$

$$\sum_{j} x_{jl} \leq 1 \qquad \forall l \tag{2.15}$$

$$x_{ik} \in \{0,1\} \qquad \forall i,k \tag{2.16}$$

which corresponds to a maximum weight bipartite matching problem on the so-called *alignment graph* $G_m = (V_1 \cup V_2, E_m)$. In the general case G_m is a complete bipartite

graph, i.e. $E_m = \{(i,k) \mid i \in V_1, v_2 \in V_2\}$. However, by exploiting biological knowledge one can make G_m more sparse by excluding biologically-unlikely edges (see Section 2.4). For the global problem, the weight of a matching edge (i,k) is set to $c_{ik} + v_{ik}(\lambda)$, where the latter term is computed as

$$v_{ik}(\lambda) = \max_y \sum_{\substack{j \\ j>i}} \sum_{\substack{l \\ l \neq k}} (w_{ikjl} + \lambda_{ikjl}) y_{ikjl} + \sum_{\substack{j \\ j<i}} \sum_{\substack{l \\ l \neq k}} (w_{ikjl} - \lambda_{jlik}) y_{ikjl} \qquad (\mathrm{LD}^{ik}_{\lambda})$$

$$\text{s.t.} \quad \sum_{\substack{l \\ l \neq k}} y_{ikjl} \leq 1 \qquad \forall j,\ j \neq i \qquad (2.17)$$

$$\sum_{\substack{j \\ j \neq i}} y_{ikjl} \leq 1 \qquad \forall l,\ l \neq k \qquad (2.18)$$

$$y_{ikjl} \in \{0,1\} \qquad \forall j,l. \qquad (2.19)$$

Again, this is a maximum weight bipartite matching problem on the same alignment graph but excluding edges incident to either i or k and using different edge weights: the weight of an edge (j,l) is $w_{ikjl} + \lambda_{ikjl}$ if $j > i$, or $w_{ikjl} - \lambda_{jlik}$ if $j < i$. So in order to compute $Z_{\mathrm{LD}}(\lambda)$, we need to solve a total number of $|V_1||V_2| + 1$ maximum weight bipartite matching problems, which, using the Hungarian algorithm [134, 152] can be done in $O(n^5)$ time, where $n = \max(|V_1|, |V_2|)$. In case the alignment graph is sparse, i.e. $O(|E_m|) = O(n)$, $Z_{\mathrm{LD}}(\lambda)$ can be computed in $O(n^4 \log n)$ time using the successive shortest path variant of the Hungarian algorithm [70]. It is important to note that for any λ, $Z_{\mathrm{LD}}(\lambda)$ is an upper bound on the score of an optimal alignment. This is because any alignment a is feasible to $Z_{\mathrm{LD}}(\lambda)$ and does not violate the original linking constraints and therefore has an objective value equal to $s(a)$. In particular, the optimal alignment a^* is also feasible to $Z_{\mathrm{LD}}(\lambda)$ and hence $a^* \leq Z_{\mathrm{LD}}(\lambda)$. Since the two sets of problems resulting from the decomposition both have the integrality property [69], the smallest upper bound we can achieve equals the linear programming (LP) bound of the continuous relaxation of (ILP) [91]. Given solution (x,y) to $Z_{\mathrm{LD}}(\lambda)$, we obtain a lower bound on $s(a^*)$, denoted $Z_{\mathrm{lb}}(\lambda)$, by considering the score of the alignment encoded in x.

2.3.1 Solving strategies

In this section we will discuss strategies for identifying Lagrangian multipliers λ that yield an as small as possible gap between the upper and lower bound resulting from the solution to $Z_{\mathrm{LD}}(\lambda)$.

Subgradient optimization. We start by discussing subgradient optimization, which is originally due to Held and Karp [98]. The idea is to generate a sequence $\lambda^0, \lambda^1, \ldots$ of Lagrangian multiplier vectors starting from $\lambda^0 = \mathbf{0}$ as follows:

$$\lambda^{t+1}_{ikjl} = \lambda^t_{ikjl} - \frac{\alpha \cdot (Z_{\mathrm{LD}}(\lambda) - Z_{\mathrm{lb}}(\lambda))}{\|g(\lambda^t)\|^2} g(\lambda^t_{ikjl}) \qquad \forall i,j,k,l,\ i<j, k \neq l \qquad (2.20)$$

where $g(\lambda^t_{ikjl})$ corresponds to the subgradient of multiplier λ^t_{ikjl}, i.e. $g(\lambda^t_{ikjl}) = y_{ikjl} - y_{jlik}$, and α is the step size parameter. Initially α is set to 1 and it is halved if neither

$Z_{\mathrm{LD}}(\lambda)$ nor $Z_{\mathrm{lb}}(\lambda)$ have improved for over N consecutive iterations. Conversely, α is doubled if M times in a row there was an improvement in either $Z_{\mathrm{LD}}(\lambda)$ or $Z_{\mathrm{lb}}(\lambda)$ [39]. In case all subgradients are zero, the optimal solution has been found and the scheme terminates. Note that this is not guaranteed to happen. Therefore we abort the scheme after exceeding a time limit or a pre-specified number of iterations. In addition, we terminate if α has dropped below machine precision. Algorithm 1 gives the pseudo code of this procedure.

Algorithm 1: SUBGRADIENTOPT(λ, M, N)

$\alpha \leftarrow 1$; $n \leftarrow N$; $m \leftarrow M$
$[\mathrm{LB}^*, \mathrm{UB}^*] \leftarrow [Z_{\mathrm{lb}}(\lambda), Z_{\mathrm{LD}}(\lambda)]$
while $g(\lambda) \neq 0$ **do**
    $\lambda \leftarrow \lambda - \frac{\alpha(Z_{\mathrm{LD}}(\lambda) - Z_{\mathrm{lb}}(\lambda))}{\|g(\lambda^t)\|^2} g(\lambda^t)$
    if $[\mathrm{LB}^*, \mathrm{UB}^*] \setminus [Z_{\mathrm{lb}}(\lambda), Z_{\mathrm{LD}}(\lambda)] = \emptyset$ **then**
        $n \leftarrow n - 1$
    else
        $\mathrm{LB}^* \leftarrow \max[\mathrm{LB}^*, Z_{\mathrm{lb}}(\lambda)]$
        $\mathrm{UB}^* \leftarrow \min[\mathrm{UB}^*, Z_{\mathrm{LD}}(\lambda)]$
        $m \leftarrow m - 1$
    if $n = 0$ **then**
        $\alpha \leftarrow \alpha/2$; $n \leftarrow N$
    if $m = 0$ **then**
        $\alpha \leftarrow 2\alpha$; $m \leftarrow M$
return $[\mathrm{LB}^*, \mathrm{UB}^*]$

Dual descent. In this section we derive a dual descent method which is an extension of the one presented in [1]. The dual descent method takes as a starting point the dual of $Z_{\mathrm{LD}}(\lambda)$:

$$Z_{\mathrm{LD}}(\lambda) = \min_{\alpha,\beta} \quad \sum_i \alpha_i + \sum_k \beta_k \tag{2.21}$$

$$\text{s.t.} \quad \alpha_i + \beta_k \geq c_{ik} + v_{ik}(\lambda) \qquad \forall i,k \tag{2.22}$$

$$\alpha_i \geq 0 \qquad \forall i \tag{2.23}$$

$$\beta_k \geq 0 \qquad \forall k \tag{2.24}$$

where the dual of $v_{ik}(\lambda)$ is

$$v_{ik}(\lambda) = \min_{\mu,\nu} \quad \sum_{\substack{j \\ j\neq i}} \mu_j^{ik} + \sum_{\substack{l \\ l\neq k}} \nu_l^{ik} \tag{2.25}$$

$$\text{s.t.} \quad \mu_j^{ik} + \nu_l^{ik} \geq w_{ikjl} + \lambda_{ikjl} \qquad \forall j,l,\ j > i, l \neq k \tag{2.26}$$

$$\mu_j^{ik} + \nu_l^{ik} \geq w_{ikjl} - \lambda_{jlik} \qquad \forall j,l,\ j < i, l \neq k \tag{2.27}$$

$$\mu_j^{ik} \geq 0 \qquad \forall j \tag{2.28}$$

$$\nu_l^{ik} \geq 0 \qquad \forall l. \tag{2.29}$$

Suppose that for a given λ^t we have computed dual variables (α,β) solving (2.21) with objective value $Z_{\text{LD}}(\lambda^t)$, as well as dual variables (μ^{ik},ν^{ik}) yielding values $v_{ik}(\lambda)$ to linear programs (2.25). The goal now is to find λ^{t+1} such that the resulting bound is better or just as good, i.e. $Z_{\text{LD}}(\lambda^{t+1}) \leq Z_{\text{LD}}(\lambda^t)$. We prevent the bound from increasing, by ensuring that the dual variables (α,β) remain feasible to (2.21). This we can achieve by considering the slacks: $\pi_{ik}(\lambda) = \alpha_i + \beta_k - c_{ik} - v_{ik}(\lambda)$. So for (α,β) to remain feasible, we can only allow every $v_{ik}(\lambda^t)$ to increase by as much as $\pi_{ik}(\lambda^t)$. We can achieve such an increase by considering linear programs (2.25) and their slacks defined as

$$\gamma_{ikjl}(\lambda) = \begin{cases} \mu_j^{ik} + \nu_l^{ik} - w_{ikjl} + \lambda_{ikjl}, & \text{if } j > i, \\ \mu_j^{ik} + \nu_l^{ik} - w_{ikjl} - \lambda_{jlik}, & \text{if } j < i, \end{cases} \qquad \forall j,l,\ j \neq i, l \neq k, \tag{2.30}$$

and update the multipliers in the following way.

Lemma 2.1 *The adjustment scheme below yields solutions to linear programs* (2.25) *with objective values* $v_{ik}(\lambda^{t+1})$ *at most* $\pi_{ik}(\lambda^t) + v_{ik}(\lambda^t)$ *for all* i,k.

$$\begin{aligned} \lambda_{ikjl}^{t+1} = \lambda_{ikjl}^t &+ \varphi_{ikjl}\left[\gamma_{ikjl}(\lambda^t) + \tau_{ik}\left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{ik}(\lambda^t)\right] \\ &- \varphi_{jlik}\left[\gamma_{jlik}(\lambda^t) + \tau_{jl}\left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{jl}(\lambda^t)\right] \end{aligned} \tag{2.31}$$

for all $j,l,\ i<j, k\neq l$, *where* $n_1 = |V_1|$, $n_2 = |V_2|$, *and* $0 \leq \varphi_{ikjl}, \tau_{jl} \leq 1$ *are parameters.*

Proof We prove the lemma by showing that for any i,k there exists a feasible solution (μ'^{ik},ν'^{ik}) to (2.25) whose objective value $v_{ik}(\lambda^{t+1})$ is at most $\pi_{ik}(\lambda^t) + v_{ik}(\lambda^t)$. Let (μ^{ik},ν^{ik}) be the solution to (2.25) given multipliers λ^t. We claim that setting

$$\mu_j'^{ik} = \mu_j^{ik} + \frac{\pi_{ik}(\lambda^t)}{2(n_1-1)} \qquad \forall j,\ j \neq i$$

$$\nu_l'^{ik} = \nu_j^{ik} + \frac{\pi_{ik}(\lambda^t)}{2(n_2-1)} \qquad \forall l,\ l \neq k,$$

results in a feasible solution to (2.25) given multipliers λ^{t+1}. We start by showing that constraints (2.26) and (2.27) are satisfied. From (2.31) the following bounds on

λ^{t+1} follow.

$$\lambda^t_{ikjl} - \gamma_{jlik}(\lambda^t) - \left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{jl}(\lambda^t) \leq \lambda^{t+1}_{ikjl} \quad \forall j,l,\, j<i, l\neq k$$
$$\lambda^{t+1}_{ikjl} \leq \lambda^t_{ikjl} + \gamma_{ikjl}(\lambda^t) + \left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{ik}(\lambda^t) \quad \forall j,l,\, j<i, l\neq k.$$

Therefore we have that the following inequalities imply constraints (2.26) and (2.27) for all $j,l,\, j>i,\, l\neq k$:

$$\mu'^{ik}_j + \nu'^{ik}_l \geq w_{ikjl} + \lambda^t_{ikjl} + \gamma_{ikjl}(\lambda^t) + \left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{ik}(\lambda^t)$$

and for all $j,l,\, j<i,\, l\neq k$

$$\mu'^{ik}_j + \nu'^{ik}_l \geq w_{ikjl} - \lambda^t_{jlik} + \gamma_{ikjl}(\lambda^t) + \left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{ik}(\lambda^t).$$

Constraints (2.26) and (2.27) are indeed implied, as, for all $j,l,\, j>i,\, l\neq k$,

$$\begin{aligned}\mu'^{ik}_j + \nu'^{ik}_l &= \mu^{ik}_j + \nu^{ik}_l + \left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{ik}(\lambda^t)\\ &\geq w_{ikjl} + \lambda^t_{ikjl} + \gamma_{ikjl}(\lambda^t) + \left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{ik}(\lambda^t)\end{aligned}$$

and for all $j,l,\, j<i,\, l\neq k$

$$\begin{aligned}\mu'^{ik}_j + \nu'^{ik}_l &= \mu^{ik}_j + \nu^{ik}_l + \left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{ik}(\lambda^t)\\ &\geq w_{ikjl} - \lambda^t_{jlik} + \gamma_{ikjl}(\lambda^t) + \left(\frac{1}{2(n_1-1)} + \frac{1}{2(n_2-1)}\right)\pi_{ik}(\lambda^t).\end{aligned}$$

Since $\mu^{ik}_j, \nu^{ik}_l \geq 0$ ($\forall j,l,\, j\neq i, l\neq k$) and by definition $\pi_{ik}(\lambda^t)\geq 0$, constraints (2.28) and (2.29) are satisfied as well. The objective value of (μ'^{ik}, ν'^{ik}) is given by

$$\sum_{\substack{j\\ j\neq i}} \mu'^{ik}_j + \sum_{\substack{l\\ l\neq k}} \nu'^{ik}_l = \sum_{\substack{j\\ j\neq i}} \mu^{ik}_j + \sum_{\substack{l\\ l\neq k}} \nu^{ik}_l + \pi_{ik}(\lambda^t) = v_{ik}(\lambda^t) + \pi_{ik}(\lambda^t).$$

Since (2.25) are minimization problems and there exist, for all i,k, feasible solutions with objective values $v_{ik}(\lambda^t)+\pi_{ik}(\lambda^t)$, we can conclude that the objective values of the solutions are bounded by this quantity. The lemma now follows. □

We use $\varphi = 0.5$, $\tau = 1$, and perform the dual descent method L successive times (see Algorithm 2).

Overall method. Our overall method combines both the subgradient optimization and dual descent method. We do this performing the subgradient method until termination and then switching over to the dual descent method. This procedure is repeated K times (see Algorithm 3).

Algorithm 2: DUALDESCENT(λ, L)

$\varphi \leftarrow 0.5$; $[\mathrm{LB}^*, \mathrm{UB}^*] \leftarrow [Z_{\mathrm{lb}}(\lambda), Z_{\mathrm{LD}}(\lambda)]$
for $n \leftarrow 1$ **to** L **do**
  foreach $i,k,j,l, i<j, k \neq l$ **do**
    $\lambda_{ikjl} \leftarrow \lambda_{ikjl} + \varphi(\gamma_{ikjl} + \frac{\pi_{ik}(\lambda)}{2(n_1-1)} + \frac{\pi_{ik}(\lambda)}{2(n_2-1)}) - \varphi(\gamma_{jlik} + \frac{\pi_{jl}(\lambda)}{2(n_1-1)} + \frac{\pi_{jl}(\lambda)}{2(n_2-1)})$
  $\mathrm{LB}^* \leftarrow \max[\mathrm{LB}^*, Z_{\mathrm{lb}}(\lambda)]$
  $\mathrm{UB}^* \leftarrow Z_{\mathrm{LD}}(\lambda)$
return $[\mathrm{LB}^*, \mathrm{UB}^*]$

Algorithm 3: NATALIE(K, L, M, N)

$\lambda \leftarrow \mathbf{0}$; $[\mathrm{LB}^*, \mathrm{UB}^*] \leftarrow [0, \infty]$
for $k \leftarrow 1$ **to** K **do**
  $[\mathrm{LB}^*, \mathrm{UB}^*] \leftarrow$ SUBGRADIENTOPT(λ, M, N) $\cap [\mathrm{LB}^*, \mathrm{UB}^*]$
  $[\mathrm{LB}^*, \mathrm{UB}^*] \leftarrow$ DUALDESCENT(λ, L) $\cap [\mathrm{LB}^*, \mathrm{UB}^*]$
return $[\mathrm{LB}^*, \mathrm{UB}^*]$

We implemented NATALIE in C++ using the LEMON graph library (http://lemon.cs.elte.hu/). The successive shortest path algorithm for maximum weight bipartite matching was implemented and contributed to LEMON. Special care was taken to deal with the inherent numerical instability of floating point numbers. Our implementation supports both the GraphML and GML graph formats. Rather than using one big alignment graph, we store and use a different alignment graph for every local problem (LD_λ^{ik}). This proved to be a huge improvement in running times, especially when the global alignment graph is sparse. NATALIE is publicly available at http://planet-lisa.net.

2.4 Experimental evaluation

From the STRING database v8.3 [193], we obtained PPI networks for the following six species: *C. elegans* (cel), *S. cerevisiae* (sce), *D. melanogaster* (dme), *R. norvegicus* (rno), *M. musculus* (mmu) and *H. sapiens* (hsa). We only considered interactions that were experimentally verified. Table 2.1 shows the sizes of the networks. We performed, using the BLOSUM62 matrix, an all-against-all global sequence alignment on the protein sequences of all $\binom{6}{2} = 15$ pairs of networks. We used affine gap penalties with a gap-open penalty of 2 and a gap-extension penalty of 10. The first experiment in Section 2.4.1 compares the raw performance of IsoRank, GRAAL and NATALIE in terms of objective value. In Section 2.4.2 we evaluate the biological relevance of the alignments produced by the three methods. All experiments were conducted on a compute cluster with 2.26 GHz processors with 24 GB of RAM.

species	nodes	annotated	interactions
cel (c)	5,948	4,694	23,496
sce (s)	6,018	5,703	131,701
dme (d)	7,433	6,006	26,829
rno (r)	8,002	6,786	32,527
mmu (m)	9,109	8,060	38,414
hsa (h)	11,512	9,328	67,858

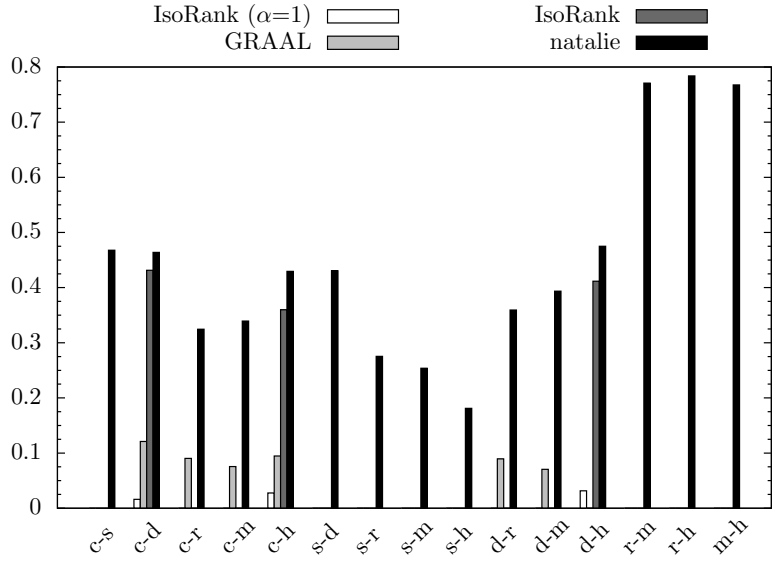
Table 2.1: Characteristics of input networks considered in this study. The columns contain species identifier, number of nodes in the network, number of annotated nodes thereof, and number of interactions

2.4.1 Edge-correctness

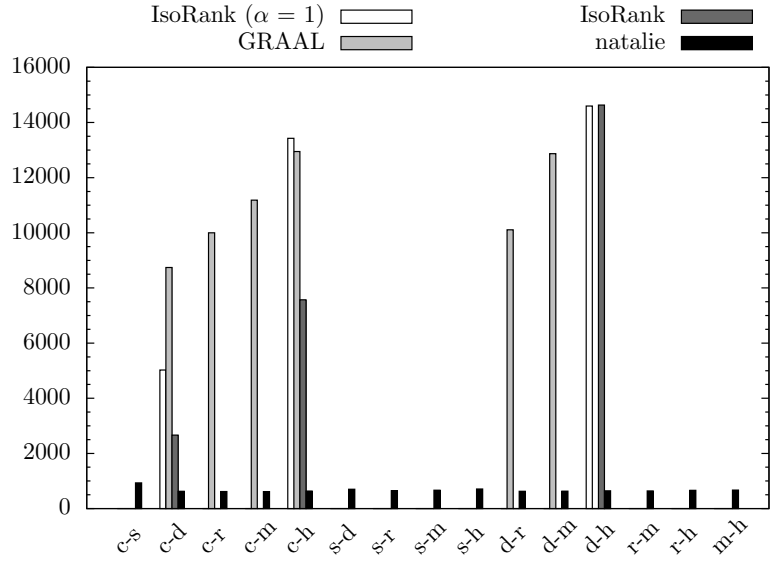
The objective function used for scoring alignments in GRAAL counts the number of mapped edges. Such an objective function is easily expressible in our framework using $s(a) = |\{(v,w) \in E_1 \mid (a(v),a(w)) \in E_2\}|$ and can also be modeled using the ISORANK scoring function. In order to compare performance of the methods across instances, we normalize the scores by dividing by $\min(|E_1|,|E_2|)$. This measure is called the edge-correctness by Kuchaiev et al. [133].

As mentioned in Section 2.3, our method benefits greatly from using a sparse alignment graph. To that end, we use the e-values obtained from the all-against-all sequence alignment to prohibit biologically unlikely matchings by only considering protein-pairs whose E-value is at most 100. Note that this only applies to NATALIE as both GRAAL and ISORANK consider the complete alignment graph. On each of the 15 instances, we ran GRAAL with 3 different random seeds and sampled the input parameter which balances the contribution of the graphlets with the node degrees uniformly within the allowed range of $[0,1]$. As for ISORANK, when setting the parameter α | which controls to what extent topological similarity plays a role | to the desired value of 1, very poor results were obtained. Therefore we also sampled this parameter within its allowed range and re-evaluated the resulting alignments in terms of edge-correctness. NATALIE was run with a time limit of 10 minutes and $K = 3, L = 100$, $M = 10$, $N = 20$. For both GRAAL and ISORANK only the highest-scoring results were considered.

Figure 2.2 shows the results. ISORANK was only able to compute alignments for three out of the 15 instances. On the other instances ISORANK crashed, which may be due to the large size of the input networks. For GRAAL no alignments concerning *sce* could be computed, which is due to the large number of edges in the network on which the graphlet enumeration procedure choked: in 12 hours only for 3% of the nodes the graphlet degree vector was computed. As for the last three instances, GRAAL crashed due to exceeding the memory limit inherent to 32-bit processes. Unfortunately no 64-bit executable was available. On the instances for which GRAAL could compute alignments, the performance | both in solution quality and running time | is very poor when compared to ISORANK and NATALIE. NATALIE outperforms ISORANK in both running time and solution quality.



(a) Edge correctness



(b) Running times in seconds

Figure 2.2: Performance of the three different methods for the all-against-all species comparisons (15 alignment instances). Missing bars correspond to exceeded time/memory limits or software crashes.

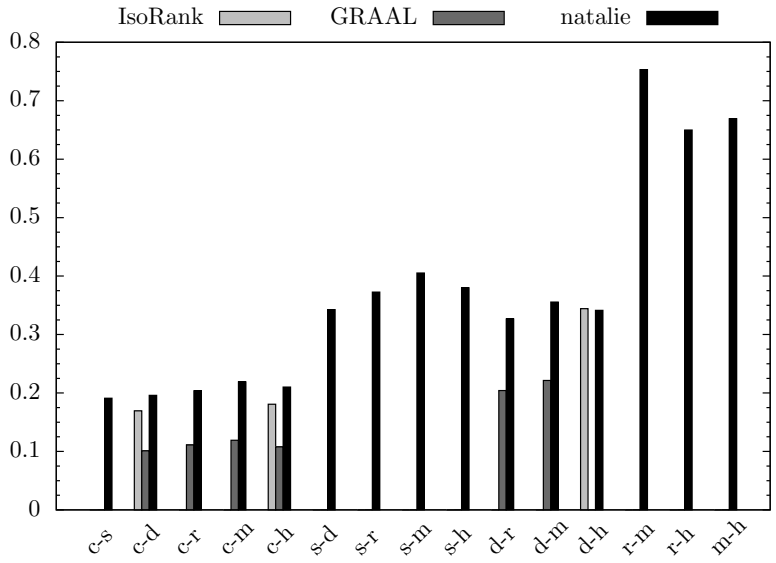


Figure 2.3: Biological relevance of the alignments measured via GO similarity

2.4.2 GO similarity

In order to measure the biological relevance of the obtained network alignments, we make use of the Gene Ontology (GO) [16]. For every node in each of the six networks we obtained a set of GO annotations (see Table 2.1 for the exact numbers). Each annotation set was extended to a multiset by including all ancestral GO terms for every annotation in the original set. Subsequently we employed a similarity measure that compares a pair of aligned nodes based on their GO annotations and also takes into account the relative frequency of each annotation [112]. Since the similarity measure assigns a score between 0 and 1 to every aligned node pair, the highest similarity score one can get for any alignment is the minimum number of annotated nodes in either of the networks. Therefore we can normalize the similarity scores by this quantity. Unlike the previous experiment, this time we considered the bitscores of the pairwise global sequence alignments. Similarly to IsoRank parameter α, we introduced a parameter $\beta \in [0,1]$ such that the sequence part of the score has weight $(1-\beta)$ and the topology part has weight β. For both IsoRank and natalie we sampled the weight parameters uniformly in the range $[0,1]$ and showed the best result in Figure 2.3. There we can see that both natalie and IsoRank identify functionally coherent alignments.

2.5 Conclusions

Inspired by results for the closely related quadratic assignment problem (QAP), we have presented new algorithmic ideas in order to make a Lagrangian relaxation approach for global network alignment practically useful and competitive. In particular,

we have generalized a dual descent method for the QAP. We have found that combining this scheme with the traditional subgradient optimization method leads to fastest progress of upper and lower bounds.

Our implementation of the new method, NATALIE 2.0, works very well and fast when aligning biological networks, which we have shown in an extensive study on the alignment of cross-species PPI networks. We have compared NATALIE 2.0 to those state-of-the-art methods whose scoring schemes can be expressed as special cases of the scoring scheme we propose. Currently, these methods are IsoRank and Graal. Our experiments show that the Lagrangian relaxation approach is a very powerful method and that it currently outperforms the competitors in terms of quality of the results and running time.

Currently, all methods, including ours, approach the global network alignment problem heuristically, that is, the computed alignments are not guaranteed to be optimal solutions of the problem. While the other approaches are intrinsically heuristic—both IsoRank and Graal, for instance, approximate the neighborhood of a node and then match it with a similar node—the inexactness in our methods has two causes that we plan to address in future work: On the one hand, there may still be a gap between upper and lower bound of the Lagrangian relaxation approach after the last iteration. We can use these bounds, however, in a branch-and-bound approach that will compute provably optimal solutions. On the other hand, we currently do not consider the complete bipartite alignment graph and may therefore miss the optimal alignment. Here, we will investigate preprocessing strategies, in the spirit of [216], to safely sparsify the input bipartite graph without violating optimality conditions.

The independence of the local problems (LD_λ^{ik}) allows for easy parallelization, which, when exploited would lead to an even faster method. Another improvement in running times might be achieved when considering more involved heuristics for computing the lower bound, such as local search. More functionally-coherent alignments can be obtained when considering a scoring function where node-to-node correspondences are not only scored via sequence similarity but also for instance via GO similarity. In certain cases, even negative weights for topological interactions might be desired in which case one needs to reconsider the assumption of entries of matrix W being positive.

Acknowledgments. We thank SARA Computing and Networking Services (www.sara.nl) for their support in using the Lisa Compute Cluster. In addition, we are very grateful to Bernd Brandt for helping out with various bioinformatics issues and also to Samira Jaeger for providing code and advice on the GO similarity experiments.

Chapter 3

A web server for PPI network querying

Published as:

M. El-Kebir[†], B. W. Brandt[†], J. Heringa, and G. W. Klau. NatalieQ: A web server for protein-protein interaction network querying. *BMC Systems Biology*, 8(1):40, 2014.

[†]joint first authorship

Abstract

Background: Molecular interactions need to be taken into account to adequately model the complex behavior of biological systems. These interactions are captured by various types of biological networks, such as metabolic, gene-regulatory, signal transduction and protein-protein interaction networks. We recently developed NATALIE, which computes high-quality network alignments via advanced methods from combinatorial optimization.
Results: Here, we present NATALIEQ, a web server for topology-based alignment of a specified query protein-protein interaction network to a selected target network using the NATALIE algorithm. By incorporating similarity at both the sequence and the network level, we compute alignments that allow for the transfer of functional annotation as well as for the prediction of missing interactions. We illustrate the capabilities of NATALIEQ with a biological case study involving the Wnt signaling pathway.
Conclusions: We show that topology-based network alignment can produce results complementary to those obtained by using sequence similarity alone. We also demonstrate that NATALIEQ is able to predict putative interactions. The server is available at: http://www.ibi.vu.nl/programs/natalieq/.

Keywords: Network alignment, protein-protein interaction, sequence similarity, topology, Wnt signaling pathway.

3.1 Background

To adequately model complex behavior of biological systems one needs to take molecular interactions into account. These interactions are captured by various types of biological networks such as metabolic, gene-regulatory, signal transduction and protein-protein interaction (PPI) networks. Recent advances in technological developments and computational methods have resulted in large amounts of network data. For instance, STRING [77], a database of experimentally verified and computationally predicted protein interactions, grew from 261,033 proteins in 89 organisms in 2003 to 5,214,234 proteins in 1,133 organisms in January 2014. However, the development of solid methods for analyzing network data is lagging behind, particularly in the field of comparative network analysis. Here, one wants to detect commonalities between biological networks from different strains or species, or derived from different conditions. In contrast to traditional comparison at sequence level, topology-based comparison methods explicitly take interactions into account and are thus more suitable to compare networks. Subnetworks with shared interactions across species allow for improved transfer of functional annotations from one species to the other by using more information than sequence alone [17].

We have developed NATALIEQ, a web server for accurate topology-based protein-protein interaction network queries. It provides an interface to the general network alignment method NATALIE [72, 126], which is fast and supports various scoring schemes taking both node-to-node correspondences and network topologies into account. Briefly, NATALIE views the network alignment problem as a generalization of the well-studied quadratic assignment problem and solves it using techniques from integer linear programming.

Currently, only few web servers for comparative network analysis exist. The PathBLAST web server [124] reports exact and approximate hits in a target PPI network for a user-defined simple query, expressed as a linear path of up to five proteins. The NetworkBLAST web server [117] finds locally-conserved protein complexes between species-specific PPI networks. NetAligner [156], a recent web server, allows the comparison of user-defined networks or whole interactomes within a set of fixed species using a heuristic network alignment with no guarantees on the optimality of the identified solutions.

Our contribution is twofold. First, NATALIEQ employs a new scoring function to produce high-quality pairwise alignments between a user-specified query network of arbitrary topology and interactomes of several model species and human. The score of an alignment is primarily based on the number of conserved interactions, while sequence similarity is used as a secondary, subordinate optimization goal. In addition, the alignments computed by the underlying NATALIE algorithm come with a quality guarantee that often proves their optimality. Second, through an interactive visualization of the alignment, the user can quickly get an overview of conserved and non-conserved interactions and can use the protein descriptions of the nodes to assess the alignment. We illustrate a usage scenario of the web server on the Wnt signaling pathway and demonstrate that NATALIEQ is able to predict putative interactions that are not detected by other methods.

3.2 Implementation

3.2.1 Network alignment algorithm

NATALIE, the alignment method of NATALIEQ, is applicable to any type of network and supports any additive score function taking both node-to-node correspondences and topology into account. Here, we take as input a pair of PPI networks whose nodes and edges correspond to proteins and their interactions. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two PPI networks whose edges have a confidence value above a user-defined threshold $c_{\min}$. We denote by $E(v_1, v_2)$ the E-value of proteins $v_1 \in V_1$ and $v_2 \in V_2$ obtained by an all-against-all sequence alignment. Typically, G_1 is a smaller query network such as a specific pathway of interest, and G_2 is a large species-specific PPI network.

A *network alignment* is a partial injective function $a : V_1 \to V_2$ with the additional requirement that if $v_1 \in V_1$ is aligned then $a(v_1) \in \{v_2 \in V_2 \mid E(v_1, v_2) \leq E_{\max}\}$. That is, every node $v_1 \in V_1$ is related to at most one node $v_2 \in V_2$ with E-value $E(v_1, v_2)$ below a pre-specified cut-off $E_{\max}$ and vice versa. We score the topology component of an alignment a as follows

$$t(a) = \frac{1}{\min\{|E_1|, |E_2|\}} \sum_{uv \in E_1} w(u, a(u), v, a(v))$$

with

$$w(u, a(u), v, a(v)) = \begin{cases} 1 & \text{if } (a(u), a(v)) \in E_2, \\ 0 & \text{otherwise.} \end{cases}$$

This score is also known as *edge correctness* and denotes the fraction of edges from the smaller query network that have been aligned. The problem of global pairwise network alignment is to find the highest-scoring alignment. Should there be several alignments with the same maximum edge correctness, we would prefer the alignment with the highest total bit score as obtained by an all-against-all sequence alignment—a bit score is an alignment quality score that, given a sequence database, takes all possible pairwise alignments into account. We achieve this in the following way. Let $b(v_1, v_2) \in [0, 1]$ be the normalized bit score of aligning protein $v_1 \in V_1$ with protein $v_2 \in V_2$. The total score of an alignment a is then

$$s(a) = t(a) + \frac{1}{1 + \min\{|E_1|, |E_2|\} \cdot \min\{|V_1|, |V_2|\}} \sum_{u \in V_1} b(u, a(u)).$$

That is, the score component is ensured to be strictly smaller than the score contribution of one conserved edge. Therefore ties among alignments with the same edge correctness are broken in favor of those with the highest overall bit score.

We use NATALIE to compute alignments with maximum total score. A specific feature of NATALIE is that any identified solution comes with an upper bound on the optimal score value. In the NATALIEQ setting with small query networks, the upper bound equals the score of the alignment found, thereby proving its optimality. The identified alignment is not necessarily optimal if there is a gap between the score and the upper bound. In that case the relative size of the gap provides a bound on the

error due to suboptimality. In a recent study [72] on aligning PPI networks of six different species, NATALIE was compared to state-of-the-art network alignment methods, evaluating the number of conserved edges as well as functional coherence of the modules in terms of Gene Ontology annotation. The study established NATALIE as a top network alignment method with respect to both alignment quality and running time.

3.2.2 Databases

We currently provide eight model species from STRING [77] and IntAct [125] as target databases. We added textual descriptions to the protein IDs. For the STRING networks, these descriptions are available as a separate publicly available download. We retrieved the protein descriptions for the IntAct networks by cross-referencing the IntAct UniProt identifiers with the Swiss-Prot and TrEMBL databases [196]. To allow NATALIEQ to take protein sequence information into account, we stored the amino acid sequences of the proteins in separate FASTA files per network. We retrieved these sequences from the STRING and IntAct databases. The target databases will be updated upon new releases of STRING and IntAct.

3.2.3 Processing

NATALIEQ computes a network alignment in a two-step fashion implemented in a Perl wrapper script. First, the wrapper invokes BLAST [9, 10] to create pairwise protein alignments between the sequences corresponding to the nodes of the query and target network. Next, the wrapper invokes NATALIE [72, 126] for different E-value cut-offs $E_{\max} \in \{0, 10^{-100}, 10^{-50}, 10^{-10}, 1, 10, 100\}$. Each cut-off $E_{\max}$ imposes restrictions on the allowed pairings, that is, only pairs $(u, a(u))$ with $u \in V_1$ whose E-value is at most $E_{\max}$ are allowed. During these computations, which take a few minutes for a typical network query, the user is updated about the progress and may bookmark the unique web page for this run or leave an e-mail address to be notified upon completion.

3.3 Results and discussion

3.3.1 Web server

The input of NATALIEQ consists of a query network that can be in several formats: a simple edge list format, Cytoscape's SIF format, IntAct's MITAB format or STRING's text-based format. The input file format is automatically detected. Optionally, the user can provide a FASTA file containing the protein sequences corresponding to the network nodes. In case no FASTA file is supplied and the node labels correspond to UniProt, RefSeq or GI identifiers, the corresponding sequences are retrieved automatically from the NCBI Protein database [214]. The user can select one of two well-known protein interaction databases (IntAct or STRING) and one of currently eight model species as target network. Options are the score function and the confidence threshold $c_{\min}$. We support two score functions: *topology*, which is the scoring function as defined previously, as the default option, and *sequence only*, which results

Overview for Dme from String (topology)

Run	E_{max}	Edge correctness	Sequence contribution
1	0	1/17 = 0.0588235	0.0091093
2	1e-100	8/17 = 0.470588	0.00794499
3	1e-50	10/17 = 0.588235	0.00845966
4	1e-10	16/17 = 0.941176	0.00890019
5	1	17/17 = 1	0.011376
6	10	17/17 = 1	0.011376
7	100	17/17 = 1	0.011376

Figure 3.1: NATALIEQ computation overview of the alignments of the Wnt query network against the target PPI network (STRING) of *D. melanogaster* using the *topology* score function.

in the best network alignment in terms of sequence similarity, disregarding topological information.

The output page first gives an overview of the results for the different E-value cut-offs (Figure 3.1). The user can select a result for detailed inspection. Interesting results to inspect are, for example, the one with best sequence similarity among the top-scoring topological similarities or the one with best topological score at lowest E-value cut-off. The detailed view starts with summary statistics about the input networks and the computational process (Figure 3.2). It then displays an interactive network alignment visualization using the Javascript D3 library (http://mbostock.github.com/d3/), which is a data-driven framework for information visualization. The visualization (Figure 3.3) shows the aligned part of the two networks, overlaying nodes and links using red color for the query and grey for the target network. Thus, a matched query-target node or link pair will be colored in both red and grey. This interactive network visualization shows the user which parts of the query and target networks are matched. Hovering over nodes and links displays tooltips with protein names and descriptions and link confidence, respectively, and allows for a quick overview of the alignment. If the user clicks on a node, information about that node is shown in a separate table, which in addition to the protein names and descriptions includes the bit score and E-value of the BLAST pairwise alignment and a hyperlink to the original database for more information about the target protein. The interface allows for a more detailed analysis by toggling the visibility of node labels, background target nodes and edges, unmatched query nodes and edges, and unmatched target edges.

In addition, the detailed view shows tables containing aligned query-target nodes, edges conserved in both query and target network, edges in the query network that remain unaligned, and unaligned edges in the target network whose incident nodes are aligned (Figure 3.4). The interactive visualization can be exported to a static SVG file and the user can download the alignment and the interaction tables for further off-line analysis. We support Cytoscape [186] by providing Cytoscape-compatible files

Input

Target network name	data/string/dme-7227.string
Target network size	13144 nodes and 1996782 edges
Query network size	11 nodes and 17 edges
Number of matching edges	475 edges
E-value cut-off	1
Confidence threshold	10%

Statistics

Elapsed time	0.208884s
Edge correctness	17/17 = 1
Sequence contribution	0.011376
Optimality gap	0%
Number of aligned pairs	11
Conserved interactions	17
Non-conserved interactions in query	0
Non-conserved interactions in target	22

Figure 3.2: NATALIEQ summary statistics for run number 5 ($E_{\max} = 1$). Alignment of the Wnt query network against the target PPI network (STRING) of *D. melanogaster* using the *topology* score function.

containing the entire alignment and query network as well as matched parts of the target network.

3.3.2 Case study: Wnt signaling pathway

To illustrate the capabilities of NATALIEQ, we consider a biological case study involving the Wnt signaling pathway whose abnormal signaling has been associated with cancer. This pathway is initiated by binding of secreted Wnt signaling proteins to the cell surface receptors Frizzled and LRP. This causes the activation of the signaling protein Dishevelled, which in turn inhibits the assembly of the degradation complex GSK-3β/axin/APC/β-catenin. As a result, the degradation of β-catenin is prevented causing it to accumulate in the nucleus. There, β-catenin forms a complex with LEF-1/TCF thereby displacing Groucho. The newly formed complex induces the transcription of various Wnt target genes, including c-myc which is a proto-oncogene encoding for a protein involved in cell growth and proliferation [3].

We manually constructed a PPI network of the pathway by using a subset of the proteins involved, namely WNT1, A2MR (LRP1), FZD1 (Frizzled-1), DVL1 (Dishevelled), AXIN1, GSK3B, CTNNB1 (β-catenin), APC, TCF7, TLE1 (Groucho), and MYC. For each of these proteins, we obtained their respective sequences from the STRING database. The edges we used correspond to the interactions described above. The

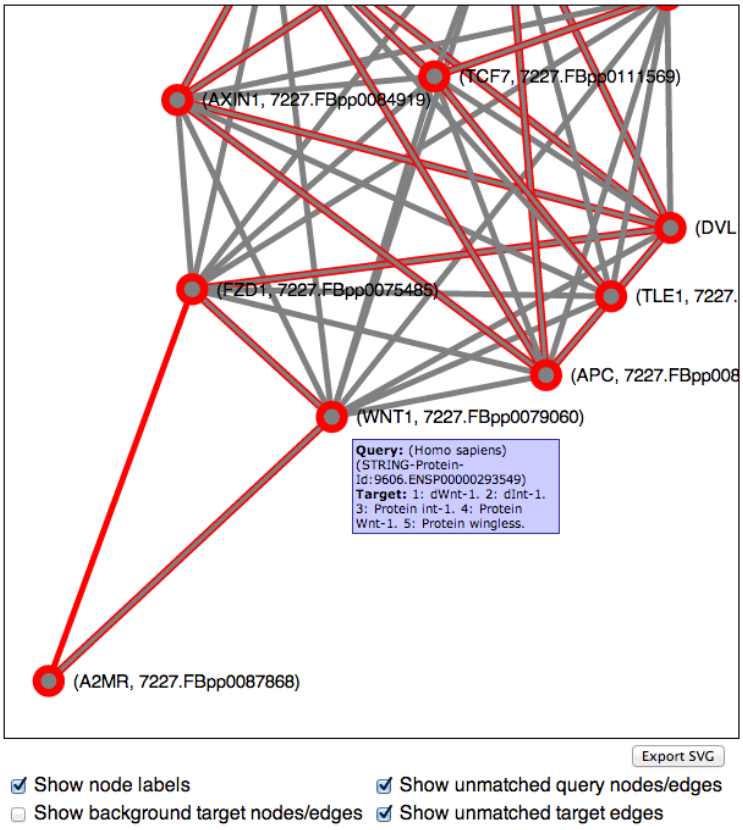


Figure 3.3: NATALIEQ interactive visualization component for aligning the Wnt query network (red) against the target PPI network (STRING, grey, matched part shown) of *D. melanogaster* using the *sequence only* score function at E-value cut-off 1. The purely red edges, for example, (FZD1, A2MR), hint at interactions that have been missed by the alignment. See also Figure 3.4, bottom table. The tool-tip appears when hovering over the nodes.

query network consists of 11 nodes and 17 edges and is available as the example network file on the main page of NATALIEQ.

As a first sanity check, we queried against the human PPI network from STRING with link confidence threshold $c_{\min} = 0.1$. For all E-value cut-offs, NATALIEQ found the optimal alignment where indeed all interactions are present and all query proteins are aligned with their identical counterparts in the human network as we could verify from the descriptions and interaction tables in the output.

For our next experiment, we used the PPI network of *D. melanogaster* as target. See also Figures 3.1–3.4 for an illustration. To study whether topological information improves comparative analysis, we compare the results of NATALIEQ using both the *topology* and *sequence only* score functions. We see that in the resulting *sequence only* alignments for E-value cut-offs larger than 10^{-10} one interaction of the query network is not mapped. This is the interaction between A2MR and FZD1. The counterpart of FZD1 in the sequence only alignment is FBpp0075485 with a bit score of 519 (E-value: $5 \cdot 10^{-177}$). The web server also provides the BLAST output, which shows that FZD1 is indeed sequence-wise most similar to FBpp0075485. NATALIEQ with the *topology* score function at E-value cut-offs larger than 10^{-10} is able to match all (17) query interactions and pairs FZD1 and FBpp0077788 with a bit score of only 150 (E-value: $6 \cdot 10^{-38}$). Although the bit score is less than the one obtained in the sequence-only alignment, the interaction A2MR–FZD1 is now present in the target network and has a normalized confidence of 0.172. So using NATALIEQ, we find that FZD1 may functionally be more related to FBpp0077788 than its sequence-wise most similar counterpart FBpp0075485. This hypothesis is corroborated by UniProtKB/SwissProt annotation indicating that the protein FBpp0077788 contains a Frizzled domain. Running the same example using the NetAligner web server [156] results in only 5 conserved interactions using default settings.

This example illustrates how NATALIEQ can facilitate the transfer of functional annotation across species. For instance, we could transfer functional annotation concerning the Wnt pathway between the human and fly networks by using the alignments we obtained.

3.4 Conclusions

We developed NATALIEQ, a web server for global pairwise network alignment of a prespecified query PPI network to a selected target network. The underlying alignment method computes alignments with a worst-case bound on their quality. For the biological query networks we considered, the optimality gap was closed and provably optimal alignments with respect to the used score function were thus found. The user can quickly get an overview of the alignment through the interactive visualization, where conserved and non-conserved interactions are easily visible.

Currently, we support eight different target species from both STRING and IntAct. NATALIEQ is extendible, and we will add more target networks in the future. In addition, we plan to exploit the general applicability of the underlying NATALIE method by facilitating the identification of network motifs through more sophisticated query networks where nodes are labeled by GO terms and edges are labeled by different interaction types, such as inhibition and activation.

Alignment

Query node	Target node	E-value	Bit score	Query description	Target description
WNT1	7227.FBpp0079060	5e-92	286	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000293549)	1: dWnt-1. 2: dInt-1. 3: Protein int-1. 4: Protein Wnt-1. 5: Protein wingless.
A2MR	7227.FBpp0087868	0	2724	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000243077)	1: CG33087.
FZD1	7227.FBpp0075485	5e-177	519	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000287934)	1: dFz1. 2: Frizzled. 3: Frizzled-1.
DVL1	7227.FBpp0073310	2e-143	435	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000368169)	1: Dishevelled protein. 2: Dishevelled, isoform B. 3: Segment polarity protein dishevelled.
AXIN1	7227.FBpp0084919	7e-18	87.8	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000262320)	1: Axin. 2: d-Axin. 3: AT13274p. 4: RH74443p. 5: LD38584p. 6: Axin, isoform B. 7: Axin, isoform D. 8: Axin, isoform C. 9: Axis inhibition protein.
GSK3B	7227.FBpp0070454	0	619	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000324806)	1: SGG. 2: FI05468p. 3: MIP03616p. 4: Shaggy, isoform J. 5: Shaggy, isoform I. 6: Shaggy, isoform M. 7: Shaggy, isoform C. 8: Shaggy, isoform E. 9: Shaggy, isoform L. 10: Shaggy, isoform K. 11: Shaggy, isoform F. 12: Shaggy, isoform H. 13: Protein kinase shaggy. 14: Protein zeste-white 3.
APC	7227.FBpp0084720	7e-146	509	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000257430)	1: APC-like. 2: Adenomatous polyposis coli.
CTNNB1	7227.FBpp0089031	0	1059	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000344456)	1: Armadillo protein. 2: Armadillo segment polarity protein.
TCF7	7227.FBpp0111569	2e-41	157	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000340347)	1: dTCF. 2: HL03718p. 3: RE59509p. 4: RE55961p. 5: RT01139p. 6: Protein pangolin, isoform J. 7: Protein pangolin, isoforms A/H/I.
TLE1	7227.FBpp0089115	0	660	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000365682)	1: Protein groucho. 2: Enhancer of split m9/10 protein.
MYC	7227.FBpp0070501	7e-10	60.5	(Homo sapiens) (STRING-Protein-Id:9606.ENSP00000367207)	1: dMyc1. 2: Diminutive. 3: Myc protein. 4: Diminutive protein.

Conserved interactions

Query node	Query node	Target node	Target node	Target confidence
WNT1	A2MR	7227.FBpp0079060	7227.FBpp0087868	0.174
WNT1	FZD1	7227.FBpp0079060	7227.FBpp0075485	0.999
FZD1	DVL1	7227.FBpp0075485	7227.FBpp0073310	0.999
DVL1	AXIN1	7227.FBpp0073310	7227.FBpp0084919	0.997
DVL1	GSK3B	7227.FBpp0073310	7227.FBpp0070454	0.929
DVL1	APC	7227.FBpp0073310	7227.FBpp0084720	0.938
DVL1	CTNNB1	7227.FBpp0073310	7227.FBpp0089031	0.953
AXIN1	GSK3B	7227.FBpp0084919	7227.FBpp0070454	0.999
AXIN1	APC	7227.FBpp0084919	7227.FBpp0084720	0.996
AXIN1	CTNNB1	7227.FBpp0084919	7227.FBpp0089031	0.997
GSK3B	APC	7227.FBpp0070454	7227.FBpp0084720	0.991
GSK3B	CTNNB1	7227.FBpp0070454	7227.FBpp0089031	0.999
APC	CTNNB1	7227.FBpp0084720	7227.FBpp0089031	0.99
CTNNB1	TCF7	7227.FBpp0089031	7227.FBpp0111569	0.999
TCF7	TLE1	7227.FBpp0111569	7227.FBpp0089115	0.998
TCF7	MYC	7227.FBpp0111569	7227.FBpp0070501	0.791

Interactions in query network but not in target network

Query node	Query node	Target node	Target node
A2MR	FZD1	7227.FBpp0087868	7227.FBpp0075485

Interactions in target network but not in query network

Target node	Target node	Target confidence	Query node	Query node
7227.FBpp0079060	7227.FBpp0073310	0.999	WNT1	DVL1
7227.FBpp0079060	7227.FBpp0084919	0.993	WNT1	AXIN1
7227.FBpp0079060	7227.FBpp0070454	0.998	WNT1	GSK3B
7227.FBpp0079060	7227.FBpp0084720	0.742	WNT1	APC
7227.FBpp0079060	7227.FBpp0089031	0.999	WNT1	CTNNB1
7227.FBpp0079060	7227.FBpp0111569	0.999	WNT1	TCF7
7227.FBpp0079060	7227.FBpp0089115	0.299	WNT1	TLE1
7227.FBpp0079060	7227.FBpp0070501	0.842	WNT1	MYC
7227.FBpp0075485	7227.FBpp0084919	0.936	FZD1	AXIN1
7227.FBpp0075485	7227.FBpp0070454	0.989	FZD1	GSK3B
7227.FBpp0075485	7227.FBpp0084720	0.723	FZD1	APC
7227.FBpp0075485	7227.FBpp0089031	0.974	FZD1	CTNNB1
7227.FBpp0075485	7227.FBpp0111569	0.877	FZD1	TCF7
7227.FBpp0075485	7227.FBpp0089115	0.276	FZD1	TLE1
7227.FBpp0075485	7227.FBpp0070501	0.346	FZD1	MYC
7227.FBpp0073310	7227.FBpp0111569	0.854	DVL1	TCF7
7227.FBpp0073310	7227.FBpp0070501	0.605	DVL1	MYC
7227.FBpp0084919	7227.FBpp0111569	0.989	AXIN1	TCF7
7227.FBpp0084919	7227.FBpp0089115	0.201	AXIN1	TLE1
7227.FBpp0084919	7227.FBpp0070501	0.835	AXIN1	MYC
7227.FBpp0070454	7227.FBpp0111569	0.962	GSK3B	TCF7
7227.FBpp0070454	7227.FBpp0089115	0.152	GSK3B	TLE1
7227.FBpp0070454	7227.FBpp0070501	0.953	GSK3B	MYC
7227.FBpp0084720	7227.FBpp0111569	0.933	APC	TCF7
7227.FBpp0084720	7227.FBpp0070501	0.486	APC	MYC
7227.FBpp0089031	7227.FBpp0089115	0.928	CTNNB1	TLE1
7227.FBpp0089031	7227.FBpp0070501	0.639	CTNNB1	MYC
7227.FBpp0089115	7227.FBpp0070501	0.887	TLE1	MYC

Figure 3.4: NATALIEQ alignment tables for the alignment of the Wnt query network against the target PPI network (STRING) of *D. melanogaster* using the *sequence only* score function at E-value cut-off 1. Blue entries are links to the STRING database.

Availability and requirements

- Project name: NATALIEQ
- Project home page: http://www.ibi.vu.nl/programs/natalieq/
- Operating system(s): Platform independent
- Programming language: PHP and Perl
- Other requirements: modern web browser (Internet Explorer 9 or higher, Firefox, Chrome or Safari)
- Any restrictions to use by non-academics: no license required

Competing interests. The authors declare they have no competing interests.

Authors' contributions. NATALIEQ was conceived by all authors. MEK, BWB and GWK designed and implemented the web interface and processed the data. All authors contributed to the writing of the manuscript and approved the final manuscript.

Acknowledgments. We thank Sonja Boas for providing crucial insights on the Wnt signaling pathway case study. BWB was supported by the University of Amsterdam under the research priority area "Oral Infections and Inflammation".

Chapter 4

The paralog mapping problem

Published as:

M. El-Kebir[†], T. Marschall[†], I. Wohlers[†], M. Patterson, J. Heringa, A. Schönhuth, and G. W. Klau. Mapping proteins in the presence of paralogs using units of coevolution. *BMC Bioinformatics*, 14(Suppl 15):S18, 2013.

[†]joint first authorship

Abstract

Background: We study the problem of mapping proteins between two protein families in the presence of paralogs. This problem occurs as a difficult subproblem in coevolution-based computational approaches for protein-protein interaction prediction.
Results: Similar to prior approaches, our method is based on the idea that coevolution implies equal rates of sequence evolution among the interacting proteins, and we provide a first attempt to quantify this notion in a formal statistical manner. We call the units that are central to this quantification scheme the *units of coevolution*. A unit consists of two mapped protein pairs and its score quantifies the coevolution of the pairs. This quantification allows us to provide a maximum likelihood formulation of the paralog mapping problem and to cast it into a binary quadratic programming formulation.
Conclusion: CUPID, our software tool based on a Lagrangian relaxation of this formulation, makes it, for the first time, possible to compute state-of-the-art quality pairings in a few minutes of runtime. In summary, we suggest a novel alternative to the earlier available approaches, which is statistically sound and computationally feasible.

4.1 Introduction

Protein-protein interactions are essential for understanding cellular mechanisms and their malfunctioning in disease [107]. Both experimental and computational methods exist for their prediction [200]. Among the latter, many are based on the observation that interacting proteins often have coevolved due to a positive selection

pressure on preserving the interaction [49, 81, 205, 223]. This observation allows to predict protein-protein interactions by quantifying the degree of similarity between the evolution of two protein families. Coevolution-based methods map proteins across the families in order to maximize a similarity measure between the phylogenetic trees or the underlying distance matrices. In settings with only orthologous proteins (e.g. [116], a study on coevolution in prokaryotes), the mapping task is trivial as every protein family contains only one protein per species. In the presence of paralogous proteins (paralogs), however, the mapping task becomes difficult.

There are only a handful of existing approaches for the *paralog mapping problem* [93, 111, 197]. Izarzugaza et al. [111], in their method TAG-TSEMA, and most earlier approaches establish mappings by swapping rows and columns of the distance matrices to achieve similarity between the matrices. Tillier et al. [197] take a different approach in their method MMM by heuristically determining submatrices of the two distance matrices to be paired. The recent approach TreeTop by Hajirasouliha et al. [93] computes mappings by comparing two phylogenetic trees derived from the multiple sequence alignments using dynamic programming. Compared to the matrix-based method [111] this yields a speed-up of several orders of magnitude, which, however, comes at the expense of significantly reduced, incomplete mappings.

Here, we present a new mathematical model and method, which are based on statistically quantifying the degree of coevolution reflected by a mapping. Similar to prior approaches, our method is based on the idea that coevolution implies equal rates of sequence evolution among the interacting proteins, and we provide a first attempt to quantify this notion in a formal statistical manner. We call the units that are central to this quantification scheme the *units of coevolution*. A unit consists of two mapped protein pairs and its score quantifies the coevolution of the pairs. The quality of a mapping is then rated in terms of the units of coevolution it consists of. We establish and exploit a connection to the global network alignment problem and are thus able to find provably near-optimal or optimal mappings. Due to the design of our quality scores, an optimal mapping corresponds to a maximum likelihood estimate of a generative statistical model built upon the participating units of coevolution. We extend a recent Lagrangian relaxation approach for network alignment [72] to deal with the new scoring scheme. We apply our method to an approved benchmark of coevolving protein domains. In terms of recall and precision, we outperform MMM, perform better than TreeTop and slightly better than TAG-TSEMA. In terms of runtime, we outperform TAG-TSEMA by an order of magnitude, are faster than MMM and much slower than TreeTop.

Our software tool CUPID (Coevolution Units Paralog Interaction Detector) as well as all data and scripts to reproduce the results are freely available as part of the NINA project for network analysis and integration at http://www.cwi.nl/research/nina.

4.2 Mathematical model

4.2.1 Units of coevolution

The data we take as input are multiple alignments of two supposedly interacting protein families. In line with previous work [93, 111, 158, 197], we assess coevolution

in terms of the differences of sequence identities derived from the multiple alignments. Here we stick to earlier practice and define sequence identity as the number of mismatches divided by the sum of matches and mismatches without counting gap columns. Given sets of sequences A and B representing the two supposedly interacting families whose members are to be paired, let a^* and b^* be common ancestral sequences of A and B, respectively. Now, we look for pairs $(a,b) \in A \times B$ such that the sequence identity between a and a^* equals the sequence identity between b and b^*. The caveat here, however, is that a^* and b^* are unknown. Hence, we cannot infer the degree of coevolution of two family members $a \in A$ and $b \in B$ by considering the pair (a,b) alone. To overcome this, we consider quadruples, i.e., pairs of pairs (a,b) and (a',b'), and assess them based on the following idea: if a and a' are significantly more similar to each other than b is to b', or vice versa, then at least one of the pairs $(a,b),(a',b')$ is likely to represent non-coevolving proteins. This is because the differences in sequence identity among each other imply different rates of divergence from the virtual, common ancestors a^* and b^*. Using a^* and b^* instead of the two most recent common ancestors is justified by the common assumption that the trees of interacting protein families are near identical [93, 111]. We call quadruples $((a,b),(a',b'))$ *units of coevolution*. The main theme of this paper is to determine a matching (i.e., a mapping) of family members that is optimal with respect to the quadruples it contains. See Figure 4.1 for an illustration and the next subsection for how to assign statistically motivated values to units of coevolution.

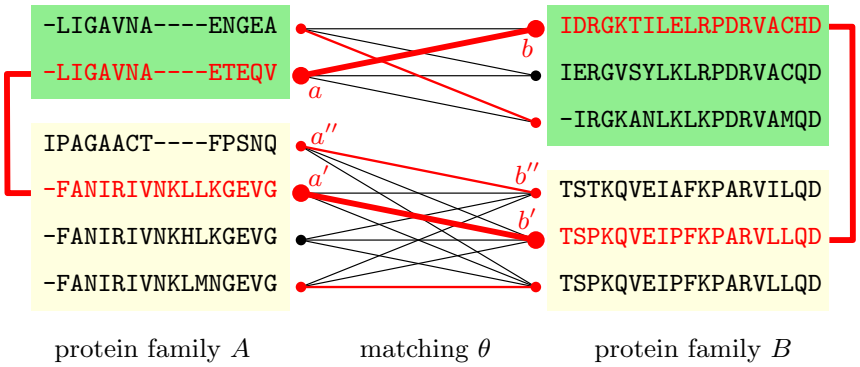


Figure 4.1: Two alignments of protein families A and B with proteins from two species, which are indicated by different background colors. Black and red nodes and edges compose the matching graph G. A matching θ is shown in red. A unit of coevolution $((a,b),(a',b'))$ within θ is highlighted in bold. For this toy example, we have $\ell_A(a,a') = 12$ (matches + mismatches), $\Delta_A(a,a') = 11$ (mismatches), $\ell_B(b,b') = 19$ and $\Delta_B(b,b') = 15$ and a resulting probability $f(\Delta_A(a,a'),\Delta_B(b,b')) = \binom{12}{11}\binom{19}{15}/\binom{31}{26} \approx 0.274$. Note the lower score of the unit $((a',b'),(a'',b''))$, which is $\binom{12}{11}\binom{19}{3}/\binom{31}{14} \approx 4.4 \cdot 10^{-5}$.

4.2.2 Maximum likelihood maximum cardinality matchings

In the following, we provide a formal definition of *units of coevolution*. Based on this, we develop a statistical model that can be interpreted as generating units of coevolution and that is parameterized by matchings. Determining an optimal matching then translates to determining the maximum likelihood estimate of the observed data. To do this, we need the following notation:

Definition 4.1 (Matching graph) *Let A and B be protein families whose members $v \in A \cup B$ are labeled by their species $s(v)$. The* matching graph *is a bipartite graph $G = (A \cup B, E)$ where $E = \{(a,b) \in A \times B \mid s(a) = s(b)\}$.*

A *matching* of G is a subset of edges such that no two edges are incident to the same node. When S is the set of all species, the mapping $s : A \cup B \to S$ used above induces partitions of A and B. We define $A_t := \{a \in A \mid s(a) = t\}$ and $B_t := \{b \in B \mid s(b) = t\}$ to refer to the respective parts of species t. Because G consists of $|S|$ connected components, which are complete bipartite subgraphs, all maximal matchings of G have the same cardinality

$$n = \sum_{t \in S} \min\{|A_t|, |B_t|\} \ .$$

Now, we define our search space as follows.

Definition 4.2 (Search space) *The* search space Θ *is the set of matchings of maximum cardinality n.*

Next, we develop a parametrized statistical model whose parameters can be identified with the search space Θ. As pointed out above, a maximum likelihood estimate $\theta^* \in \Theta$ then corresponds to an optimal matching and hence an optimal pairing of putatively coevolving family members. Let $\Delta_A(a,a')$ be the number of sequence mismatches between a and a' and let $\ell_A(a,a')$ be the number of sequence matches and mismatches between a and a' in the multiple alignment A. See Figure 4.1 for an example.

We make two simplifying assumptions to derive a suitable problem formulation. First, we assume a hidden substitution rate $p_{a,a'}$ for each pair of sequences $a, a' \in A$ such that the observed quantity of $\Delta_A(a,a')$ follows a binomial distribution with parameter $p_{a,a'}$. That is, we model mismatches by independent Bernoulli trials with probability $p_{a,a'}$. We make the analogous assumption for all $b, b' \in B$. Therefore, if a interacts with b and a' with b', observing numbers $\Delta_A(a,a')$ and $\Delta_B(b,b')$ together is described by a hypergeometric distribution. Formally, the probability for observing $\Delta_A(a,a')$ and $\Delta_B(b,b')$ given $\ell_A(a,a')$, $\ell_B(b,b')$, and $\Delta_A(a,a') + \Delta_B(b,b')$ is given by

$$\begin{aligned} f\big(\Delta_A(a,a'), \Delta_B(b,b')\big) &= P_H\big(\Delta_A(a,a'), \Delta_B(b,b') \,\big|\, \ell_A(a,a'), \ell_B(b,b'), \\ &\qquad\qquad \Delta_A(a,a') + \Delta_B(b,b')\big) \\ &= \frac{\begin{pmatrix} \ell_A(a,a') \\ \Delta_A(a,a') \end{pmatrix} \begin{pmatrix} \ell_B(b,b') \\ \Delta_B(b,b') \end{pmatrix}}{\begin{pmatrix} \ell_A(a,a') + \ell_B(b,b') \\ \Delta_A(a,a') + \Delta_B(b,b') \end{pmatrix}} \ , \end{aligned} \tag{4.1}$$

where H is the assumption of equal evolutionary rates due to coevolution.

Definition 4.3 (Unit of coevolution) *We refer to* (4.1) *as the value of the unit of coevolution* $((a,b),(a',b'))$.

We now assume that all units of coevolution are independent. The overall likelihood of a matching θ is thus

$$f\big(\Delta_A,\Delta_B;\theta\big) = \prod_{\substack{(a,b),(a',b')\in\theta\\(a,b)<(a',b')}} f\big(\Delta_A(a,a'),\Delta_B(b,b')\big) \ , \tag{4.2}$$

where "$<$" is an arbitrary ordering on E.

The independence assumption may, at first glance, appear unjustified because a pair (a,b) can take part in many units of coevolution. Note, however, first that it is equivalent to maximize $\sqrt[(n-1)/2]{f(\Delta_A,\Delta_B;\theta)}$ instead of (4.2) where n is the size of the matching θ. Rewriting

$$\sqrt[(n-1)/2]{f\big(\Delta_A,\Delta_B;\theta\big)} = \prod_{(a,b)\in\theta} C(a,b;\theta)$$

where

$$C(a,b;\theta) := \sqrt[n-1]{\prod_{(a',b')\in\theta,(a',b')\neq(a,b)} f(\Delta_A(a,a'),\Delta_B(b,b'))}$$

which one can—as the (harmonic) mean of all units of coevolution (a,b) takes part in—interpret as a measure for the degree of coevolution of the individual pair (a,b). It is now reasonable to believe that the degrees of coevolution of (a,b) and (a',b') are independent of one another: This clearly applies if the two pairs stem from two different species (that is, a is orthologous to a' and b is orthologous to b'), because there is usually no genetic crosstalk across species, at least not in eukaryotes. Even in the case of a being paralogous to a' and b being paralogous to b', the assumption of independence may be reasonable, because paralogs often assume functions that considerably diverge from their paralogous partners, hence are subject to independent selective pressures. So, one can decompose (4.2) into factors, for which the assumption of independency makes sense, while each factor has a reasonable interpretation. This may justify the assumption of independency overall.

The problem is now as follows.

Problem 4.1 (Maximum likelihood maximum cardinality matching) *Let A and B be two protein families whose proteins $v \in A \cup B$ are labeled by their species $s(v)$, let G be the corresponding bipartite graph and let Θ be the set of maximum cardinality matchings as given in Definitions 4.1 and 4.2, respectively. Then, the goal is to find the maximum likelihood matching*

$$\theta^* = \operatorname*{arg\,max}_{\theta\in\Theta} f\big(\Delta_A,\Delta_B;\theta\big) \ .$$

4.3 Method

We start by formulating the problem as a binary quadratic program (BQP). For notational convenience, we switch from using $a,a' \in A$ and $b,b' \in B$ to using $i,j \in A$ and

$k,l \in B$. As a first step, we take the logarithm of (4.2), which yields the log likelihood

$$\log f\big(\Delta_A, \Delta_B; \theta\big) = \sum_{\substack{(i,k),(j,l)\in\theta \\ (i,k)<(j,l)}} \log f\big(\Delta_A(i,j), \Delta_B(k,l)\big) \ . \tag{4.3}$$

We represent a matching θ by binary variables x_{ik} which are equal to 1 if and only if the edge (i,k) is in θ. As a shorthand we use $f_{ijkl} = \log f\big(\Delta_A(i,j), \Delta_B(k,l)\big)$. Now the corresponding quadratic program is

$$\max_x \quad \sum_{\substack{i,j \\ i<j}} \sum_{\substack{k,l \\ k\neq l}} f_{ijkl} x_{ik} x_{jl} \tag{BQP-1}$$

$$\text{s.t.} \quad \sum_k x_{ik} \leq 1 \qquad \forall i \tag{4.4}$$

$$\sum_i x_{ik} \leq 1 \qquad \forall k \tag{4.5}$$

$$\sum_{i,k} x_{ik} = n \tag{4.6}$$

$$x_{ik} = 0 \qquad \forall i,k, s(i) \neq s(k) \tag{4.7}$$

$$x_{ik} \in \{0,1\} \qquad \forall i,k \tag{4.8}$$

Constraints (4.4) and (4.5) are the standard constraints for bipartite matching. Equality (4.6) ensures that the matching will have maximum cardinality. Constraints (4.7) ensure that only proteins of the same species are mapped. The quadratic objective function scores the contribution of units of coevolution, which may consist of protein pairs that belong to different species. We formally show how to transform this integer linear programming formulation into a well-studied formulation used for the Quadratic Assignment Problem [84] and for network alignment [72, 126].

To this end, we eliminate constraint (4.6) by shifting all f_{ijkl} by an offset $K > 0$ such that they become strictly positive. Correcting for this in the objective function leads to

$$\max_x \quad \sum_{\substack{i,j \\ i<j}} \sum_{\substack{k,l \\ k\neq l}} (f_{ijkl} + K) x_{ik} x_{jl} - \binom{n}{2} \cdot K \quad \text{s.t.} \quad (4.4), (4.5), (4.7) \text{ and } (4.8). \tag{BQP-2}$$

(BQP-1) and (BQP-2) are the same as shown in the following lemma.

Lemma 4.1 *A solution $\theta \in \Theta$ is optimal to* (BQP-1) *if and only if it is optimal to* (BQP-2). *Furthermore, the objective value of θ in* (BQP-1) *is equal to the objective value of θ in* (BQP-2).

Proof Let θ_1 be an optimal solution to (BQP-1) and θ_2 an optimal solution to (BQP-2). Let $G = (A \cup B, E)$ be the matching graph as introduced in Def. 4.1.

We start by showing that $|\theta_1| = |\theta_2| = n$. By constraint (4.6), we have that $|\theta_1| = n$. To prove $|\theta_2| = n$, we recall that G consists of connected components induced by $A_t \cup B_t$ for $t \in S$, each of which is a complete bipartite subgraph. Suppose that θ_2 is

not maximal, i.e., $|\theta_2| < n$. Observe that every component $A_t \cup B_t$ can have at most $\min\{|A_t|,|B_t|\}$ matched nodes in θ_2. As $n = \sum_{t\in S} \min\{|A_t|,|B_t|\}$ and $|\theta_2| < n$, there must exist a component t with unmatched nodes $a \in A_t$ and $b \in B_t$. Since $f_{ijkl} + K > 0$ for all quadruples $((i,j),(k,l))$ with $i < j$ and $k \neq l$, we have that θ_2 is not an optimal solution for (BQP-2) as including (a,b) in the matching would result in a matching with a greater objective value. Therefore, it follows that $|\theta_1| = |\theta_2| = n$.

The number of quadruples, or units of coevolution, induced by any maximum cardinality matching is $\binom{n}{2}$. Therefore, any *maximum cardinality matching* that is a feasible solution to (BQP-1) and (BQP-2) has an objective value of

$$\sum_{\substack{i,j \\ i<j}} \sum_{\substack{k,l \\ k\neq l}} (f_{ijkl} + K) x_{ik} x_{jl} - \binom{n}{2} \cdot K = \sum_{\substack{i,j \\ i<j}} \sum_{\substack{k,l \\ k\neq l}} f_{ijkl} x_{ik} x_{jl} \quad . \tag{4.9}$$

As $|\theta_1| = |\theta_2| = n$, the above equality also holds for matchings θ_1 and θ_2. In addition, θ_1 is by definition feasible to (BQP-2). Conversely, θ_2 is feasible to (BQP-1) as $|\theta_2| = n$. Therefore, we have that optimal solutions to (BQP-1) and (BQP-2) have equal objective values. □

Our starting point for the Lagrangian relaxation is (BQP-2) where the weights assigned to the quadruples are strictly positive. We obtain the relaxation along the same lines as in [72]. The main resulting theorem is as follows.

Theorem 4.2 *Let* $m = \binom{\sum_{t\in S} |A_t|\cdot|B_t|}{2}$. *For any* $\lambda \in \mathbb{R}^m$, *an* upper bound *on* (BQP-2) *is given by*

$$Z_{\mathrm{LD}}(\lambda) = \max_x \quad \sum_{i,k} v_{ik}(\lambda) \cdot x_{ik} \tag{LD$_\lambda$}$$

$$\text{s.t.} \quad \sum_k x_{ik} \leq 1 \qquad \forall i \tag{4.10}$$

$$\sum_i x_{ik} \leq 1 \qquad \forall k \tag{4.11}$$

$$x_{ik} = 0 \qquad \forall i,k, s(i) \neq s(k) \tag{4.12}$$

$$x_{ik} \in \{0,1\} \qquad \forall i,k \tag{4.13}$$

where

$$v_{ik}(\lambda) = \max_y \sum_{\substack{j \\ j>i}} \sum_{\substack{l \\ l\neq k}} (w_{ijkl} + \lambda_{ijkl}) y_{ijkl} + \sum_{\substack{j \\ j<i}} \sum_{\substack{l \\ l\neq k}} (w_{ijkl} - \lambda_{jilk}) y_{ijkl} \tag{LD$_\lambda^{ik}$}$$

$$\text{s.t.} \quad \sum_{\substack{l \\ l\neq k}} y_{ijkl} \leq 1 \qquad \forall j, j \neq i \tag{4.14}$$

$$\sum_{\substack{j \\ j\neq i}} y_{ijkl} \leq 1 \qquad \forall l, l \neq k \tag{4.15}$$

$$y_{ijkl} \in \{0,1\} \qquad \forall j,l \tag{4.16}$$

and where $w_{ijkl} = (f_{ijkl} + K)/2$. *The upper bound* $Z_{\mathrm{LD}}(\lambda)$ *can be computed in time* $\mathcal{O}(n^5)$.

In the theorem above each variable y_{ijkl} refers to a unit of coevolution. Since (BQP-2) is the formulation used for global network alignment in [126] and [72], upper bound and runtime follow directly from the proof given in [72]. We obtain solutions to (LD_λ) and (LD_λ^{ik}) by solving the corresponding maximum weight bipartite matching problems. From a solution (x, y) to (LD_λ), we compute a feasible solution to (BQP-2) by using the matching encoded in x whose score is a lower bound on the value of the optimal solution to (BQP-2). The goal now is to identify λ^* which results in the smallest gap between upper and lower bound. We do this using a hybrid procedure combining subgradient optimization and a specially crafted dual descent scheme. For details we refer again to [72].

4.4 Results

4.4.1 Benchmark data set

Designing a large benchmark data set for our problem is difficult as there is insufficient information on the interaction between the individual members of protein families and the correct mapping of paralogs is thus usually unknown. We therefore rely on the reference data set of Izarzugaza et al. [111] in which the protein families are in fact domain families and the type of interaction is the co-occurrence in the same protein chain. The task is to determine a correct matching between protein domains of the same species. In this benchmark, a correct matching maps only domains that occur in the same protein chain and are therefore known to coevolve. Izarzugaza et al. [111] compiled the data set by first selecting Pfam [21] domains that co-occur in known yeast proteins and then took from these domains all eukaryotic sequences present in SwissProt which are not labeled "fragment", "hypothetic" or "putative". Finally they selected those domain pairs which (i) per family cover at least four species with at least three sequences each, (ii) in which at least 15 sequences are mapped, i.e., co-occur in a protein chain, and (iii) which have at least 50% of the sequences of the domain with fewest members mapped. The resulting benchmark instances comprise 488 pairs of multiple sequence alignments of domain families whose domains co-occur in the same protein chain. The total number of domain families in the benchmark is 604 and the number of domains per domain family ranges from 21 up to 212.

In previous work, phylogenetic trees were constructed from the alignments and either the trees themselves [93] or the distance matrices derived from them [111, 197] were compared. In contrast, our algorithm uses data from the multiple alignments directly for scoring, as detailed in the Mathematical Model section. In addition to the alignments, the species from which each sequence originates is provided as input to the algorithms. We ran the experiments for CUPID and MMM on a 2.26 GHz processor with 24 GB of RAM, running 64-bit Linux. For MMM we vary the allowance parameter a between 0.1 and 0.5. For TAG-TSEMA and TreeTop we took the numbers from [93]. Note that TAG-TSEMA was run on one of the fastest supercomputers at the time (2007/8). TreeTop was run on a similar machine as used for CUPID.

Table 4.1: The average recall and precision values in percent as well as the runtime in hours of TAG-TSEMA [111], TreeTop [93], MMM [197] and our method CUPID are shown. CUPID was terminated when either optimality was reached or a time limit of 5 minutes was hit; in the latter case, the best solution found until that time was used. TAG-TSEMA and TreeTop values are taken from [93]. MMM runs were subject to a time limit of 1 hour; the number of instances solved within this time limit are given in the last column. Precision and recall values are only determined for the set of solved instances. For the same set of solved instances the CUPID quality measure is given in square brackets.

	Recall	Precision	Runtime	#Instances
TAG-TSEMA [111]	**56 %**	45 %	730 h	488
TreeTop [93]	38 %	48 %	**0.02 h**	488
CUPID	**56 %**	**50 %**	30 h	488
MMM, $\alpha = 0.1$ [197]	6 %	35 %	55 h	488
MMM, $\alpha = 0.2$ [197]	15 % [61 %]	46 % [55 %]	121 h	394
MMM, $\alpha = 0.3$ [197]	26 % [70 %]	57 % [64 %]	250 h	270
MMM, $\alpha = 0.4$ [197]	35 % [71 %]	53 % [65 %]	323 h	214
MMM, $\alpha = 0.5$ [197]	37 % [70 %]	44 % [65 %]	363 h	149

4.4.2 Recall and precision

For each instance, we compute the recall and precision of the predicted matching with respect to the reference solution, which is the largest matching in which only domains of the same protein are paired, i.e., domains that are known to coevolve. Recall is defined as the percentage of correctly predicted pairings with respect to the cardinality of the reference solution. Precision is defined as the number of correctly predicted pairings divided by the cardinality of the predicted solution.

4.4.3 Solution quality and runtime

Table 4.1 lists recall and precision for TAG-TSEMA [111], TreeTop [93], MMM [197], and CUPID. For MMM we applied a wall-time limit of 1 hour per instance. The number of instances that MMM could solve within the time limit rapidly decreases with increasing α. Our method CUPID achieves a recall of 56 % and a precision of 50 %, improving on the other methods. Also in comparison with MMM, CUPID achieves higher recall and precision on the subset of instances that were solved by MMM for varying values of α. Further, CUPID outperforms TAG-TSEMA by an order of magnitude in terms of runtime. TreeTop is much faster than CUPID (0.02 h as compared to 30 h) at the expense of a substantially worse recall (38 % compared to 56 %).

CUPID terminates if either a maximum runtime is reached or the optimal solution has been found. If the time limit is hit, it returns a feasible solution and an upper bound on the optimal score. By definition, the score of the returned solution is a lower bound on the optimal score. We define the *relative gap* as the difference

Table 4.2: Effect of time limit on solution quality of CUPID.

Time limit	10 sec	30 sec	1 min	5 min	10 min	20 min
Total runtime	1.3 h	3.8 h	7.3 h	30.2 h	51.6 h	81.0 h
Precision	46.8 %	47.8 %	48.2 %	49.6 %	49.8 %	50.3 %
Recall	52.6 %	53.7 %	54.4 %	55.9 %	56.2 %	56.7 %
Median relative gap	10.4 %	5.4 %	3.1 %	2.1 %	1.7 %	1.3 %
Instances solved optimaly	6.1 %	9.4 %	11.9 %	16.0 %	16.8 %	17.0 %

between upper and lower bound relative to the absolute value of the lower bound. To determine a good maximum runtime, we ran CUPID on all instances with maximum single-CPU-core runtimes of 10 sec, 30 sec, 1 min, 5 min, 10 min, and 20 min. Table 4.2 summarizes the effect on solution quality in terms of precision, recall, median relative gap size, and the number of instances solved to optimality. These results confirm that precision and recall increase with maximum runtime, while the median relative gap size decreases. This converging behavior suggests that our scoring function correlates well with precision and recall and that our algorithm is robust with respect to the choice of the time limit. Based on Table 4.2, we decided that stopping after 5 min represents a good trade-off between runtime and solution quality. By increasing the runtime from 5 min to 20 min, recall and precision both increase only by less than one percentage point. On the other hand, going from 5 min to 1 min, recall and precision both drop by more than 1.4 percentage points.

When setting the maximum runtime to 5 min, all 488 instances were solved in a total runtime of 30.2 h, out of which 78 instances were solved to optimality (16.0 %). The median relative gap was 2.1 %, which indicates that our method is able to identify matchings with a likelihood close to the maximum likelihood in many cases. Figure 4.2 displays a histogram of the observed relative gap. For most instances it is small, but for a few instances it constitutes more than 50 % of the likelihood of the returned solution.

4.4.4 Scoring function assessment

Using the proven near-optimality of most of our solutions, we can assess the scoring function that we introduced in the Mathematical Model section. We relate the log likelihood of the reference matching to the log likelihood of our computed matching. To this end, we normalize the log likelihood of a matching such that it corresponds to the average log likelihood of a unit of coevolution. The results are displayed in Figure 4.3.

For instances below the bisecting line, our matching has smaller average log likelihood than the reference matching. For 64 out of the 488 instances, this applies with a difference in log likelihood of more than 0.5. This can have two reasons. First, CUPID might fail to compute a good matching, which is possible if the gap is large. Indeed, 27 out of these 64 instances have a relative gap larger than 20 %, see Figure 4.3(b). The second reason for a reference log likelihood larger than our solution's

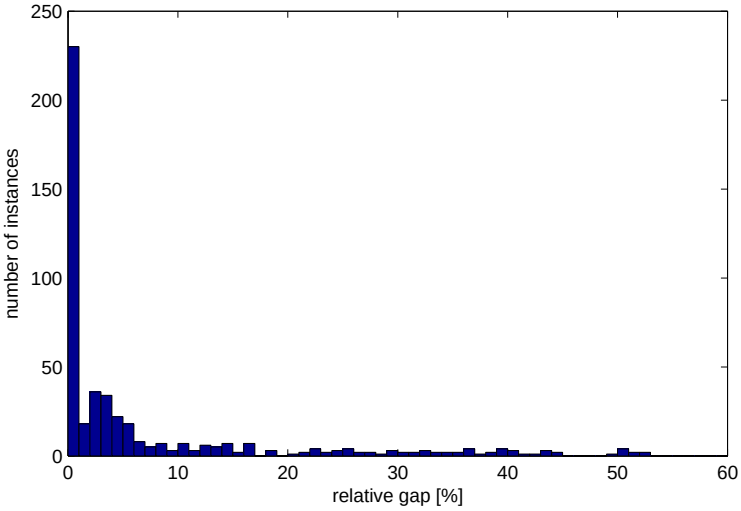
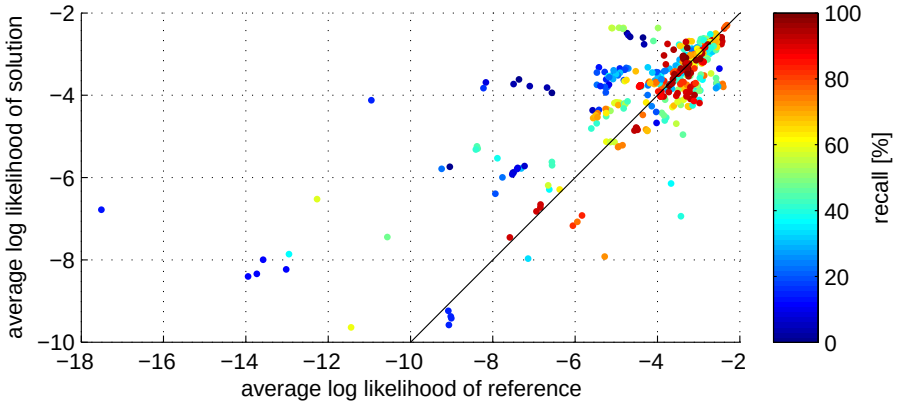


Figure 4.2: Distribution of the relative gap in percent for the 488 instances.

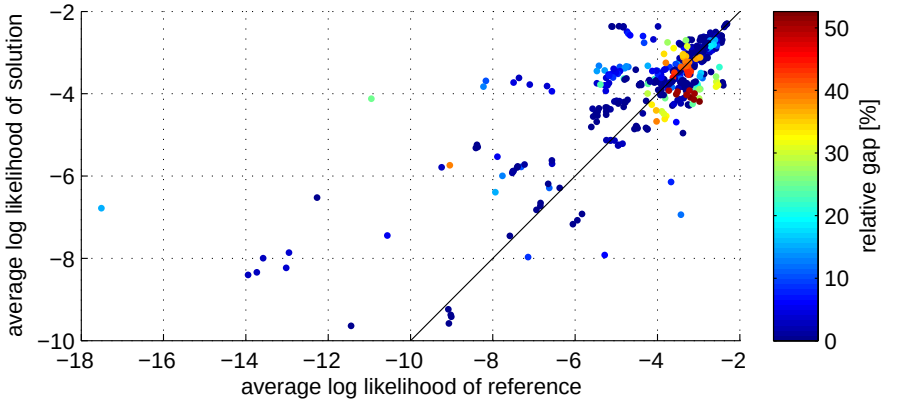
log likelihood lies in different cardinalities of the reference matching and our solution. In these instances, a smaller matching size leads to a larger average log likelihood. Since CUPID determines maximum cardinality matchings, it cannot obtain an average log likelihood as large as the one of the reference matching, even if it solves an instance to optimality. The performance on these instances can only be improved by allowing for smaller matchings.

Instances for which the average log likelihood of our solution is larger than the average log likelihood of the reference matching are located above the bisecting line in Figure 4.3. For 127 out of the 488 instances, this applies with a difference in log likelihood of more than 0.5. These are instances for which the reference matching is not the matching with the highest likelihood according to the data. This can have two reasons. First, our maximum likelihood model might need to be refined. Second, the data, i.e. the multiple alignments, might be insufficient or not accurate enough to distinguish a correct from an incorrect matching. We consider the latter issue to be the more significant one as obtaining multiple alignments that accurately reflect evolutionary history is a difficult problem.

Instances close to the bisecting line are favorable instances for our scoring and algorithm. There the solution and reference matchings have similar log likelihood. In total, for 297 of the 488 instances, the difference between these two log likelihoods is at most 0.5. These are the instances for which we indeed obtain a large recall as indicated in Figure 4.3(a) by the accumulation of red points near the bisecting line. In fact, these 297 instances have an average recall of 62.5 % while it is 46.3 % for the remaining instances, which is a significant difference ($p < 10^{-10}$ according to a Wilcoxon test).



(a) recall



(b) relative gap size

Figure 4.3: The plots show the quality of the scoring function as measured by the average log likelihood of a unit of coevolution in our solutions versus the average log likelihood of a unit of coevolution in the reference matchings. Points are colored according to (a) recall and (b) relative gap size.

4.5 Conclusions and discussion

In this article, we introduce a novel approach for predicting a matching of proteins in the presence of paralogs given multiple sequence alignments of two protein families. Our line of reasoning is centered around *units of coevolution*, which we identify as the minimal units of evidence for coevolution. Several properties distinguish our approach CUPID from previous ones. First, we employ a generative statistical model and score putative matchings based on their likelihood. Second, we make use of a close connection to the network alignment problem to compute provably near-maximum or maximum likelihood matchings. We observe a median relative tightness

of these bounds as small as 2.1% while limiting the runtime to at most 5 minutes per instance. Third, on a commonly-used benchmark data set, CUPID performs better than three state-of-the-art methods in terms of recall and precision.

Bounds on the optimal score facilitate drawing conclusions on the quality of the scoring function. We can attribute false predictions to weaknesses of the scoring function, while for heuristic methods they could also be caused by a failure to find a good, high-scoring solution.

Our analysis shows that for many instances a matching that does not have maximum cardinality will likely result in a larger average log likelihood for a unit of coevolution. Further, reference matchings usually do not have maximum cardinality. Recall and especially precision of the predicted matching can thus be improved by allowing matchings of smaller cardinality. This could be addressed, for example, by introducing constraints into our optimization scheme that influence the matching size. Subsequently, one could apply model selection approaches to predict the size of the true matching.

So far, we have restricted ourselves to the quantities $\Delta_A(a,a')$ and $\ell_A(a,a')$ to assess sequence identity, as done previously. The corresponding scoring model is very simple and depends greatly on the quality of the underlying multiple sequence alignment, which is error-prone. We therefore consider exploring the effect of using different alignment methods and other, more fine-grained, scoring models as an interesting topic for future research. For example, we expect that results improve if alignment features such as secondary structure, amino acid substitution type or alignment confidence (using e.g. the head-or-tails [135] or GUIDANCE score [159]) are quantified and considered during the mapping. By doing so, relatively well-conserved alignment regions that are likely to participate in an interaction that is shared family-wide are upweighted. Using our current model, we could straightforwardly use only selected alignment columns for scoring a unit of coevolution, for example those with alignment confidence higher than a threshold. In order to weigh alignment columns, the scoring model would need to be revised.

Inspired by a discussion in Tillier et al. [197], another possible extension is to allow many-to-many instead of only one-to-one mappings. The scoring based on units of coevolution could immediately be adapted to such a situation. However, adapting the Lagrangian relaxation approach is less straightforward and requires more effort.

As a closing remark, we recall that mapping paralogs is only a small ingredient to the successful prediction of protein-protein interaction networks, which remains a challenging and interesting field of research.

Competing interests. The authors declare that they have no competing interests.

Authors' contributions. MEK, TM, IW, AS and GWK conceived and developed the method and designed the experiments. MEK implemented CUPID. MEK, TM, IW and MP carried out and analyzed the experiments. AS and GWK guided the research. All authors drafted, read and approved the final manuscript.

Acknowledgments. We thank SARA Computing and Networking Services (www.sara.nl) for their support in using the Lisa Compute Cluster. We also thank the anonymous referees for their insightful comments and Elisabeth Tillier for valuable help with MMM.

Declarations. Publication of this article was funded by Centrum Wiskunde & Informatica.

Part II

Modules

Chapter 5

The maximum-weight connected subgraph problem

In submission:

M. El-Kebir and G. W. Klau. Solving the Maximum-Weight Connected Subgraph Problem to Optimality. Presented at the 11th DIMACS Challenge workshop, 4/5 Dec 2014, Providence (RI), U.S.A.

Abstract

Given an undirected node-weighted graph, the Maximum-Weight Connected Subgraph problem (MWCS) is to identify a subset of nodes of maximal sum of weights that induce a connected subgraph. MWCS is closely related to the well-studied Prize-Collecting Steiner Tree problem and has many applications in different areas, including computational biology, network design and computer vision. The problem is NP-hard and even hard to approximate within a constant factor. In this work we describe an algorithmic scheme for solving MWCS to provable optimality, which is based on preprocessing rules, new results on decomposing an instance into its biconnected and triconnected components and a branch-and-cut approach combined with a primal heuristic. We demonstrate the performance of our method on the benchmark instances of the 11th DIMACS implementation challenge consisting of MWCS as well as transformed PCST instances.

Keywords: maximum-weight connected subgraph, algorithm engineering, divide-and-conquer, SPQR tree, prize-collecting Steiner tree, branch-and-cut

5.1 Introduction

We consider the Maximum-Weight Connected Subgraph problem (MWCS). Given an undirected node-weighted graph, the task is to find a subset of nodes of maximal sum of weights that induce a connected subgraph. A formal definition of the unrooted and rooted variant is as follows.

Definition 5.1 (MWCS) *Given an undirected graph $G = (V,E)$ with node weights $w : V \to \mathbb{R}$, find a subset $V^* \subseteq V$ such that the induced graph $G[V^*] := \left(V^*, E \cap \binom{V^*}{2}\right)$ is connected and the weight $w(G[V^*]) := \sum_{v \in V^*} w(v)$ is maximal.*

Definition 5.2 (R-MWCS) *Given an undirected graph $G = (V,E)$, a node set $R \subseteq V$ and node weights $w : V \to \mathbb{R}$, find a subset $V^* \subseteq V$ such that $R \subseteq V^*$, the induced graph $G[V^*] := \left(V^*, E \cap \binom{V^*}{2}\right)$ is connected and the weight $w(G[V^*]) := \sum_{v \in V^*} w(v)$ is maximal.*

Johnson mentioned MWCS in his NP-completeness column [113]. The problem and its cardinality-constrained and budget variants have numerous important applications in different areas, including designing fiber-optic networks [138], oil-drilling [102], systems biology [19, 63, 219], wildlife corridor design [61], computer vision [43] and forest planning [40].

The maximum-weight connected subgraph problem is closely related to the well-studied Prize-Collecting Steiner Tree problem (PCST) [114, 140], which is defined as follows.

Definition 5.3 (PCST) *Given an undirected graph $G = (V,E)$ with node profits $p : V \to \mathbb{R}_{\geq 0}$ and edge costs $c : E \to \mathbb{R}_{\geq 0}$, find a connected subgraph $T = (V^*, E^*)$ of G such that $p(T) := \sum_{v \in V^*} p(v) - \sum_{e \in E^*} c(e)$ is maximal.*

In [63] we described a reduction from MWCS to PCST and showed that a prize-collecting Steiner tree T in the transformed instance is a connected subgraph in the original instance with weight $p(T) - w'$, where w' is the minimum weight of a node. We also gave a simple approximation-preserving reduction from PCST to MWCS: Given an instance $(G = (V,E), p, c)$ of PCST, the corresponding instance (G', w) of MWCS is obtained by splitting each edge (v,w) in E into two edges (v,u) and (u,w), and setting the weight $w(u)$ of the introduced split vertex u to $-c(e)$.

Theorem 5.1 *A maximum-weight connected subgraph T' in the transformed instance corresponds to an optimal prize-collecting Steiner tree T in the original instance, and $w(T') = p(T)$.*

Proof We first observe that if a split vertex u is part of T', then also its neighbors v and w must be in T', otherwise $T' \setminus \{u\}$ would be a better solution. We then can simply map each split vertex back to its original edge. The solution clearly has profit $p(T) = w(T')$ and is optimal, because a more profitable subgraph with respect to p would also correspond to a higher-scoring subgraph with respect to w, contradicting the optimality of T'. □

These reductions directly imply and simplify a number of results for MWCS. For example, it follows from [73] and Theorem 5.1 that MWCS is NP-hard and even hard to approximate within a constant factor. In addition, the results in [22] provide a polynomial-time exact algorithm for MWCS for graphs of bounded treewidth.

In [63] we used the close relation to PCST to develop an exact algorithm for MWCS by running the branch-and-cut approach of Ljubic et al. [11] on the transformed instance. Backes et al. [19] presented a direct integer linear programming formulation for a variant of MWCS based only on node variables. Álvarez-Miranda et al. [11] recently introduced a stronger formulation based on the concept of node-separators.

Here, we introduce an algorithm engineering approach that combines existing and new results to solve MWCS instances efficiently in practice to provable optimality. We describe new and adapted preprocessing rules in Section 5.2. Section 5.3 is dedicated to an overall divide-and-conquer scheme, which is based on novel results on decomposing an instance into its biconnected and triconnected components. In Section 5.4 we describe a branch-and-cut approach using a new primal heuristic based on an exact dynamic programming algorithm for trees. We demonstrate in Section 5.5 the performance of our approach and the benefits of preprocessing and the divide-and-conquer scheme.

5.2 Preprocessing

We describe reduction rules that simplify an instance of MWCS without losing optimality. We define three classes of increasingly complex reduction rules and apply them exhaustively in successive phases of a preprocessing scheme, see Figure 5.1.

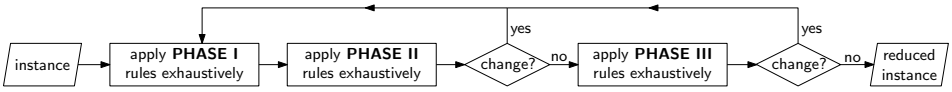


Figure 5.1: **Preprocessing scheme.** An MWCS instance passes through three phases of increasingly complex rules that are run exhaustively until no rules apply anymore. The result is a reduced instance.

The rules make use of three operations on node sets: MERGE, ISOLATE and REMOVE, see Figure 5.2. Given a node set V', MERGE (V') combines the nodes in V' into a supernode of weight $\sum_{v \in V'} w(v)$, which is connected to all neighbors of nodes in V' outside V'. Operation ISOLATE (V') adds a copy of V' without edges and merges it. Operation REMOVE (V') removes all nodes in V' from the graph. We keep a mapping from the merged nodes to sets of original nodes to map solutions of the reduced instance to solutions of the original instance. These operations will also be used in our divide-and-conquer scheme, which we will present in Section 5.3.

- **Phase I rules.** The first phase consists of three simple rules.

  1. *Remove isolated negative node rule.* Let v be an isolated vertex with $w(v) < 0$. We can safely remove v by calling REMOVE ($\{v\}$), because it will never be part of any optimal solution. Identifying all nodes that satisfy the condition takes $O(|V|)$ time.
  2. *Merge adjacent positive nodes rule.* Let (u,v) be an edge with $w(u) > 0$ and $w(v) > 0$. If one vertex will be part of the solution the other one will

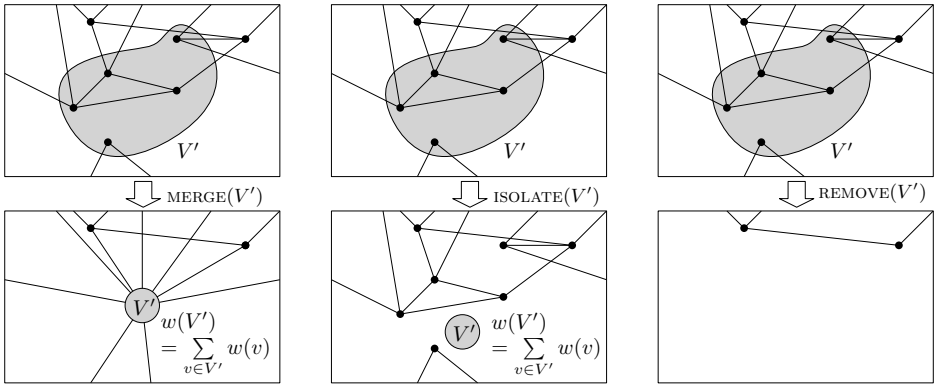


Figure 5.2: **Operations** MERGE, ISOLATE, REMOVE.

be as well, so we perform MERGE ($\{u,v\}$). Finding all adjacent positively-weighted nodes takes $O(|E|)$ time.

3. *Merge negative chain rule.* Let P be a chain of negative degree 2 vertices. It is safe to perform MERGE (P). Either none of the vertices in P will be part of an optimal solution or all of them. In the latter case P is used as a bridge between positive parts. Identifying all negatively-weighted chains takes $O(|E|)$ time.

- **Phase II rules.** The second phase consists of one rule.

1. *Dominated hubs rule.* Let $u,v \in V$ be two distinct nodes with $w(u) \leq 0$ and $w(u) \leq w(v)$. If the neighborhood of u is a subset of the neighborhood of v then we can REMOVE ($\{u\}$). The reason is that v will always be preferred over u in an optimal solution, because it is adjacent to all neighbors of u and adds more to the objective function. Finding all pairs of negatively-weighted dominated nodes takes $O(\Delta \cdot |V|^2)$ time where Δ is the maximum degree of the graph. See also Sect. 3.5 in [31].

- **Phase III rule.** The last phase consists of the most expensive rule.

1. *Least-cost rule.* This rule is adapted from the least-cost test, which was described by Duin and Volgenant [64] for the node-weighted Steiner tree problem. Let (u,v) and (v,w) be two edges in the graph, and let v have degree 2 and $w(v) < 0$. We construct a directed graph whose node set is V and whose arc set A is obtained by introducing for every edge (a,b) in G two oppositely directed arcs ab and ba. We can REMOVE ($\{v\}$), if the shortest path from u to w with respect to lengths $d(ab) := \max\{-w(b),0\}$ for all $ab \in A$ is shorter than $-w(v)$. The reason is that if u and w were to be in an optimal solution there is a better way to connect them than using v. This rule takes $O(|V'| \cdot (|E| + |V| \log |V|))$ time where V' is the set of all negative-weighted nodes having degree 2.

Algorithm 4: SolveMWCS($G = (V, E), w$)

```
1  foreach connected component C of G do
2      Preprocess (C)
3      let T_B be the block-cut vertex tree of C
4      while T_B has block B of degree 0 or 1 do
5          ProcessBicomponent (B)
6          update T_B
7      V_C = SolveUnrooted (C)
8      Merge (V_C); Remove (C \ V_c)
9  V* ← SolveUnrooted (G)
10 return V*
```

5.3 Divide-and-Conquer Scheme

We propose a three-layer divide-and-conquer scheme for solving MWCS to provable optimality. It is based on decomposing the input graph into its connected, biconnected and triconnected components. Hüffner et al. have already considered data reduction rules based on heuristically found separators of size k for the Balanced Subgraph problem [105]. Here, we present the first data reduction approach that considers all separators of size 1 and 2 in a rigorous manner by processing them using the block-cut and SPQR tree data structures.

In the first layer, we consider the connected components of the input (G, w) one-by-one, see Algorithm 4. In the next layer, we construct a block-cut vertex tree T_B for each connected component C. We process the block leaves B of T_B iteratively. Processing a block B of degree 1 will result in the removal of $B \setminus \{c\}$, where c is the corresponding cut vertex. In addition, a new degree 0 node may be introduced. Processing a block B of degree 0 will result in the replacement of B by a single isolated node. Therefore, at the end of the loop, the graph $G[C]$ will only consist of isolated nodes. Among these nodes, the node with maximum weight corresponds to the maximum weight connected subgraph of $G[C]$. We retain only this node in the graph, and remove all other nodes in C. After processing all connected components, a similar situation arises in G: each component is an isolated node, and the solution V^* will correspond to the node that has maximum weight.

Next, we describe how to process a block B. The idea here is to account for the situation where the final optimal solution V^* contains parts of B, i.e. $V^* \cap B \neq \emptyset$. For this to happen, either V^* must be a proper subset of B, or a cut node of B must be part of V^*. Since B corresponds to a degree 0 or 1 block in T_B, it contains at most one cut node c. Let us consider the case where B does have a cut node c, as the other case is straightforwardly resolved by introducing an isolated node. Two subcases can be distinguished: $c \in V^* \cap B$ and $c \notin V^* \cap B$. We encode both cases using the following gadget. Let V_1 be the unrooted maximum-weight connected subgraph of $G[B]$, and let V_2 be the maximum-weight connected subgraph of $G[B]$ rooted at c. The corresponding gadget Γ_1 is obtained by merging the nodes in V_2, and, if $V_1 \neq V_2$, by additionally introducing an isolated vertex corresponding to V_1—see Figure 5.3 D

and E. Replacing B by the gadget preserves optimality as stated in the following lemma.

Lemma 5.2 *Let $B \subseteq V$ be a block in $G = (V, E)$ containing exactly one cut node c. Let $G' = G[(V \setminus B) \cup \Gamma_1]$ be the graph where B is replaced by gadget Γ_1. A maximum weight connected subgraph of $G'[U^*]$ has the same weight as a maximum weight connected subgraph $G[V^*]$, i.e., $w(U^*) = w(V^*)$.*

Proof The gadget Γ_1 consists of two parts V_1 and V_2, which correspond to the unrooted and $\{c\}$-rooted maximum weight connected subgraph of $G[B]$, respectively. By definition V_1 and V_2 induce connected subgraphs in G. Therefore the operations MERGE (V_2) and ISOLATE (V_1)—resulting in the construction of Γ_1—combined with the optimality of V^* ensure that $w(U^*) \leq w(V^*)$.

We now distinguish two subcases: $V^* \cap B = \emptyset$ and $V^* \cap B \neq \emptyset$. Consider the first case. Since the introduction of the gadget only concerns nodes in B, we have that $w(U^*) \geq w(V^*)$. Hence, $w(U^*) = w(V^*)$.

In the other case, $V^* \cap B \neq \emptyset$, we either have that $c \notin V^* \cap B$ or $c \in V^* \cap B$. If $c \notin V^* \cap B$ then $V^* \subseteq B$. By construction of the gadget, we then have $w(V_1) = w(V^* \cap B) = w(V^*)$. Conversely, if $c \in V^* \cap B$ then $w(V_2) = w(V^* \cap B)$. Observe that $w(U^* \setminus \Gamma_1) = w(V^* \setminus B)$. Therefore $w(U^*) = w(V^*)$. □

As an optimization, we preemptively remove a leaf block B if all its nodes $v \in B \setminus \{c\}$ have nonpositive weights $w(v) \leq 0$.

In the third layer, we start by constructing an SPQR-tree T_{SPQR} of B. We then iteratively consider each triconnected component A that does not contain the cut node c and contains at least three nodes. Let $\{u, v\}$ be the cut pair of such a triconnected component A. If A consists of only negatively weighted nodes, its only purpose is to connect u with v. To find the cheapest way of doing this, we construct a directed graph whose node set is A and whose arcs are obtained by introducing for every edge (a, b) in $G[A]$ two oppositely directed arcs. We define the cost of an arc (a, b) to be $-w(b)$. The cheapest way of going from u to v now corresponds to the shortest path from u to v in the directed graph. Triconnected components that contain positively-weighted nodes are processed separately and may be replaced by gadgets of smaller size, which we describe next.

Let us consider the situation where the final solution V^* contains parts of a triconnected component A with cut nodes $\{u, v\}$, i.e., $V^* \cap A \neq \emptyset$. We can distinguish four cases: (i) $u \in V^*$, (ii) $v \in V^*$, (iii) $\{u, v\} \subseteq V^*$, and (iv) $V^* \subseteq A$. In the following we introduce a gadget Γ_2 that encodes all four cases. The first three cases correspond to finding a rooted maximum weighted connected subgraph in $G[A]$ with $\{u\}$, $\{v\}$ and $\{u, v\}$ as the root node sets, respectively. Let V_1, V_2, V_3 be the solutions sets of the three rooted maximum weight connected problems from which the respective root nodes have been removed. The fourth case corresponds to finding an unrooted maximum weight connected subgraph in $G[A]$ whose solution we denote by V_4. To encode the fourth case, we ISOLATE set V_4. As for the first three cases, we MERGE the sets $V_1 \setminus V_2$, $V_2 \setminus V_1$, $V_1 \cap V_2$ and $V_3 \setminus (V_1 \cup V_2)$ resulting in the nodes v_1, v_2, v_3 and v_4, respectively. As some of these sets may be empty, we need to take care when connecting the gadget. For instance, if $V_1 \setminus V_2 = \emptyset$ and $V_1 \cap V_2 \neq \emptyset$ then we need to connect u

Procedure ProcessBicomponent(B)

let c be the corresponding cut node, if applicable
if all v in $B \setminus \{c\}$ have $w(v) \leq 0$ **then** Remove ($B \setminus \{c\}$)
else
    let T_{SPQR} be the SPQR tree of B
    foreach triconnected component A of size > 3 not containing c **do**
        let $\{u, v\}$ be the cut pair of A
        if all v in A have $w(v) \leq 0$ **then**
            compute shortest path P from u to v
            Merge ($P \setminus \{u, v\}$); Remove ($A \setminus P$)
        else
            ProcessTricomponent (A)
        Preprocess (B); update T_{SPQR}
    $V_1 \leftarrow$ SolveUnrooted (B)
    $V_2 \leftarrow$ SolveRooted ($B, \{c\}$)
    if $V_1 = V_2$ **then** Merge (V_2); Remove ($B \setminus V_2$)
    else Isolate (V_1); Merge (V_2); Remove ($B \setminus V_2$)

directly with v_3. Also, we ensure that we do not break biconnectivity. For instance, if $V_1 \cap V_2 = \emptyset$ and $V_1 \neq \emptyset$ then we merge v_1 and u as to prevent u from becoming an articulation point. See Figure 5.4 and the pseudocode below for more details.

Lemma 5.3 *Let $A \subseteq V$ be a triconnected component in $G = (V, E)$ not containing any cut node of G. Let $G' = G[(V \setminus A) \cup \Gamma_2]$ be the graph where A is replaced by gadget Γ_2. A maximum weight connected subgraph of $G'[U^*]$ has the same weight as a maximum weight connected subgraph $G[V^*]$, i.e., $w(U^*) = w(V^*)$.*

Proof Let $\{u, v\}$ be the cut pair of A. The gadget Γ_2 encodes four node sets: V_1, V_2 and V_3 representing the rooted maximum weight connected subgraphs of $G[A]$—without their respective root nodes—rooted at $\{u\}$, $\{v\}$ and $\{u, v\}$, respectively; and V_4 representing the unrooted maximum weight connected subgraph of $G[A]$. Let $v_1 :=$ Merge($V_1 \setminus V_2$), $v_2 :=$ Merge($V_2 \setminus V_1$), $v_3 := V_1 \cap V_2$ and $v_4 := V_3 \setminus (V_1 \cup V_2)$—see Figure 5.4.

We start by proving $w(U^*) \leq w(V^*)$. Since A is a triconnected component, we have that $V_1 \setminus V_2$, $V_2 \setminus V_1$, $V_1 \cap V_2$ and $V_3 \setminus (V_1 \cup V_2)$ are connected in G. In addition, as these node sets are obtained by Merge operations only and they are pairwise disjoint, we have that $w(U^*) \leq w(V^*)$.

We distinguish two cases: $V^* \cap A = \emptyset$ and $V^* \cap A \neq \emptyset$. The first case holds, because the introduction of the gadget Γ_2 only concerns nodes in A. Therefore, $w(U^*) \geq w(V^*)$, which implies $w(U^*) = w(V^*)$. The second case, $V^* \cap A \neq \emptyset$, has the following four subcases:

1. $u \notin V^*$ and $v \notin V^*$;
   This implies that $V^* \subseteq A$. We then have $w(V_4) = w(V^* \cap A) = w(V^*)$.

Procedure ProcessTriComponent(A)

let $\{u,v\}$ be the cut pair
$V_1 \leftarrow$ SolveRooted $(A, \{u\}) \setminus \{u\}$
$V_2 \leftarrow$ SolveRooted $(A, \{v\}) \setminus \{v\}$
$V_3 \leftarrow$ SolveRooted $(A, \{u,v\}) \setminus \{u,v\}$
$V_4 \leftarrow$ SolveUnrooted (A)
Isolate (V_4)
$\Gamma_2 \leftarrow \{u,v\}$
if $V_1 \setminus V_2 \neq \emptyset$ **then** $v_1 \leftarrow$ Merge $(V_1 \setminus V_2)$; add edge (u,v_1); add v_1 to Γ_2
if $V_2 \setminus V_1 \neq \emptyset$ **then** $v_2 \leftarrow$ Merge $(V_2 \setminus V_1)$; add edge (v,v_2); add v_2 to Γ_2
if $V_1 \cap V_2 = \emptyset$ **then**
    if $V_1 \neq \emptyset$ **then** Merge $(\{u,v_1\})$; remove v_1 from Γ_2
    if $V_2 \neq \emptyset$ **then** Merge $(\{v,v_2\})$; remove v_2 from Γ_2
else
    $v_3 \leftarrow$ Merge $(V_1 \cap V_2)$; add v_3 to Γ_2
    if $V_1 \subseteq V_2$ **then** add edge (u,v_3) **else** add edge (v_1,v_3)
    if $V_2 \subseteq V_1$ **then** add edge (v,v_3) **else** add edge (v_2,v_3)
if $V_3 \setminus (V_1 \cup V_2) \neq \emptyset$ **then**
    $v_4 \leftarrow$ Merge $(V_3 \setminus (V_1 \cup V_2))$
    add v_4 to Γ_2
    add edges (u,v_4),(v,v_4)
if $V_1 \cap V_2 = \emptyset$ and $V_3 \setminus (V_1 \cup V_2) = \emptyset$ **then**
    if $V_1 \setminus V_2 \neq \emptyset$ **then** add edge (v_1,v)
    if $V_2 \setminus V_1 \neq \emptyset$ **then** add edge (v_2,u)
Remove $(A \setminus \Gamma_2)$

2. $u \in V^*$ and $v \notin V^*$;
   By optimality of V^*, we have that $w(V_1 \cup \{u\}) = w(u)+w(v_1)+w(v_3) = w(V^* \cap A)$. Since $w(U^*) \leq w(V^*)$, it follows that $w(U^*) = w(V^*)$.

3. $u \notin V^*$ and $v \in V^*$;
   Symmetric to previous subcase.

4. $u \in V^*$ and $v \in V^*$;
   There are two cases: $V_1 \cap V_2 = \emptyset$ or $V_1 \cap V_2 \neq \emptyset$. We claim that for both cases $w(V_3 \cup \{u,v\}) = w(u)+w(v)+w(v_1)+w(v_2)+w(v_3)+w(v_4) = w(V^* \cap A)$. To see this, first observe that $V_1 \cup V_2 \subseteq V_3$. In the first case we have that $w(v_3) = 0$ whereas in the second case $w(v_4) = 0$. Since $w(U^*) \leq w(V^*)$, it follows in both cases that $w(U^*) = w(V^*)$.

□

Lemmas 5.2 and 5.3 imply the correctness of our divide-and-conquer scheme.

Theorem 5.4 *Given an instance of MWCS, Algorithm 4 returns an optimal solution.*

5.4 Branch-and-Cut Algorithm

To solve the nontrivial instances within our divide-and-conquer scheme, we use a branch-and-cut approach. We obtain strong upper bounds from solving the linear programming (LP) relaxation of an integer linear programming formulation and lower bounds from an integrated primal heuristic that is guided by the optimal solution of the LP relaxation.

5.4.1 Integer linear programming formulation

We use a formulation that only used node variables for both the unrooted and the rooted MWCS problem. The formulations are equivalent to the generalized node-separator formulation described in [11].

Unrooted. Variables $\mathbf{x} \in \{0,1\}^V$ encode the presence of a node in the solution. To encode connectivity in the unrooted case, we use auxiliary variables $\mathbf{y} \in \{0,1\}^V$ that encode the presence of the root node. The ILP is as follows.

$$\max \sum_{v \in V} w_v x_v \tag{5.1}$$

$$\sum_{v \in V} y_v = 1 \tag{5.2}$$

$$y_v \leq x_v \qquad \forall v \in V \tag{5.3}$$

$$x_v \leq \sum_{u \in \delta(S)} x_u + \sum_{u \in S} y_u \qquad \forall v \in V, \{v\} \subseteq S \subseteq V \tag{5.4}$$

$$x_v \in \{0,1\} \qquad \forall v \in V \tag{5.5}$$

$$y_v \in \{0,1\} \qquad \forall v \in V \tag{5.6}$$

Constraint (5.2) states that there is exactly one root node. A node can only be the root node if it is present in the solution, which is captured by constraints (5.3). Constraints (5.4) state that a node v can only be present in the solution if for all sets S containing v, either the root node is in S, or a node in the set $\delta(S) = \{u \in V \setminus S \mid \exists v \in S : (u,v) \in E\}$ is in the solution. In the next subsection we describe how we separate these constraints.

To strengthen the formulation, we use the following additional cuts.

$$y_v = 0 \qquad \forall v \in V, w(v) < 0 \tag{5.7}$$

$$\sum_{v > u} y_v \leq 1 - x_u \qquad \forall u \in V, w(u) > 0 \tag{5.8}$$

$$x_v \leq x_u \qquad \forall (u,v) \in E, w(u) > 0, w(v) < 0 \tag{5.9}$$

$$2 \cdot x_v \leq \sum_{u \in \delta(v)} x_u \qquad \forall v \in V, w(v) < 0 \tag{5.10}$$

$$x_v \leq y_v + \sum_{u \in \delta(v)} x_u \qquad \forall v \in V \tag{5.11}$$

In (5.7) we require the root node to have a strictly positive weight. We use symmetry breaking constraints (5.8) to force the node with the smallest index to be the root

node. Constraints (5.9) state that a negatively-weighted node can only be in the solution if all its adjacent positively-weighted nodes are in the solution. In addition, the presence of a node with negative weight in the solution implies that at least two of its neighbors must be in the solution, which is modeled by constraints (5.10). Constraints (5.11) are implied by (5.4) in the case that $|S| = 1$. Adding these constraints results in a tighter upper bound in the initial node of the branch-and-bound tree.

Rooted. The rooted formulation is as follows.

$$\max \sum_{v \in V} w_v x_v \tag{5.12}$$

$$x_r = 1 \qquad \forall r \in R \tag{5.13}$$

$$x_v \leq \sum_{u \in \delta(S)} x_u \qquad \forall r \in R, v \in V \setminus R, \{v\} \subseteq S \subseteq V \setminus \{r\} \tag{5.14}$$

$$x_v \in \{0,1\} \qquad \forall v \in V \tag{5.15}$$

Constraints (5.13) enforce the presence of root nodes in the solution. The cut constraints (5.14) state that a node $v \in V \setminus R$ can only be in the solution if for any root $r \in R$ and for all supersets $S \subseteq V \setminus \{r\}$ of v it holds that a node in the set $\delta(S)$ is in the solution.

We strengthen the formulation using the following cuts.

$$x_v \leq x_u \qquad \forall (u,v) \in E, w(u) > 0, w(v) < 0 \tag{5.16}$$

$$x_v \leq \sum_{u \in \delta(v)} x_u \qquad \forall v \in V \setminus R \tag{5.17}$$

Constraints (5.16) are the same as constraints (5.9) for the unrooted case. Similarly to the unrooted formulation, constraints (5.17) correspond to manually adding cuts for the case that $|S| = 1$ in (5.14).

5.4.2 Separation

Unrooted. Similarly to [11], the separation problem in the unrooted formulation corresponds to a minimum cut problem on an auxiliary directed support graph D defined as follows: each node $v \in V$ corresponds to an arc (v_1, v_2), and each edge $(u,v) \in E$ corresponds to two arcs (u_2, v_1) and (v_2, u_1). In addition, an artificial root node r is introduced as well as arcs (r, v_1) for all $v \in V$. Given a fractional solution $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$, the arc capacities c are set as follows: $c(r, v_1) = \bar{y}_v$, $c(v_1, v_2) = \bar{x}_v$ and $c(v_2, u_1) = 1$ for all distinct $u, v \in V$. Given a node $v \in V$, we identify violated constraints by solving a minimum cut problem from r to v_2. Let C be a minimum cut set from r to v_2. In case the cut value $c(C)$ is smaller than $\bar{x}_v$, the cut set will admit a set S and $\delta(S)$ such that $\bar{x}_v > \bar{x}(\delta(S)) + \bar{y}(S) = c(C)$. We add such violated constraints to the formulation and resolve again.

Rooted. For the rooted formulation the auxiliary graph D is defined as follows: each node $v \in V \setminus R$ corresponds to an arc (v_1, v_2), and each edge $(u,v) \in E$ corresponds to two arcs (u_2, v_1) and (v_2, u_1) if both u and v not in R. For each root node $r \in R$, a

single node is introduced in D. Edges (r,v) incident to a root node $r \in R$ where $v \notin R$ correspond to an arc (r,v_1). We identify violated constraints by identifying minimum cuts between r and v_2 for all $r \in R$ and $v \in V \setminus \{r\}$.

5.4.3 Primal heuristic

As stated in Section 5.1, MWCS is solvable in polynomial time for graphs of bounded treewidth. In fact, for trees R-MWCS is solvable in linear time by first rooting the tree at a node $r \in R$ and then solving a dynamic program based on the recurrence:

$$M(v) = w(v) + \sum_{u \in \delta^+(v) \setminus R} \max\{M(u), 0\} + \sum_{u \in \delta^+(v) \cap R} M(u),$$

where $\delta^+(u)$ are the children of the node u.

Our primal heuristic transforms the input graph into a tree by considering the fractional values $\bar{\mathbf{x}}$ given by the solution of the LP relaxation. We use these values to assign an edge cost $c(u,v) = 2 - (\bar{x}_u + \bar{x}_v)$ for each edge $(u,v) \in E$. Next, we compute a minimum-cost spanning tree using Kruskal's algorithm [132]. In the unrooted MWCS case, we root the spanning tree at every positively-weighted node r and assign the solution with maximum weight to be the primal solution. This leads to running time $O(|V|^2)$. In the R-MWCS case, we only root the spanning tree once at an arbitrary vertex $r \in R$, resulting in running time $O(|V|)$.

5.4.4 Implementation details

Since CPLEX version 12.3, there is a distinction between the user cut callback and the lazy constraint callback. The latter is only called for integral solutions, see Figure 5.5. Separation of (5.4) in the case of integral $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ can be done by considering the connected components of the induced subgraph $G[\bar{\mathbf{x}}]$. Let r be the root node encoded in $\bar{\mathbf{y}}$. Recall that (5.2) ensures that there is only one root node. A connected component C of $G[\bar{\mathbf{x}}]$ that does not contain r corresponds to a violated constraint with $S := C$ and $\delta(S) := \delta(C)$. Violated constraints for R-MWCS in the case of integrality can be separated analogously.

As can be seen in Figure 5.5, CPLEX calls the user cut callback at every considered node in the branch-and-bound tree. To prevent spending too much time in the separation and to allow more time for branching, we choose not to separate violated constraints at every callback invocation. Instead we make use of a linear back-off function with an initial waiting period of 1. Upon a successful attempt, the waiting period is incremented by one, thereby gradually decreasing the time spent in separating violated constraints.

5.5 Results on DIMACS Benchmark

We implemented our algorithm in C++ using the LEMON graph library [60], the OGDF library [44] for building the SPQR tree and the CPLEX v12.6 library for implementing the branch-and-cut approach. Our software tool is called Heinz 2.0 and is available for download at http://software.cwi.nl/heinz. The code of the Heinz 2.0 software is

managed using github and publicly available under the MIT license at https://github.com/ls-cwi/heinz.

We ran all computational experiments on a 12 core Linux machine with a 2.26 GHz Intel Xeon Processor L5640 and 24 GB of RAM, using 2 threads per instance. We used all MWCS instances from the 11th DIMACS Implementation Challenge (http://dimacs11.cs.princeton.edu). These are the ACTMOD set of 8 instances from integrative network analysis in systems biology and the JMP_ALM set of 72 instances, which are based on the random Euclidean instances introduced in [114]. We also considered prize-collecting Steiner tree instances from the DIMACS benchmark, transforming them to MWCS instances using the rule given in Section 5.1. These are JMP (34 instances), CRR (80), PUCNU (18), i640 (100), H (14), H2 (14) and RANDOM (68). In total we ran computational experiments on 408 instances coming from different applications.

We ran three versions of Heinz 2.0: (i) A pure branch-and-cut approach without preprocessing, to establish a baseline, (ii) preprocessing followed by branch-and-cut, to evaluate the effects of data reduction and (iii) the divide-and-conquer scheme described in Section 5.3, to evaluate the benefits of the results described in this paper. To allow for a fair comparison, we report only results on instances for which all three methods found feasible solutions. This resulted in 271 instances. A full table of results for all these instances is in the appendix.

For each instance we recorded its size in terms of number of nodes and edges, before and after preprocessing, the best upper and lower bounds that could be found by each of the three methods within a time limit of 6 hours wall time, the running time in wall time, as well as the number of processed biconnected and triconnected components for the divide-and-conquer scheme.

Figure 5.6 shows the effect of preprocessing. We can observe that preprocessing is effective, reducing more than half of the instances to at most 84% of their original size. Some instances can even be solved by preprocessing. Figure 5.7 shows the distribution of the optimality gap for the different version of Heinz 2.0. It can be seen that while some instances are hard to solve, both preprocessing and the novel divide-and-conquer scheme provide significant improvements. Also, it can be seen that the PCST instances are harder than the MWCS instances for which all three methods achieve a median gap of 0% Figure 5.8 shows the distribution of the running times of the instances that were solved to optimality by all three methods. We can see that the divide-and-conquer scheme (median running time of 0.5 s) is faster than the branch-and-cut approach without preprocessing (median running time of 16.4 s). On the MWCS instances, the branch-and-cut approach with processing achieves the same median running time of 0.4 s as the divide-and-conquer scheme. For the PCST instances, however, the divide-and-conquer scheme has the lowest median running time (3.3 s). Moreover, the number of instances that were solved to optimality is the highest for the divide-and-conquer scheme (134), followed by the branch-and-cut approach with preprocessing (129) and the branch-and-cut approach without preprocessing (97).

5.6 Conclusions

We have presented a divide-and-conquer scheme for solving the maximum-weight connected subgraph problem to provable optimality. The scheme combines effective preprocessing with a novel decomposition data reduction approach that divides an instance into biconnected and triconnected components and solves the core pieces of an instance using branch-and-cut. This is the first time that a data reduction approach considers all separators of size 1 and 2 in a rigorous manner by processing them using the block-cut and SPQR tree data structures. We have demonstrated the performance and benefits of our scheme on the benchmark instances of the 11th DIMACS Implementation Challenge.

The scheme is modular and allows for the integration of new preprocessing rules or alternative exact algorithms to solve the core instances. We plan, for example, to evaluate a branch-and-cut approach based on an edge-based ILP formulation, which is similar to the one we used for the prize-collecting Steiner tree problem in [140]. Also, we plan to implement an FPT algorithm that can be plugged into the scheme. The modularity of our approach will make it possible to perform extensive algorithm engineering studies and to improve upon the results presented in this paper.

We also want to stress that our new data reduction approach is not specific to MWCS, but also applicable to other types of graph problems. Vice versa, techniques that have been proven useful for related problems may be beneficial for solving MWCS, and we will evaluate their integration into our scheme.

Acknowledgments. We thank the participants of the March 2014 NII Shonan Meeting *Towards the ground truth: Exact algorithms for bioinformatics research*, and in particular Christian Komusiewicz and Falk Hüffner, for helpful comments.

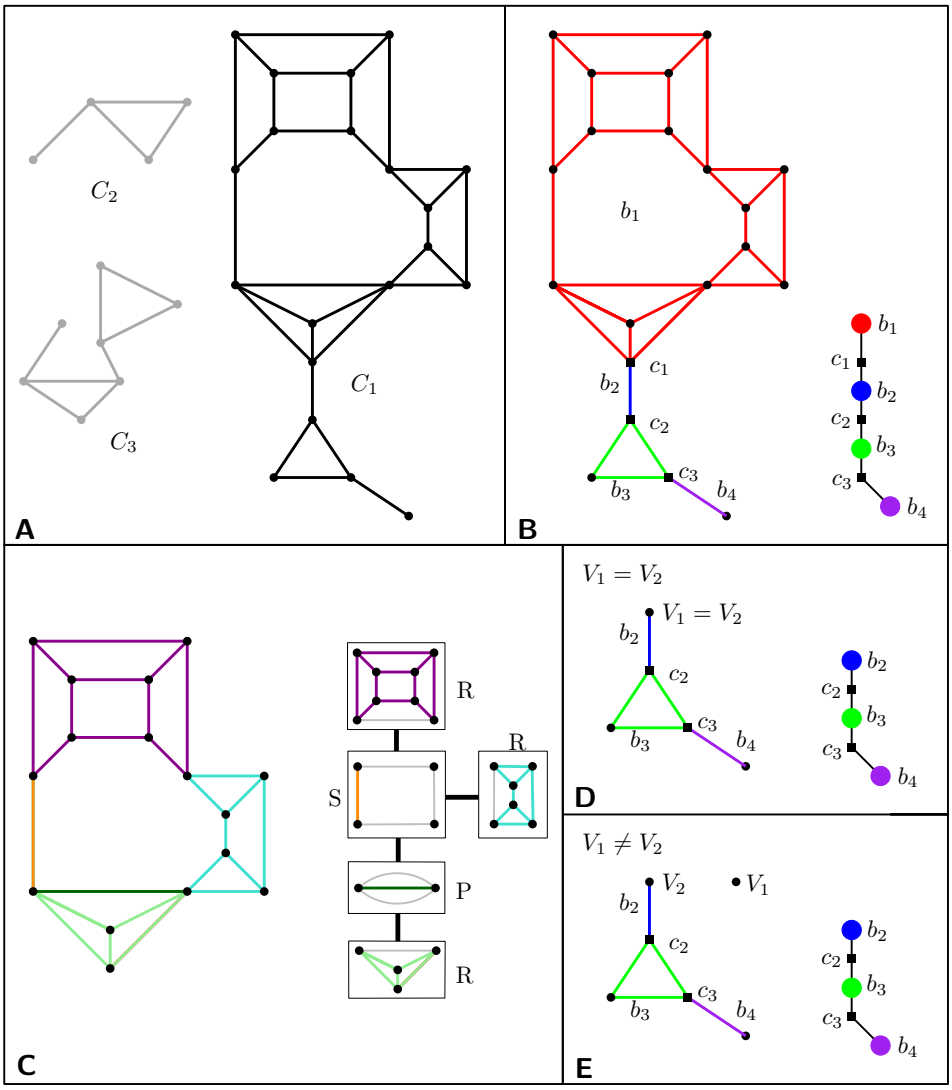


Figure 5.3: **The three layers of the divide-and-conquer scheme. A:** Three connected components of an MWCS instance. **B:** Biconnected components and the block-cut vertex tree of connected component C_1. **C:** Triconnected components and the SPQR tree of biconnected component b_1. **D:** Gadget Γ_1 in the first case. **E:** Gadget Γ_1 in the second case.

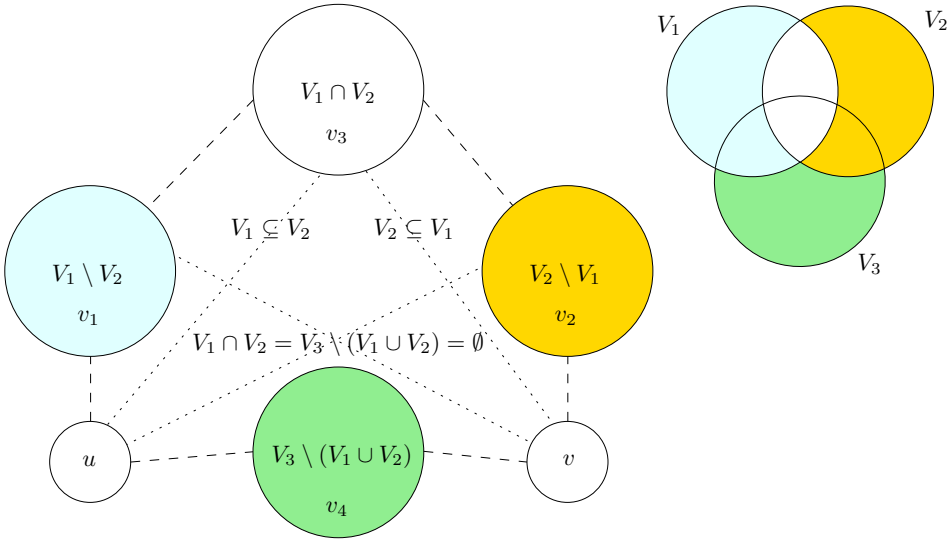


Figure 5.4: **Triconnected component gadget.** The gadget Γ_2 consists of at most four nodes. In case the corresponding set is empty no node is introduced. The dotted edges are only introduced if the condition on the edge is met, e.g., there is an edge from u to v_3 if $V_1 \subseteq V_2$ and $V_1 \cap V_2 \neq \emptyset$.

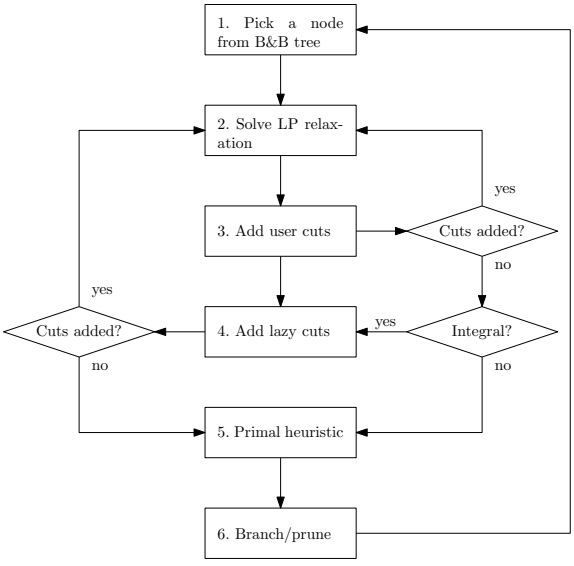


Figure 5.5: **CPLEX flow diagram.**

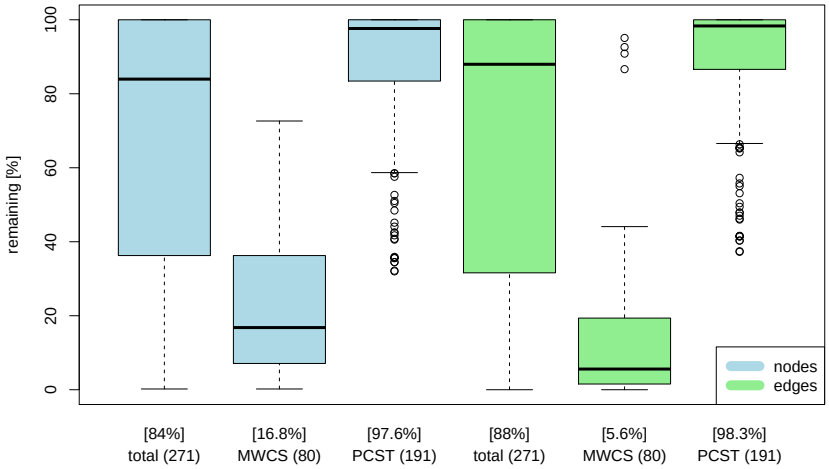


Figure 5.6: **Effect of preprocessing.** The boxplots show the reduction in number of nodes and edges after preprocessing as a fraction of the original value for the 271 instances. The median value is shown in between square brackets, and the number of instances is in between parentheses.

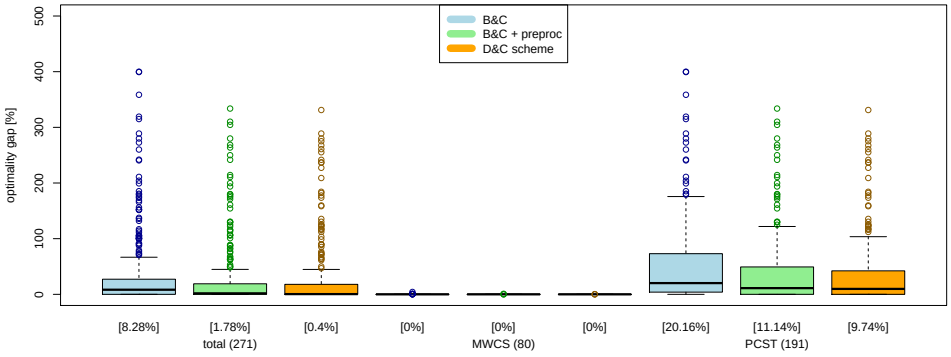


Figure 5.7: **Distribution of gaps.** Boxplots of optimality gap for the three different variants of Heinz 2.0. The median value is shown in between square brackets, and the number of instances is in between parentheses.

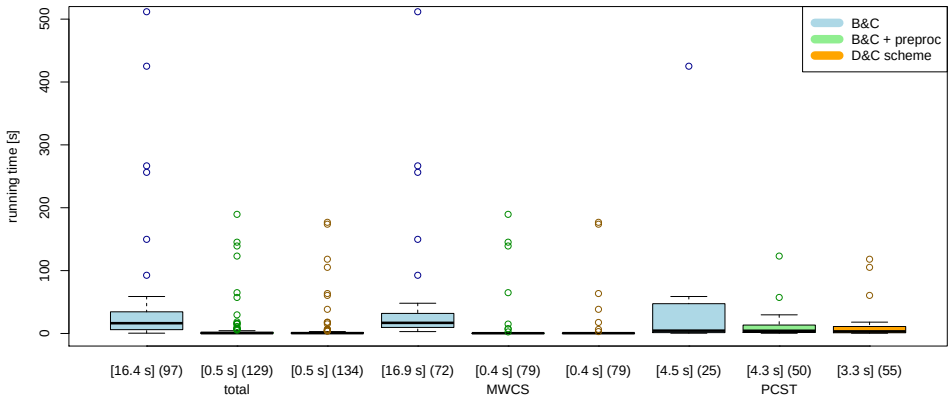


Figure 5.8: **Distribution of running times.** Boxplots of running time (s) of the instances solved to optimality by all three methods within the time limit. The median value is shown in between square brackets, and for each method the total number of instances that it solved to optimality is in between parentheses.

Chapter 6

Exploring annotated modules in networks

Published as:

K. Dinkla[†], M. El-Kebir[†], C.-I. Bucur, M. Siderius, M. J. Smit, M. A. Westenberg, and G. W. Klau. eXamine: Exploring annotated modules in networks. *BMC Bioinformatics*, 15(1):201, 2014.

[†]joint first authorship

Abstract

Background: Biological networks have a growing importance for the interpretation of high-throughput "omics" data. Integrative network analysis makes use of statistical and combinatorial methods to extract smaller subnetwork modules, and performs enrichment analysis to annotate the modules with ontology terms or other available knowledge. This process results in an annotated module, which retains the original network structure and includes enrichment information as a set system. A major bottleneck is a lack of tools that allow exploring both network structure of extracted modules and its annotations.
Results: This paper presents a visual analysis approach that targets small modules with many set-based annotations, and which displays the annotations as contours on top of a node-link diagram. We introduce an extension of self-organizing maps to lay out nodes, links, and contours in a unified way. An implementation of this approach is freely available as the Cytoscape app eXamine.
Conclusions: eXamine accurately conveys small and annotated modules consisting of several dozens of proteins and annotations. We demonstrate that eXamine facilitates the interpretation of integrative network analysis results in a guided case study. This study has resulted in a novel biological insight regarding the virally-encoded G-protein coupled receptor US28.

Keywords: Network analysis, module, set-based annotation, visualization, Cytoscape

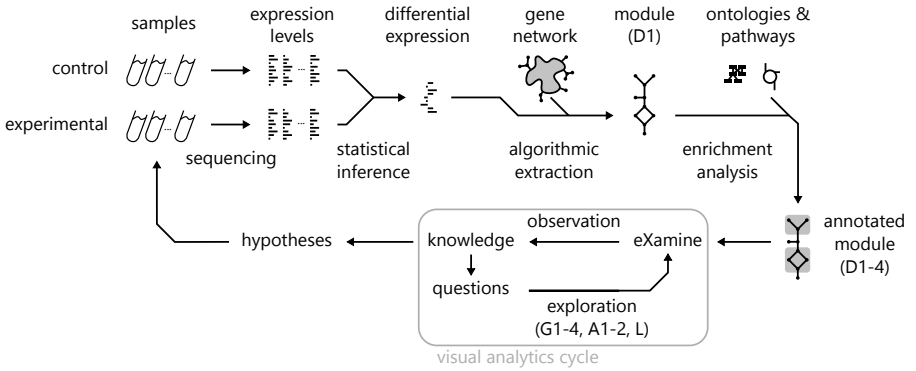


Figure 6.1: **Data and analysis pipeline.** First, control and experimental samples are analyzed to estimate expression levels. Subsequently, gene expression differences (between experiment and control) and their significance are determined. These differences are then mapped to an interaction network, from which a module is extracted with overall significantly-differential gene expression. This module is annotated with overrepresented cell mechanisms from ontology and pathway databases. Finally, the enriched module undergoes iterative visual analysis via eXamine.

6.1 Background

High-throughput "omics" data provide snapshots of cellular states in a specific condition. Computational approaches can be used to relate these low-level measurements with high-level changes in phenotype. Traditionally, these approaches were *gene-centric* and typically resulted in ranked lists of differentially expressed genes [5, 89, 201]. Later, gene-centric approaches were complemented by *pathway-* [194, 207] and *network-based* methods [63, 108] to provide inter-gene context for mechanistic insights. Pathway-based approaches identify overrepresented pathways from databases such as the Kyoto Encyclopedia of Genes and Genomes (KEGG) [118]. Network-based approaches yield small, *de novo* subnetwork modules that may span several known pathways, and reveal their crosstalk [150].

Extracted network modules are analyzed in the context of established gene annotations to hypothesize about the module's role in high-level cell conditions (see Fig. 6.1). Genes are often related to very many terms (too many for human comprehension), most of which are likely irrelevant to the analysis context. Therefore, overrepresentation analysis is performed to rank information items by their significance. These items originate from ontologies such as the Gene Ontology (GO) [15], which identifies cellular functions, processes and components that nodes relate to, or from KEGG [118], which relates nodes to pathways. This results in an *annotated module*, which retains the original network structure and includes enrichment information as a *set system*.

Existing tools focus on visualizing large networks, and have only limited or separate set system support or no support at all. Our proposed visual analysis approach displays sets as contours on top of a node-link layout (see Fig. 6.2). It treats mod-

ule edges and annotation sets in a unified way, and contributes the following to the analysis of annotated modules:

- Identification of elementary module analysis tasks and their composition into a visual analysis process;
- Extension of the self-organizing maps (SOM) algorithm to lay out module interactions and annotations in a unified approach;
- Implementation in the form of the Cytoscape app eXamine;
- Demonstration of eXamine via a guided study of an annotated module that is activated by the virally-encoded G protein-coupled receptor US28;
- Discussion on how eXamine facilitates the analysis process.

6.1.1 Data characteristics

The annotated modules—targeted by the presented method—have the following characteristics.

D1 Small and sparse network topology, in which genes and interactions number in the dozens;

D2 Many annotation sets, outnumbering gene interactions;

D3 Annotation sets vary in cardinality, from a single node to the entire module;

D4 Annotation sets overlap often.

Integrative network analysis methods produce small and sparse subnetwork modules (D1), rather than large lists of differentially expressed genes. Embedding the module in a rich context of annotations on overlapping sets of genes is a typical next step to gain insights in the underlying biology (D2, D3, D4).

6.1.2 Analysis tasks

The focus (or perspective) of analysts alternates between genes (and interactions within a module) and annotation sets. Important analysis tasks are supported for each of these data aspects to enable an analyst to hypothesize about the role of an extracted module in light of experimental conditions.

For genes, analysts want to determine:

G1 Level of differential expression: under- or over-expressed, or insignificant;

G2 Interacting neighbors;

G3 Annotations (set memberships);

G4 Annotations shared with other genes.

Single genes can become the focus of attention during the analysis process within the context of the module. The fact that a gene is part of a module does not imply that its under- or overexpression is significant. However, information (G1) about differential expression enables the elucidation of a gene's presence in the module. For example, it could be the case that a gene is not differentially expressed significantly itself, but that it is still part of a module, because it connects two differentially expressed submodules. An indirect involvement of the gene in a module mechanism is therefore likely. Neighboring genes might also become interesting (G2), as are any mechanisms that it is associated with already (G3), and the mechanisms that it shares with other genes in the module (G4).

For annotation sets, analysts want to determine:

A1 Significance of overrepresentation;

A2 Gene memberships.

If a specific gene is interesting, its annotations might be too (G3 and G4). Annotation sets themselves can have such significance (A1) that they become interesting, which then translates to genes contained in them (A2). Both significance in terms of an associated p-value and subjective significance are of importance to divide attention between annotation sets.

For interactions, analysts want to determine:

L Annotation transitions between interacting genes.

A change between annotations (L) may occur when the focus on a gene shifts to a neighboring gene (G2), which is of importance to an analyst to judge the role and relevance of the neighboring gene in the module.

6.1.3 Related work

Network visualization and tools. Many advanced techniques for the visualization of network topology have been developed [23, 99, 211], but few have been transferred to readily available tools. On the other hand, there are many tools for interpreting and exploring biological networks [86], including the popular open source platforms Cytoscape [186] and PathVisio [204]. However, these currently provide only limited capability to visualize annotated modules. PathVisio is a pathway analysis approach, in which sets are restricted to subsets of static, pre-defined individual pathways, and set membership is conveyed via node colors. Cytoscape's group attributes layout can be used to visualize partitions by showing disjoint parts in separate circles, but it does not support overlapping sets. The Venn and Euler diagram app [101] for Cytoscape does support overlapping sets, but it can handle only four at the same time (see Figs. 6.3(a) and (b)). In this app, network and sets are visualized separately: set membership is conveyed by selecting a set and its corresponding nodes are highlighted in Cytoscape's network view. The RBVI collection of plugins [166] facilitates creation and editing of Cytoscape groups, and provides a group viewer that

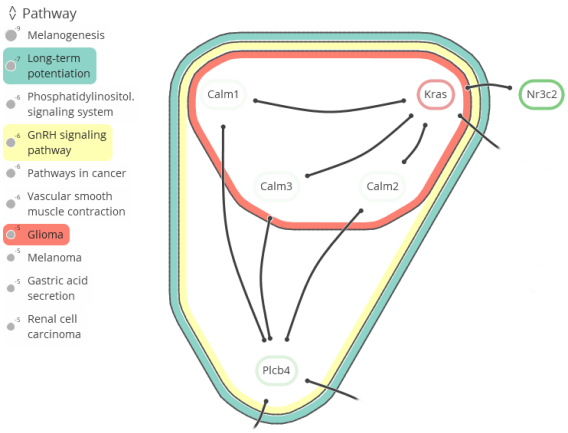
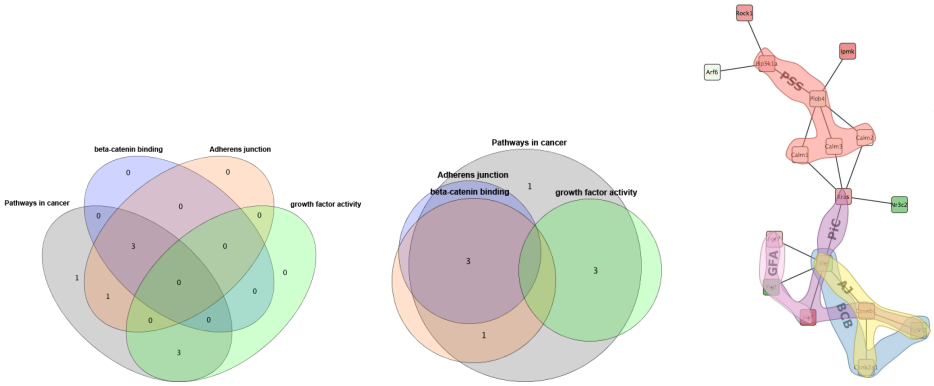


Figure 6.2: **Visualization of an annotated module.** Interacting proteins with a selection of three subsets, corresponding to overrepresented KEGG pathways. The visualization consists of a combination of a node-link diagram and an Euler diagram.



(a) (b) (c)

Figure 6.3: **Comparison.** Annotated module visualization using Cytoscape's Venn and Euler diagram app: (a) Venn diagram and (b) Euler diagram. The number of displayed sets is limited to four and no network structure is shown. (c) Module laid out by one of Cytoscape's built-in force-directed layout algorithms and BubbleSets superimposed on the network (same color scheme as in Fig. 6.9(b)). Note that it is not immediately apparent that the nodes in the β-catenin set (blue) form a subset of Adherens junction (yellow), because the BubbleSet approach applies no explicit nesting of subsets.

relies on aggregation of groups into meta-nodes. These meta-nodes can be visualized as standard nodes, as nodes containing embedded networks, or as charts. This approach, however, does not allow for visualization of overlapping sets.

Set system visualization. In the information visualization field, *Euler diagrams* are used for the intuitive visualization of set systems [27, 168, 182], in which items belonging to the same set are denoted by contours. Variants of these approaches visualize sets over items with predefined positions, e.g., over a given node-link visualization of a network. These methods range from connecting these items by simple lines (LineSets) [8], via colored shapes that are routed along the items (Kelp Diagrams) [62] and contours around the items (BubbleSets, see Fig. 6.3(c)) [51, 185] to hybrid approaches (KelpFusion) [148]. Visualizing an annotated module, however, requires an integrated layout of both its network and set system topologies, which is not possible with these approaches. Euler diagram methods focus on the layout of set relations at the expense of network topology. Likewise, laying out the network before superimposing set relations will emphasize network topology to the detriment of the set system. Some techniques exist that provide such integrated layouts [20, 67, 181, 192], and which include aesthetic concerns and design of visual metaphors [83]. However, these approaches assume constraints on the network and set system topologies, e.g., strict partitions and no overlapping sets, and they are therefore not applicable to our problem.

6.2 Method and implementation

Visualizing an annotated module amounts to visualizing a *hypergraph* consisting of binary edges (interactions) between nodes (genes) and n-ary edges (annotation sets). Analysis tasks G2-G4 and A2 establish the equal importance of associating interactions and annotation sets, which reflect on both the layout as well as the visualization of the hypergraph. Therefore, as opposed to combining multiple existing techniques—e.g., a force simulation to position the nodes according to the binary edges [80], a node overlap removal algorithm to keep nodes identifiable [66], and subsequent construction of a density field to derive contours for annotation sets [51]—our approach relies on a unified algorithm that treats binary and n-ary edges on equal terms. This allows us to compute a balanced layout, and also to choose suitable representations for the binary and n-ary edges. Mathematically, we achieve this by assigning a bit vector $\mathbf{t} = (t_1, t_2, \ldots, t_M)$ to every node $t \in V$ (the module genes) that encodes its membership in binary and n-ary edges $S_1, S_2, \ldots, S_M$. That is, $t_i = 1$ if $t \in S_i$ and $t_i = 0$ if $t \notin S_i$.

To make this representation more concrete, consider the annotated module shown in Fig. 6.2. The nodes are represented as the set $V = \{$*Calm1*, *Calm2*, *Calm3*, *Kras*, *Nr3c2*, *Plcb4*$\}$. There are seven sets representing the edges and three sets representing pathway memberships. The edge sets are $S_1 = \{v_1, v_4\}$, $S_2 = \{v_1, v_6\}$, $S_3 = \{v_2, v_4\}$, $S_4 = \{v_2, v_6\}$, $S_5 = \{v_3, v_4\}$, $S_6 = \{v_3, v_6\}$, and $S_7 = \{v_4, v_5\}$. Note that nodes v_4 (*Kras*) and v_6 (*Plcb4*) have some additional outgoing edges, but their targets are not visible in the image. Therefore, we ignore these edges in this example. The pathway memberships are the *Glioma* set $S_8 = \{v_1, v_2, v_3, v_4\}$, the *Long-term potentiation* set $S_9 = \{v_1, v_2, v_3, v_4, v_6\}$, and the *GnRH signaling pathway* set $S_{10} = \{v_1, v_2, v_3, v_4, v_6\}$.

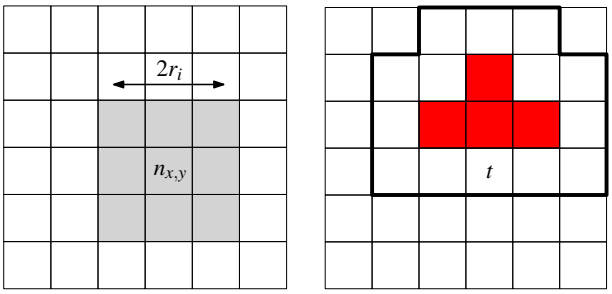


Figure 6.4: **Training neuron $n_{x,y}$.** (a) The neighborhood within range r_i is trained (colored gray). (b) Certain tiles are already reserved (colored red) in the *RSOM* algorithm, item t therefore trickles outwards to the best matching free spots (outlined).

Now, for example, node v_5 gets assigned the bit vector $\mathbf{t}_{v_5} = (0,0,0,0,0,0,1,0,0,0)$ and node v_6 the bit vector $\mathbf{t}_{v_6} = (0,1,0,1,0,1,0,0,1,1)$.

This high-dimensional representation is then used to lay out the nodes without overlap, the binary edges as curves, and the n-ary edges as contours.

6.2.1 Extension to Self Organizing Maps

Self Organizing Maps (*SOMs*), introduced by Kohonen [129], are artificial neural networks that are used to map high-dimensional data items to discretized low dimension. SOMs are used in a visualization setting to cluster similar items together in a 2D embedding, which results in a landscape of items based on their features [142, 208]. Typical SOMs consist of a square grid of size $N \times N$ with a neuron $n_{x,y} \in [0..1]^M$ at every grid cell. A neuron $n_{x,y}$ is a bit vector of size M whose dimension matches the data items' dimensions. In our case, the data items $\mathbb{T}$ correspond to the set of nodes V in the annotated module. The training algorithm applies unsupervised reinforcement learning in an iterative fashion: at every iteration $i \in \{1,\dots,I\}$ all data items $t \in \mathbb{T}$ are considered and the neuron that matches t most closely is determined using a distance function such as the Euclidean or Manhattan norm. This neuron and its neighboring neurons within radius r_i are updated to match t even more closely by setting their respective vectors q to $q + \alpha_i(t-q)$—see Fig. 6.4(a). In early iterations i, the trained neighborhoods are large with r_i close to the grid size N and the training strength α_i close to 1. The parameters r_i and α_i decrease monotonically with increasing i. As such, items that differ strongly will distribute across the map to establish their own regions in the grid at early stages. Items with smaller differences are separated along the grid at a more local level as the training iterations progress.

Reservation-based training. Similar items may end up at the same grid position in a standard SOM. This issue is usually solved by showing aggregate depictions of items, but we need to have separate depictions without overlap to support tasks G1-

G4. Therefore, each item has to map to a unique grid position. We achieve this by altering the training algorithm:

Algorithm *RSOM* $(\mathbb{T})$
1. **for** $i \leftarrow 1$ **to** I
2. **do** Initialize copy $\mathbb{U}$ of $\mathbb{T}$ and clear neuron reservations.
3. **while** $\mathbb{U}$ contains items
4. **do** Draw and remove item t from $\mathbb{U}$.
5. Find unreserved neuron $n_{x,y}$ with smallest distance $d(t, n_{x,y})$.
6. Reserve $n_{x,y}$ for t.
7. **for** any neuron q within range r_i from (x, y)
8. **do** $q \leftarrow q + \alpha_i(t - q)$

The algorithm assigns items to a unique neuron after every training iteration, because, once a neuron is reserved by an item, subsequent items will ignore it. This causes a flooding effect where similar items end up in the same area of the grid and trickle outwards as the area becomes more crowded—see Fig. 6.4(b).

Configuration. The metric distance form of cosine similarity is used as the distance function d, i.e. $d(q,p) = \cos^{-1}((q \cdot p)(|q||p|))\pi^{-1}$. This measure outperforms the Euclidean and Manhattan norms in high-dimensional spaces. The SOM is trained with a learning strength and neighborhood range that decrease linearly with increasing iteration i. A standard choice is $\alpha_i = c \cdot (1 - i/I)$ and $r_i = \lfloor (1 - i/I) \cdot N \rfloor$, where $c \in (0..1)$ is a small constant that determines the initial training strength. We use $N = 2|\mathbb{T}|$ for the number of neurons and iterations, balancing node placement freedom versus required display space, and $I = 10^6/|\mathbb{T}|$ for a gradual and accurate training, respectively.

Layout preservation. A new layout has to be computed whenever the user selects or deselects a set. The new layout should change little in comparison to the old layout to preserve the user's mental map. This is achieved by a simple addition to the SOM algorithm, where a new SOM is initialized with the previous configuration of the neurons, i.e., an item that was positioned at $n_{x,y}$ in the old SOM is placed at $n_{x,y}$ in the new SOM and its neighborhood is trained according to the new bit vector of the item. The new SOM retains much of the initial configuration by starting the training factor α_i at $c = 0.01$. Naturally, this imposes a trade-off between layout quality and conservation. The layout will sometimes change strongly to accommodate the addition of a set that contains many items. In contrast, the layout can be retained if only a small set that does not alter much of the topology is added. This approach does not consider a history of topological changes, as is done in online graph drawing [79] to capture temporal dynamics, but is sufficient to maintain a stable and interactive environment.

Set dominance. The user is enabled to make a certain set more dominant in the layout by having the training algorithm place the items of that set closer to each other than the items of other sets. This relies on weighting the components of the

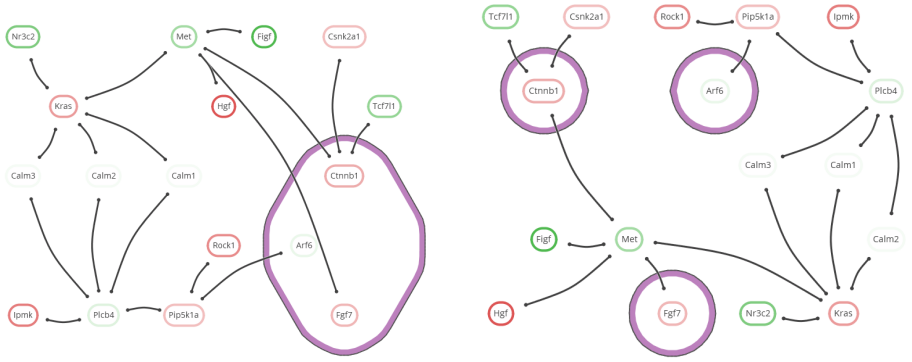


Figure 6.5: **Changing the dominance of a set.** (a) Highly dominant set, drawing proteins of the set together. (b) Non-dominant set, where the network topology fully defines the layout.

item bit vectors: every S_i is given a weight w_i with $w_i = 1$ initially. The bit vectors are augmented to incorporate these weights: $t_i = w_i$ if $t \in S_i$ and $t_i = 0$ if $t \notin S_i$. The bit vector component of S_i will therefore play a more prominent role in distance metric d when the user increases w_i—see Fig. 6.5.

Assigning greater weight to a set improves the quality of its layout by coalescing its elements, which aids tasks G4 and A2. However, it also degrades the layout quality of other sets and links when their topology conflicts with the prioritized set. This stems from the difficulty of projecting elements from a high-dimensional space down to a two-dimensional space, which sometimes results in a sub-optimal layout per set. Interactive manipulation provides a way to assign different priorities to sets, and improve their layouts.

Contours. The SOM's neuron grid is used to define the contours representing the active set system. Let S_i be an active set. The corresponding i-th components of the neurons define a scalar field that forms a fuzzy membership landscape for S_i. This field is similar to the density field used in Bubble Sets [51]. Now, the inclusion of the grid tile of neuron n in the contour body is determined by imposing a threshold, of for example $\frac{1}{2}$, on the i-th component (see Fig. 6.6(a)). The contour can then be tightened to reduce sharp corners by including parts of tiles that are free of items, as illustrated in Fig. 6.6(b).

After establishing the layout of the contours, we apply geometric post processing steps [62] to improve aesthetics, where all sets are legible (tasks G3 and A2) and contours form clear boundaries underneath interactions (task L). Sharp corners of the initial contours are rounded by a dilation of r, erosion of $2r$, and subsequent dilation of r (see Fig. 6.6). Here *dilate* and *erode* are equivalent to *Minkowski sum* and *Minkowski subtraction* operators with a circle of radius r [55], respectively. In addition, the contours are nested by applying different levels of erosion, enforcing a

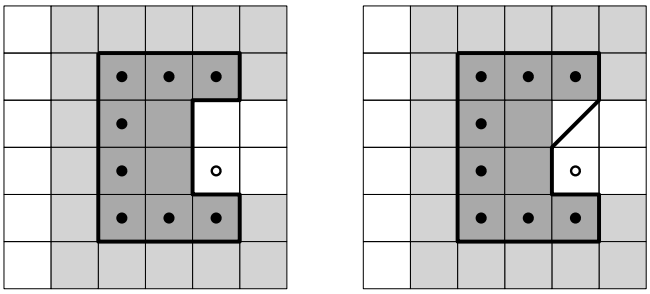


Figure 6.6: **Derivation of contours for set S_i.** The darkness of a tile represents the value of the neurons' i-th component, the thick black line is the contour, dots represent items that are in S_i, and white dots are items that are not in S_i. (a) Contour that results from the union of tiles with a value above a certain threshold. (b) Refined contour with shortcuts across free tiles.

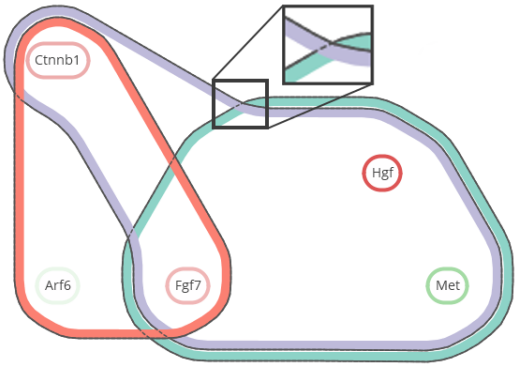


Figure 6.7: **Geometric refinement of set contours after initial layout.** Corners are smoothened by dilation and erosion operations, and contours are given a thick and colored internal ribbon. Unique erosion levels create distance between contour outlines, and contour overlap is emphasized by dashed lines.

certain distance between them. The thick colored ribbons in Fig. 6.7 are obtained by taking the body b of a contour, eroding it to get a smaller body b_e, and taking the symmetric difference $b - b_e$ of b and b_e to effectively cut b_e out of b. Here, the extent of the erosions and dilations (radius r) is bounded by a fraction of the grid's tile size. This guarantees that items are contained by a contour of S_i if, and only if, these items are contained by S_i.

Set contours are drawn in descending nesting order, which is defined by their different erosion levels; the largest contour is drawn first and the smallest contour last. The contour ribbons are assigned unique colors per set and are drawn fully opaque to prevent any confusion caused by blended colors. Occlusion is mitigated by

limiting the width of the ribbons. Finally, the contours are drawn a second time as dashed lines such that occluded contour sections can be inferred—see Fig. 6.7.

6.2.2 Implementation

We have implemented the technique in a Cytoscape app, and have emphasized simplicity of interaction and visual presentation in the design. The available sets are sorted by significance and listed in the *set overview* on the left, where the significance of a set is visualized as a circle, scaled logarithmically and accompanied by its scientific exponent as text (task A1). The user may select sets for inclusion in the annotated *network visualization* to the right—see Fig. 6.9(c). All described functionalities can be used at interactive speeds for networks up to dozens of nodes, edges, and active sets, including laying out the network with the *RSOM* training algorithm. Geometric operations on the contours, such as dilations and erosions, are performed via Java Topology Suite [210].

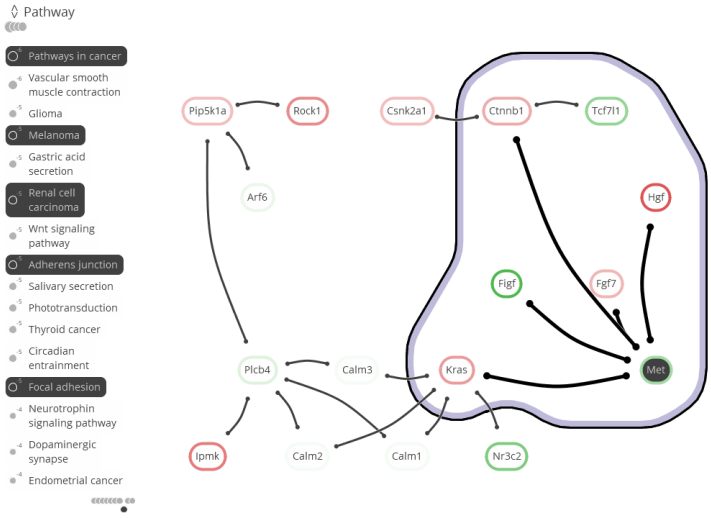
Interaction. Interactions consist of simple mouse actions (see the video in the Supplemental Material). The inclusion of a set in the network visualization is toggled via the set's label in the set overview or its contour in the network visualization (task A2). Additional information about a set or node may be obtained via a hyperlink to a web page provided in the input data, enabling quick access to external information sources such as the KEGG website. This approach keeps the tool flexible, i.e., the tool itself does not have to be altered every time a new kind of set or node from a different database is loaded.

The links of a node are emphasized when it is hovered over (see Fig. 6.8(a)) such that its direct neighborhood can be discerned from its surroundings (task G2). Moreover, sets that contain the hovered node are highlighted as well. Likewise, links can be hovered to highlight their nodes and common sets. Vice versa, the contours of a set are emphasized and its comprising nodes are highlighted when it is hovered over (see Fig. 6.8(b)). This provides immediate feedback to the user about node-set relations (tasks G3 and A2) without having to select a set and consequently changing the layout of the network visualization.

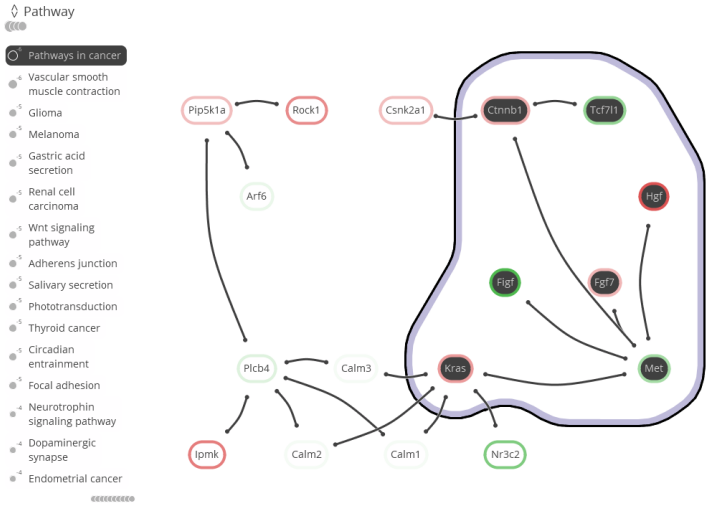
The lists of annotations sets can be expanded and collapsed by clicking on their headers, and scrolled downward to sets of lower significance by turning the mouse wheel. The set circles that convey significance remain visible at all times, grouping at the list top and bottom, to guarantee the depiction of all set memberships when a node is hovered.

The user can adjust the dominance of a set by spinning the mouse wheel while hovering over either the set's label in the set overview or contour in the network visualization. This enables the user to give a set a central role in the layout (see Fig. 6.5(a)) or to remove any of its influence (see Fig. 6.5(b)).

All changes to the visualization caused by interaction are animated. Colors and positions of items are altered gradually. Link layout changes are animated by interpolating their control points, while contour layouts are handled by fading out the old contour and fading in the new contour. The use of layout preservation, as described previously, in combination with animations helps to preserve the user's mental map.



(a)



(b)

Figure 6.8: **Item highlighting.** (a) Hovered protein (Met) with emphasized interaction links to its neighbors on the right and emphasized sets (KEGG pathways) that contain this protein on the left. Sets outside of the list scope are grouped as markers at the top and bottom, where one set in the bottom group is emphasized. (b) Hovered set (Pathways in cancer) with emphasized member proteins, interactions, and contour.

Color. Unique, distinguishable colors are derived from Color Brewer palettes [97], and assigned to annotation sets in a cyclic manner to avoid assigning the same color consecutively. In addition, large differences in contrast are avoided. For example, text and set outlines are colored dark gray instead of black to reduce their visual dominance. Black is only used when items are hovered over or highlighted such that they attract attention, as shown in Fig. 6.8. Moreover, labels of selected sets (in the set overview) are emphasized with a more intense black color to ensure that they are readable in a colored surrounding. Node labels have a white background to make sure that their text is legible when drawn on top of a set ribbon with a dark color. Likewise, links have halos that make them easier to distinguish and their intersections more pronounced.

Cytoscape integration. eXamine is tightly integrated into Cytoscape. Cytoscape's group functionality is used to represent sets and we rely on the table import functionality for importing both the set and node annotations. The user is also able to group sets into different categories. The Cytoscape node fill color map attribute is used to color the nodes in eXamine according to gene expression score (task G1). The user therefore has the freedom to define the desired color map via Cytoscape. The user can invoke eXamine on the currently selected nodes via the eXamine control panel. There the user can select which categories to show as well as the number of sets per category. In addition, the user can specify that the Cytoscape selection should be updated to match the union or intersection of the selected sets in eXamine (see Fig. 6.9). This enables the use of eXamine with any kind of module extraction algorithm and/or filter method in Cytoscape, which includes manual node selection.

6.3 Case study of US28-mediated signaling

We demonstrate how a domain expert can use eXamine by working out a case study in which a data set is re-analyzed (this work was done by the co-authors with biological expertise). While this data set has been studied extensively, it was possible to derive a new hypothesis via eXamine.

The *Human Cytomegalovirus (HCMV)* is a highly-contagious herpes virus [82]. Infection with HCMV in healthy humans usually does not result in symptoms. However, in humans with a compromised immune system the virus is correlated with diseases such as hepatitis and retinitis [188]. In addition, HCMV gene products have been detected in various tumors even though HCMV is not considered to be an oncogenic virus. Experts therefore hypothesize that the virus may act as a stimulating factor during onset and development of cancer without being a root cause [45, 48, 95].

HCMV is responsible for the production of several viral G protein-coupled receptors (vGPCRs). Of these vGPCRs, US28 is the most studied and is characterized as chemokine sink [164]. Chemokines are signaling proteins that induce cell migration. Moreover, US28 hijacks the host cell's signaling pathways, stimulates proliferative signaling pathways [41, 136, 145, 146, 184]. Previous studies focused on transcriptome analysis to evaluate pathways that are affected by US28. Differentially expressed genes involved in HCMV-induced disease symptoms were identified

and related to known pathways [146, 184]. However, this analysis did not include network-based module extraction and enrichment.

To identify additional deregulated signaling due to US28, we analyzed the same data overlaid on the KEGG mouse network [118]. The network consisted of 3863 nodes and 29293 edges. Gene p-values, reflecting whether genes are significantly differentially expressed, were derived using RMA [85] and LIMMA [187]. Heinz [63], a tool for identifying differentially expressed modules, was then applied using a false discovery rate of 0.0007. This resulted in a module of 17 proteins. Finally, enrichment analysis using TopGO [4] was performed to annotate this module with enriched GO-terms and KEGG pathways (see Fig. 6.9).

These data processing steps correspond to the initial steps in Fig. 6.1. The subsequent analysis of the annotated module aims at obtaining new insights about US28-mediated signaling. The analysis follows the visual analytics cycle consisting of *observation*, *knowledge*, *questions* and *exploration*, finalized by a hypothesis.

C1 Two familiar pathways

Observation. The KEGG pathway annotation sets show significant presence of *Pathways in cancer* and *Phosphatidylinositol signaling* (p-values of $5.6 \cdot 10^{-6}$ and $1.0 \cdot 10^{-6}$, respectively).

Knowledge. An oncomodulatory role has been proposed for US28 [45, 48, 95], which coincides with the presence of *Pathways in cancer* and makes the genes annotated by this term of interest. *Phosphatidylinositol signaling* corresponds to previous work linking US28 to Phosphatidylinositol-mediated calcium responses [41, 149].

Question. Which parts of the module are involved in *Pathways in cancer* and *Phosphatidylinositol signaling*?

Interaction. Tag the *Pathways in cancer* and *Phosphatidylinositol signaling* annotation sets (see Fig. 6.9(a)).

C2 Choosing sides

Observation. Clear division of the module is apparent after tagging the two familiar pathways. Genes *Arf6*, *Csnk2a1*, *Csnk2a1*, *Ipmk*, *Nr3c2* and *Rock1* are not part of the pathways but have direct, unambiguous interactions with either of the pathways.

Knowledge. Because of the known involvement of US28 in *Phosphatidylinositol signaling*, we do not focus on the genes of this pathway (*Calm1..3*, *Plcb4*, *Pip5k1a*), nor on the directly interacting genes (*Arf6*, *Ipmk*, *Rock1*). Instead, the *Pathways in cancer* genes *Kras*, *Met*, *Figf*, *Hgf*, *Fgf7*, *Ctnnb1* and *Tcf7l1*, and directly interacting genes *Nr3c2* and *Csnk2a1* may lead to new insights in US28-mediated signaling and ultimately the oncomodulatory role of HCMV.

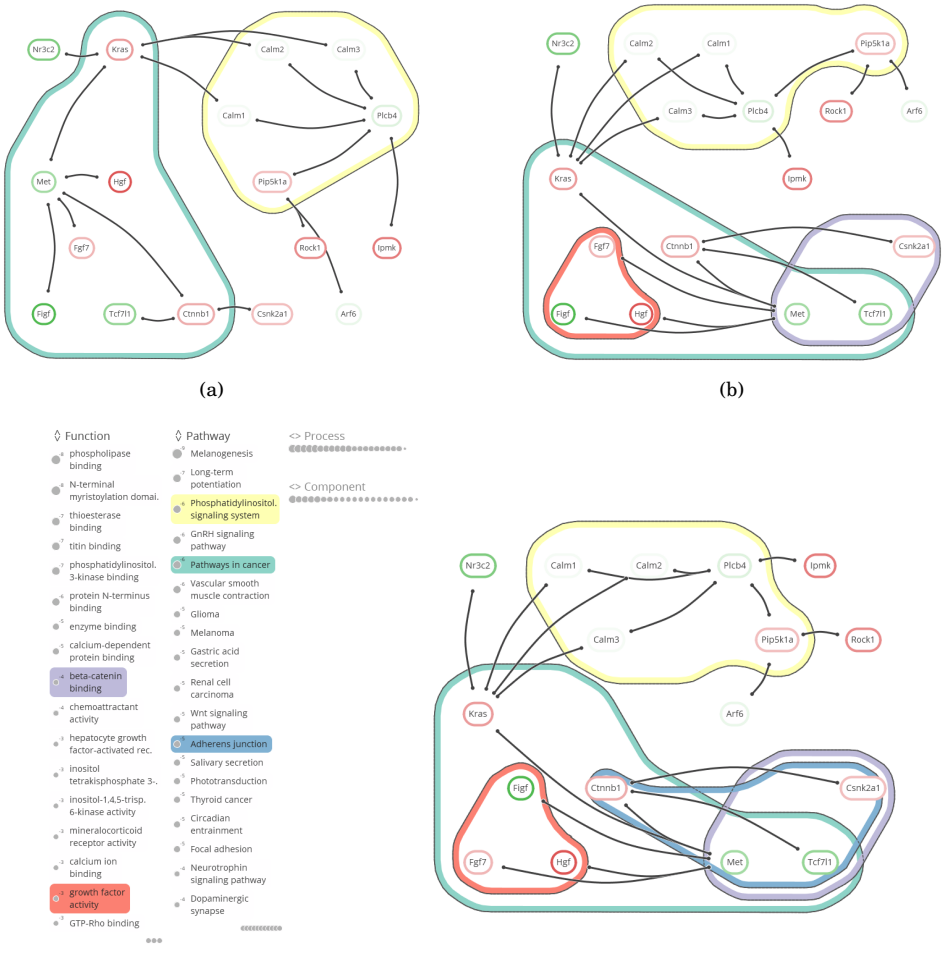


Figure 6.9: **Case study snapshots.** Gene differential expression is shown as a colored box drawn around the node label (green for under-expression and violet for over-expression). (a) The annotated module after tagging of the two familiar pathways *Pathways in cancer* and *Phosphatidylinositol signaling system* in C1. (b) The annotated module after tagging functions *Beta-catenin binding* and *Growth factor activity* in C3 and C4. (c) The fully annotated module, including annotation set overview, from which the hypothesis of C5 is derived.

Question. Do any of the aforementioned genes in or adjacent to *Pathways in cancer* lead to new insights in US28-mediated signaling?

Interaction. Hover over the genes in and close to *Pathways in cancer* to determine mechanisms of interest.

C3 A twist of β-catenin

Observation. The genes in *Pathways in cancer* can be divided roughly into two subsets: those that are annotated by *growth-factor activity* and those annotated by *β-catenin binding* (see Fig. 6.9(b)). *Csnk2a1*, *Tcf7l1* and *Met* are part of the latter annotation set, where *Tcf7l1* and *Csnk2a1* are down- and up-regulated, respectively. Expression of the neighboring *Ctnnb1* (β-catenin) is up-regulated.

Knowledge. β-catenin signaling results in elevated protein levels of the TCF/LEF transcription factor family that contains the protein encoded by *Tcf7l1*. Although *Tcf7l1* is down-regulated, a recent study shows that this is not reflected at the protein level and that US28 induces β-catenin signaling [136]. In the same study, involvement of WNT/Frizzled via the canonical signaling pathway was ruled out and a hypothesis stating that US28-mediated signaling of β-catenin proceeds via ROCK1, which is also present in the module, was postulated.

Question. Are there alternative mechanisms explaining the activation of β-catenin?

Interaction. Tag the *Growth factor activity* annotation set (see Fig. 6.9(b)).

C4 Growing knowledge

Observation. *Fgf*, *Hgf* and *Figf* are annotated with *Growth factor activity* and connected to β-catenin via *Met*.

Knowledge. MET is a receptor tyrosine kinase, whose only ligand is HGF. Therefore we can rule out the links from *Met* to *Fgf* and to *Figf*. In fact, these links are artifacts of how the mouse network was constructed from KEGG pathways. These artifacts often link whole groups of genes such as, in this case, growth factors to receptor tyrosine kinases.

Question. Does the *Hgf–Met* axis relate to β-catenin activation?

Interaction. Hover over *Met* and *Ctnnb1* (β-catenin).

C5 New insights

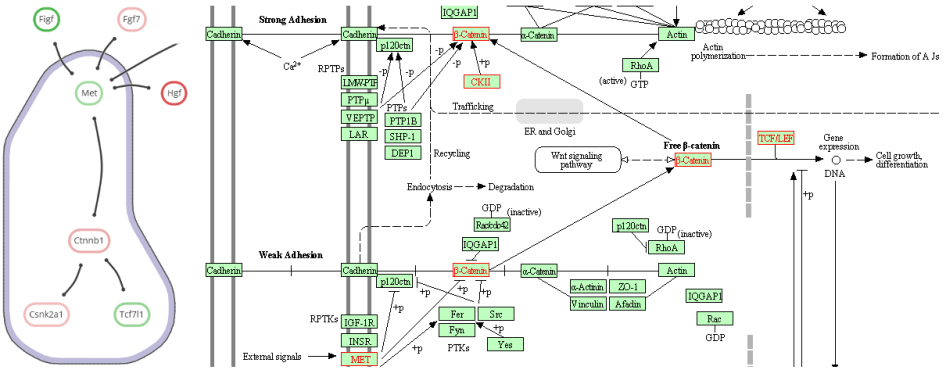


Figure 6.10: **Connection between Met and β-catenin.** Proteins that are associated to the selected *Adherens junction* at the left and corresponding KEGG pathway information at the right, where reactions catalyzed by module proteins are marked in red. Activation of MET by its ligand HGF results in the phosphorylation of β-catenin. This in turn results in its release from cadherin-complexes on the cell membrane into the cytoplasm.

Observation. *Met* and β-catenin are both part of the *Adherens junction* pathway, as are *Tcf7l1* and *Csnk2a1* (see Fig. 6.9(c)).

Knowledge. Adherens junctions bind two cells together, keeping multiple cells in place. Alternative mechanisms have been described that explain β-catenin activation via the release of β-catenin from cell to cell adherens junctions (e.g. [228]). US28 promotes cell migration [190, 191], which causes the loss of cell to cell contacts with subsequent release of β-catenin into the cytoplasm. This may explain increased levels of β-catenin as found previously [136].

By requesting additional information for *Adherens junction* via eXamine, showing an external website by KEGG, we find an indirect connection between *Met* and β-catenin in the pathway (see Fig. 6.10). Activation of MET via HGF mediates the release of β-catenin from adherens junctions, resulting in increased TCF/LEF levels [100, 162].

Hypothesis. Combining this with the growth factor observations of C4 leads to the following hypothesis.

- US28-mediated up-regulation of *Hgf* results in elevated levels of the corresponding HGF protein;
- The subsequent activation of MET results in the release of β-catenin into the cytoplasm;
- Subsequent translocation into the nucleus leads to enhanced TCF/LEF activation.

Synopsis We are currently validating the hypothesis experimentally. Preliminary results indicate that the up-regulation of *Hgf* is indeed reflected at the protein level. Should this hypothesis turn out to be true, we would obtain crucial insights into one of the mechanisms by which the HCMV-encoded chemokine receptor US28 rewires cellular signaling. Ultimately, we would like to understand how this virus achieves its oncomodulatory role and how this can be disrupted.

6.4 Discussion

The analysis tasks described in the background section guided the design decisions that we have taken in the implementation of eXamine. These decisions are motivated via the analysis cycles of the US28 case study.

Overview. The benefit of a spacious annotation set overview follows from the first cycle (C1), in which the categorized, ranked, and legible annotation lists enable the fast recognition of two familiar and significantly represented pathways (task A1). Subsequent tagging of the two pathways reveals their module genes (task A2) and concisely drawn contours emphasize the division of the module into two parts and some additional genes that are not part of the pathways.

An annotation table, separate of the network, would not have made this division as apparent. The main reason is that annotation set transitions along gene interactions are not explicit in such a representation. In contrast, such cross-contour interaction links are clearly visible in eXamine (e.g. the transition from *Kras* in *Pathways in cancer* to *Nr3c2* outside of *Pathways in cancer*).

Annotated genes. The need to focus on specific genes and their properties appears in the second analysis cycle (C2), in which genes of *Pathways in cancer* are inspected for annotations of interest (task G3). Highlighting annotations by hovering over genes enables fast identification of relevant annotations in the stable overview that oriented the analyst in C1. Vice versa, hovering an annotation of interest (β-*Catenin binding*) confirms that it is shared by *Csnk2a1*, *Tcf7l1*, and *Met* (task G4). The same observations could have been made from an annotation table. However, the topological characteristics of these three genes would have been harder to discern, i.e., their direct interaction with *Ctnnb1* (task G2). This also applies to other set visualizations without depiction of network topology, such as Venn or Euler diagrams, as shown in Fig. 6.3(a) and (b). To make the topology of the gene interactions more explicit, a node-link visualization could be used. For example, Fig. 6.3(c) shows the module laid out by one of the built-in force-directed layout algorithms of Cytoscape with all five annotation sets superimposed as BubbleSets. However, the structure of the annotation sets is hard to discern, and it is not immediately clear that nodes belonging to the β-catenin binding set (blue shape) form a proper subset of the Adherens junction set (yellow shape).

Integration. The third cycle (C3) shows the importance of gene expression values (task G1), which is not limited to the interpretation of genes in isolation but along

multiple genes, their interactions, and shared annotation sets. The importance of integrated support for all analysis tasks follows from the remaining cycles (C4-C5), where multiple deductions are made in succession via multiple tasks. Here, tagging relevant pathways enables the analyst to build up a context for making deductions.

Limitations. eXamine is designed to accurately convey small and annotated modules, consisting of up to about thirty proteins and categories of up to about twenty annotations (note that these limits are not hard). The case study shows that common analysis tasks for these modules are covered. Scalability is a concern as our approach focuses on small modules to enable accurate depiction of sets contours; it is not possible to construct a comprehensive layout if the module consists of hundreds of proteins or if there are dozens of annotation sets to visualize at the same time. Both aspects make visual analysis ineffective. This is a natural limitation of any visualization approach based on node-link diagrams and set contours, however.

Our technique relies on a focus and context approach, in which the network and set system has been pruned down to the most relevant components first. Communicating small-scale information is given priority to support hypothesis generation at the level of individual proteins and their interactions, as follows from the targeted analysis tasks. Nonetheless, the tool is capable of visualizing modules of up to a hundred proteins, albeit with less legibility of interactions and annotations.

The integration of eXamine into Cytoscape mitigates many scalability issues. Cytoscape, for example, provides a global view of the network, in which the user can zoom in on smaller subnetworks for more in-depth analysis by eXamine. In addition, the integration into Cytoscape provides access to further analysis algorithms.

The extended SOM algorithm embeds an annotated module to reflect its topology, i.e., the distances between its proteins based on common interactions and annotations. This does not guarantee optimal aesthetics however, and unnecessary link and contour intersections can sometimes occur. The analysis tasks targeted by eXamine are not much hampered by such intersections since all interactions, annotations, and their interplay remain pronounced. However, to communicate analysis results, aesthetics might need further improvement. This could be done by weighing aesthetic criteria such as the number of intersections and shape complexity against each other, and formulating this as a combinatorial optimization problem. The associated algorithms [23] are often complex, and it is not so easy to integrate them into an interactive system.

Application to other domains. eXamine is not limited to the analysis of enriched protein modules nor to data from the biological domain. It can be applied to any small network module that is accompanied by a set system, such as a social circle that consists of people, their relationships, and common interests.

6.5 Conclusions

We have proposed a visualization approach that enables the analysis of small and annotated network modules, and have implemented this in the Cytoscape app eXamine.

Our approach displays sets as contours on top of a node-link layout. We have introduced an extension to the self-organizing maps algorithm to lay out module edges and annotation sets in a unified way. The added value of our approach has been demonstrated in a case study of a US28-mediated signaling module, in which a novel hypothesis about the way US28 induces β-catenin signaling has been derived.

Availability and requirements

Project name: eXamine
Project homepage: http://apps.cytoscape.org/apps/examine
Operating system(s): all
Programming language: Java
Other requirements: Cytoscape 3.x
License: GPL2
Any restrictions to use by non-academics: None

Acknowledgments. Kasper Dinkla is supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 612.001.004.

Competing interests. The authors declare that they have no competing interests.

Author's contributions. KD, MEK, MAW and GWK conceived the visual analysis technique. KD and MEK implemented eXamine. CIB, MS and MJS applied it to the US28 case study after instructions by MEK and GWK. KD, MEK, MAW and GWK drafted the manuscript. All authors read and approved the final manuscript.

Chapter 7

Cross-species modules

Published as:

M. El-Kebir[†], H. Soueidan[†], T. Hume[†], D. Beisser, M. Dittrich, T. Müller, G. Blin, J. Heringa, M. Nikolski, L. F. A. Wessels, G. W. Klau. xHeinz: An algorithm for mining cross-species network modules under a flexible conservation model. *Bioinformatics*, 2015.

[†]joint first authorship

Abstract

Motivation: Integrative network analysis methods provide robust interpretations of differential high-throughput molecular profile measurements. They are often used in a biomedical context—to generate novel hypotheses about the underlying cellular processes or to derive biomarkers for classification and subtyping. The underlying molecular profiles are frequently measured and validated on animal or cellular models. Therefore the results are not immediately transferable to human. In particular, this is also the case in a study of the recently discovered interleukin-17 producing helper T cells (Th17), which are fundamental for anti-microbial immunity but also known to contribute to autoimmune diseases.
Results: We propose a mathematical model for finding active subnetwork modules that are conserved between two species. These are sets of genes, one for each species, which (i) induce a connected subnetwork in a species-specific interaction network, (ii) show overall differential behavior and (iii) contain a large number of orthologous genes. We propose a flexible notion of conservation, which turns out to be crucial for the quality of the resulting modules in terms of biological interpretability. We propose an algorithm that finds provably optimal or near-optimal conserved active modules in our model. We apply our algorithm to understand the mechanisms underlying Th17 T cell differentiation in both mouse and human. As a main biological result, we find that the key regulation of Th17 differentiation is conserved between human and mouse.
Availability: xHeinz, an implementation of our algorithm, as well as all input data and results, are available at http://software.cwi.nl/xheinz and as a Galaxy service at http://services.cbib.u-bordeaux2.fr/galaxy in *CBiB Tools*.

7.1 Introduction

Many computational methods have been proposed for the analysis of molecular profiles under different conditions. Studies employing these methods aim to better understand the molecular changes in the underlying cellular processes or to discover biomarkers as to classify between different conditions. Traditionally, analysis methods have been gene-centric, that is, they consider genes in isolation to establish differential patterns by simple statistical methods based on univariate statistical tests. For example, one of the first studies used gene expression measurements to differentiate between two leukemia classes [89]. With the availability of increasingly reliable biological network data for human and model organisms, gene-centric approaches have been increasingly complemented by integrative network analysis methods [63, 108, 150]. These methods yield *active modules*, that is, sets of genes that are connected in the network and show overall differential behavior. By taking the network topology into account, integrative analysis methods allow for a more robust interpretation of the measurements and result in more meaningful mechanistic insights.

Frequently, for ethical or practical reasons, molecular profiles are measured and validated on animal or cellular models and the results are therefore not immediately transferable to human [154]. In fact, the low phase-II survival rate of 25% of potential drug compounds is largely attributed to the lack of transferability between model systems and human [54]. This is also an issue in the recently discovered interleukin-17 producing helper T cells (Th17). These cells form a separate subset of helper T cells with a differentiation pathway distinct from those of the established Th1 and Th2 cells [157]. Th17 cells are known to contribute to pathogenesis of inflammatory and autoimmune diseases such as asthma, rheumatoid arthritis, psoriasis and multiple sclerosis and play also a role in cancer immunology [215]. Understanding the pathways and regulatory mechanisms that mediate the decision making processes resulting in the formation of Th17 is a critical step in the development of novel therapeutics. Unfortunately, the vast majority of data collected so far originates from studies performed on mice [198] and, most importantly, a comprehensive comparison of the Th17 differentiation process in model organisms and in human is missing. Several studies indicate that the differentiation and phenotype of human and mouse Th17 cells are similar [12]. Both subsets serve similar pro-inflammatory functions and produce the same hallmark cytokines and similar receptors. Furthermore, most of the already identified regulator genes show high sequence conservation. Other studies, however, show stimulus requirements for effective differentiation of human cells that differ from those required for mice [13, 147, 153]. A characterization of the similarities and differences will not only increase our understanding of this fundamental process, but is also essential for sound translational research.

To do so, we suggest finding *conserved active modules* whose comprising genes show overall differential behavior, induce a connected subnetwork and are largely conserved across the species. Well-conserved modules make it possible to perform the experimental work and data analysis on the model organism. At the same time, the results are likely to be transferable to human. In addition, conserved modules carry a stronger signal than individual species modules because they integrate the

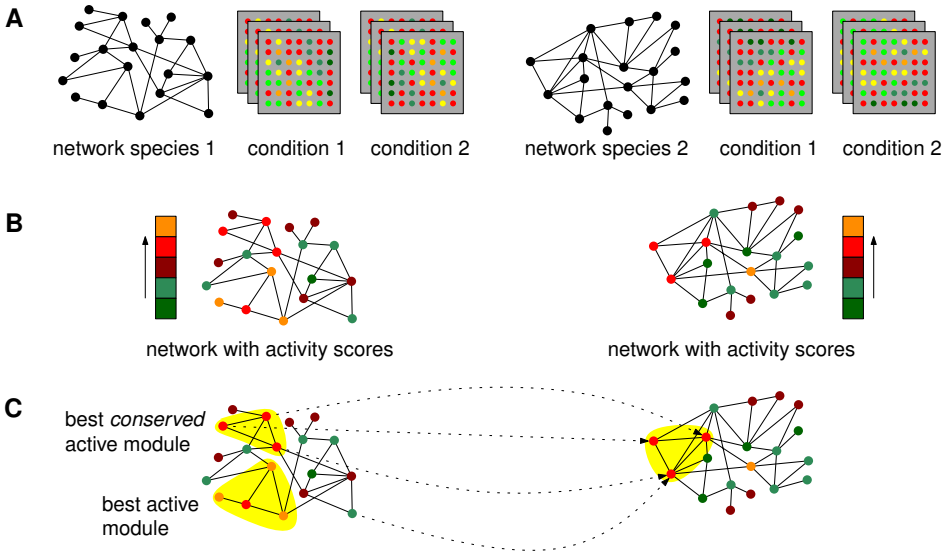


Figure 7.1: **Conserved active modules.** Given two species-specific protein networks and, for each species, two sets of expression profiles of many different samples measured under two different conditions **(A)**, we can annotate the nodes in the networks with activity scores **(B)**, and identify modules that are at the same time highly differentially expressed and well-conserved **(C)**. Cross-species conservation is indicated by dotted lines. Note that the best active module is not necessarily the best conserved active module.

signal of the individual data sources. Finding conserved active modules, however, is a difficult task. Separately computing species-specific active modules generally results in modules that are not conserved, which partially explains why experimental results are so often not transferable. Conversely, the largest conserved modules, as established, for example, with methods for network alignment, are not necessarily active. A computational model for finding conserved active modules requires thus a notion of both, activity and conservation—see Fig. 7.1.

Several authors already identified the benefits of combining and comparing cross-species experiments. At the single gene level, van Noort et al. [206] have demonstrated that conserved co-expression is a strong co-evolutionary signal. More recent studies suggested to identify conserved biological processes. Lu et al. [141] analyzed transcriptomics profiles of human and mouse macrophages and dendritic cells to derive common response genes involved in innate immunity. Kristiansson et al. [131] proposed a method for the analysis of gene expression data that takes the homology structure between the different species into account. Berthier et al. [28] found that murine and human responses to lupus nephritis involves similar gene networks. They first derived species-specific networks of significantly differentially expressed genes and then determined common subnetworks using a graph matching algorithm. Waltman et al. [212] presented a multi-species integrative method to heuristically

identify conserved biclusters. In their setting, a conserved bicluster is a subset of orthologous genes and a subset of conditions that achieve a high score with respect to co-expression, motif co-occurrence and network density. Dede and Oğul [56] introduced a method that finds triclusters consisting of genes that are coexpressed across a subset of samples and a subset of species.

Deshpande et al. [59] suggested the neXus algorithm for finding conserved active subnetworks. The authors use average fold change of genes in a module as a measure for activity. To deal with conservation, they collapse paralogous genes within a cluster of orthologous genes (COG) [195] into single nodes in the respective networks. They find modules using a seed-and-extend greedy heuristic that starts from a pair of orthologous seed nodes and then tries to simultaneously grow the two subnetworks by including pairs of neighboring orthologous genes. This strategy enforces a very stringent conservation policy: only modules whose genes are fully conserved are found. In addition, the locality of the greedy search strategy impairs the ability to find larger conserved modules and extending the search space around the seed genes drastically increases the runtime. In recent work, Zinman et al. [229] introduce ModuleBlast, a method that, similarly to neXus, represents groups of orthologous proteins as single nodes in a combined network and tries to find connected subnetworks that are differentially expressed. The novelty of the method is the classification of the found modules according to the sign of the log fold change expression values. By doing so, the authors are able to assess whether conserved active modules show consistent or inconsistent expression patterns. Like neXus, ModuleBlast requires strict conservation of module genes.

Here, we propose a mathematical model for identifying conserved active modules for two species. It builds upon a model for single-species modules described in [63] and inherits its notions for modularity and activity: A set of genes forms a module if it induces a connected subnetwork. The activity of a module is the sum of the activities of its genes, which are determined using a beta-uniform mixture model on the distribution of p-values that characterize the differential behavior. Instead of enforcing a stringent conservation policy, our model allows to specify the fraction of nodes in the solution that must be conserved. We cast our model as an integer linear programming formulation and present xHeinz, a branch-and-cut algorithm that, given enough time, solves this model to provable optimality, or, if stopped before, reports a solution with a quality guarantee. xHeinz is the first method that flexibly deals with conservation. We apply xHeinz to understand the mechanisms underlying Th17 T cell differentiation in both mouse and human. As a main biological result, we find that the key regulation factors of Th17 differentiation are conserved between human and mouse and demonstrate that all aspects of our model are needed to obtain this insight. We further demonstrate the robustness of our approach by comparing samples of the differentiation process obtained at different time points, in which we search for optimal, conserved active modules under a wide range of conservation ratios. Using a permutation test, we show that our results are statistically significant. Finally, we discuss the main differences between our results and the results obtained by the neXus tool on the same data set.

7.2 Approach

7.2.1 Mathematical model

We consider the conserved active modules problem in the context of two species networks, which we denote by $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Nodes in these networks are labeled by their activity—defined by $w \in \mathbb{R}^{V_1 \cup V_2}$ and conserved node pairs are given by the symmetric relation $R \subseteq V_1 \times V_2$. The aim is to identify two maximal-scoring connected subnetworks, one in each network, such that a given fraction α of module nodes are conserved. The formal problem statement is as follows:

Problem 7.1 (Conserved active modules) *Given $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, $w \in \mathbb{R}^{V_1 \cup V_2}$ and $R \subseteq V_1 \times V_2$, the task is to find a subset of nodes $V^* = V_1^* \cup V_2^*$ with $V_1^* \subseteq V_1$ and $V_2^* \subseteq V_2$ such that the following properties hold.*

- ***Activity:*** *Node activity scores are given by $w \in \mathbb{R}^{V_1 \cup V_2}$, where positive scores correspond to significant differential expression. For details see Section 7.3.2. We require that the sum $\sum_{v \in V^*} w_v$ is maximal.*
- ***Conservation:*** *Conserved node pairs are given by the relation $R \subseteq V_1 \times V_2$. We require that at least a certain fraction α of the nodes in the solution must be conserved, that is, $|U^*| \geq \alpha \cdot |V^*|$ where $U^* := \{u \in V_1^* \mid \exists v \in V_2^* : uv \in R\} \cup \{v \in V_2^* \mid \exists u \in V_1^* : uv \in R\}$.*
- ***Modularity:*** *We require that the induced subgraphs $G_1[V_1^*]$ and $G_2[V_2^*]$ are* connected.

The model allows a trade-off between conservation and activity. If no conservation is enforced ($\alpha = 0$), the solution will correspond to two independent maximum-weight connected subgraphs. Conversely, if complete conservation is required ($\alpha = 1$), the solution can only consist of conserved nodes, which results in lower overall activity. The user controls this trade-off by varying the value of the parameter α from 0 to 1. The activity score monotonically decreases with increasing α—see Fig. 7.2.

Since the maximum-weight connected subgraph problem, which occurs as a subproblem for $\alpha = 0$, is NP-hard [113], the problem of finding conserved active modules is NP-hard as well.

7.2.2 Integer linear programming approach

We formulate the conserved active modules problem as an integer programming (IP) problem in the following way.

$$\max \sum_{v \in V_1 \cup V_2} w_v x_v \tag{7.1}$$

$$\text{s.t. } m_u = \max_{uv \in R} \{x_u x_v\} \qquad u \in V_1 \tag{7.2}$$

$$m_v = \max_{uv \in R} \{x_u x_v\} \qquad v \in V_2 \tag{7.3}$$

$$\sum_{v \in V_1 \cup V_2} m_v \geq \alpha \sum_{v \in V_1 \cup V_2} x_v \tag{7.4}$$

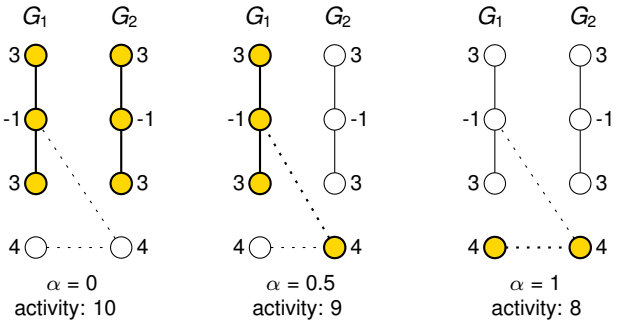


Figure 7.2: **Trade-off between activity and conservation.** Three optimal solutions (indicated in yellow) for varying conservation ratios α in a toy example instance. Node activities are given next to the nodes, conserved node pairs are linked by dotted lines. The activity of a conserved module is the sum of the activities of its comprising nodes. The parameter α denotes the minimum fraction of nodes in a solution that must be conserved, i.e. connected by a dotted line.

$$G_1[\mathbf{x}] \text{ and } G_2[\mathbf{x}] \text{ are connected} \tag{7.5}$$

$$x_v, m_v \in \{0,1\} \qquad v \in V_1 \cup V_2 \tag{7.6}$$

Variables $\mathbf{x} \in \{0,1\}^{V_1 \cup V_2}$ encode the presence of nodes in the solution, i.e., for all $v \in V_1 \cup V_2$ we want $x_v = 1$ if $v \in V^*$ and $x_v = 0$ otherwise. The objective function (7.1) uses these variables to express the activity of the solution, which we aim to maximize. Variables $\mathbf{m} \in \{0,1\}^{V_1 \cup V_2}$ encode the presence of conserved nodes in the solution. Constraints (7.2) encode that a node $u \in V_1$ that is present in the solution ($x_u = 1$) is conserved if there exists a related node $v \in V_2$ ($uv \in R$) that is also present in the solution ($x_v = 1$). Similarly, constraints (7.3) define conserved nodes in V_2 that are present in the solution. The fraction of conserved nodes in the solution is at least α as captured by (7.4). In addition, we satisfy the modularity property by requiring in (7.5) that $G_1[\mathbf{x}]$ and $G_2[\mathbf{x}]$ are connected. In Supplementary Text A.1 we give further details on how to model (7.2), (7.3) and (7.5) as linear inequalities and on the implementation that solves this formulation.

7.3 Material and Methods

7.3.1 Experimental procedure

We summarize here the experimental procedure followed by Tuomela et al. [198] and Yosef et al. [224] to generate transcriptomic profiles. In [198], CD4+ T-cells were isolated from umbilical cord blood of several healthy neonates, arranged in three different pools, then activated with anti-CD3 and anti-CD28. Cells from each pool were then divided in two batches, one to be polarized toward Th17 direction, and one serving as control (Th0). Th17 differentiating cytokines consisted of IL6 (20 ng/mL), IL1B (10 ng/mL) and TGFB (10 ng/mL), along with neutralizing anti-IFNG (1 µg/mL)

and anti-IL4 (1 μg/mL). Three biological replicates of human cells, for both conditions (coming from each pool), were collected between 0.5 – 72 h (0.5 h, 1 h, 2 h, 4 h, 6 h, 12 h, 24 h, 48 h, 72 h time points) and hybridized on Illumina Sentrix HumanHT-12 Expression BeadChip Version 3. The microarray data were analyzed using the beadarray Bioconductor package [65]. In [224], CD4+ T-cells were purified from spleen and lymph nodes from wild type C57BL/6 mice, then activated with anti-CD3 and anti-CD28. For Th17 differentiation, cells were cultured with TGFB (2 ng/mL), IL6 (20 ng/mL), IL23 (20 ng/mL) and IL1B (20 ng/mL) during 0.5 – 72 h (at time points 0.5 h, 1 h, 2 h, 4 h, 6 h, 8 h, 10 h, 12 h, 16 h, 20 h, 24 h, 30 h, 42 h, 48 h, 50 h, 52 h, 60 h, 72 h), and finally hybridized on an Affymetrix HT_MG-430A.

7.3.2 Microarray processing, statistical analysis and node scoring

Preprocessed and quantile normalized data sets were downloaded from GEO under the accession numbers GSE43955 and GSE35103. As downloaded from GEO, both the human and the mouse time-series were already filtered by retaining only the probes with detection p-values < 0.05 in at least one time point and one condition. Following the original studies, we further only retained probes having a standard deviation > 0.15 over all the conditions and time points; as well as being annotated by a single ENSEMBL gene. Finally, a single probe was selected for each gene by taking, for each ENSEMBL gene, the probe having the largest variance accross all samples. In total, 12,307 and 18,497 probes passed the filters for the mouse and human data set, respectively.

Differential expression between Th17 and Th0 conditions were estimated using the limma package [187]. Human samples were indicated as paired according to the experimental design so as to account for the pooled human samples. For mouse samples, calling was performed on all Th0 vs Th17 samples, regardless of the mouse donor. To determine which genes were differentially expressed at a given time point, we used a linear model to estimate the interaction between the treatment and the time effect. The linear models used for the human and mouse studies include one interaction term for each time point and exclude the intercept (In R, the formula reads: $\sim 0 + \text{treat} : \text{time}$). Differential expression at any time point K of interest were determined by the contrasts $\text{Th17.time}_\text{K} - \text{Th0.time}_\text{K}$. We report in this study results for the following time points: 2 h, 4 h, 24 h, 48 h, 72 h.

Following [63], we computed positive and negative scores for each gene at each time point by fitting a beta-uniform mixture model using the implementation in the BioNet package [24]. For a detailed description of this procedure, see Supplementary Text A.2. Throughout this study, FDR = 0.1 was used for all samples and species.

Due to the experimental noise and paired design, the human samples have much higher intra-group variance, resulting in significant calls having p-values orders of magnitude higher than the mouse calls. This results in a range of scores that is much narrower for human than for mouse, possibly imbalancing results towards mouse modules. To correct for this effect, scores of mouse genes were rank normalized to the scores of the human genes as follows: the scores were sorted, and for each gene the score of the i-th mouse gene was set to the score of the i-th human gene. Comparison of the distribution of scores before and after normalization showed that compared to

usual Benjamini-Hochberg FDR and log fold change cut-offs ($|\log \mathrm{FC}| \geq 1$), the loss in statistical power was inconsequential and that this procedure ensured that mouse and human genes had comparable score distributions.

7.3.3 Network and orthology databases

The human and mouse background networks were downloaded from STRING v9.1, protein.actions.detailed.v9.1.txt [78], which is a database that contains experimentally verified direct protein interactions. Note that this network also contains interactions predicted based on orthology, so-called *interologs*. Ideally, we would prefer to use only experimentally predicted interactions, but currently, for mouse, such available data is too incomplete to result in a meaningful background network. Outlier nodes with a degree above 40 times the interquartile range plus the 75th percentile of the distribution of all node degrees were removed (ELAVL1, UBC, Ubb, Ubc). The resulting mouse network has $16,821$ nodes and $483,532$ edges and the human network has $16,255$ nodes and $315,442$ edges.

Orthology information was downloaded from Ensembl release 59 [76] and all human and mouse orthologs were kept, regardless of the identity scores. The orthology mapping corresponds to a bipartite graph involving $67,304$ human proteins and $43,953$ mouse proteins linked by $104,007$ edges, grouped in $16,552$ bicliques with an average size of 6.72 proteins (SD: 5.34).

7.3.4 Implementation, input and output

xHeinz is implemented in modern C++, using the boost libraries and the LEMON graph library [60]. CPLEX 12.6 is used to solve the ILP. The source code is publicly available in a git repository linked to from http://software.cwi.nl/xheinz.

xHeinz takes as input (i) two species-specific networks, (ii) an orthology mapping between the nodes of the two networks, (iii) scores associated to each of the nodes, *e.g.*, derived from the p-value of the moderated t-test, (iv) the threshold value α, and (v) an optional time limit.

We performed a preprocessing step where we retained the subgraphs of the input networks induced by the genes that meet the microarray filtering criteria. This reduced the number of nodes to 8,453 human nodes, 6,882 mouse nodes and 14,779 nodes in the orthology mapping. Among these, up to 250 nodes (depending on the time point) have positive scores. The rank normalization as described in Sect. 7.3.2 ensured that the number of positive human nodes is in the order of the number of positive mouse nodes.

xHeinz returns two node sets corresponding to a solution found within the time limit together with an upper bound on the optimal solution value. In case the solution value equals this upper bound, the computed solution is provably optimal.

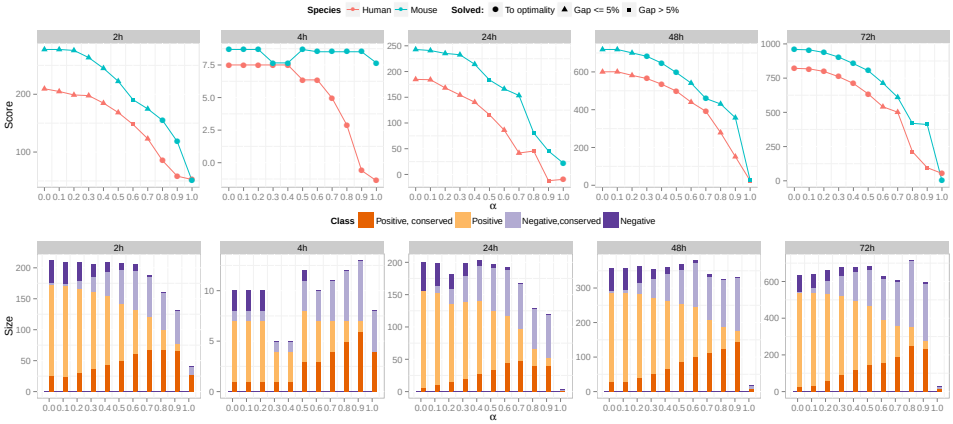


Figure 7.3: **Statistics of xHeinz solutions.** The conserved active module problem was solved for five time points (columns) over a sequence of 11 consecutive values of the α conservation parameter (x-axis). We report in the top row the score of the best solution (y-axis) and whether optimality was proven by our algorithm (circles). The second row illustrates how module contents vary as α increases. The height of each bar indicates the size of the respective module, colors indicate the fraction of positive and conserved nodes.

7.4 Results and Discussion

7.4.1 xHeinz identifies conserved modules at different levels of conservation

We applied xHeinz on samples from the Th17 human and mouse data sets for time points 2 h, 4 h, 24 h, 48 h and 72 h. We solved these instances for different values of $\alpha \in [0,1]$ with a step size of 0.1. All computations were done in single-thread mode on a desktop computer (Intel XEON e5 3 Ghz) with 16 Gb of RAM and a time limit of 12,000 CPU seconds. After this timeout, the best feasible solution is returned by the solver.

Figure 7.3 shows for the five time points and eleven values of the α parameter, the human and mouse scores of the found modules as well as the distribution of the module contents. For 26 of the 55 instances we solved the conserved active modules problem to provable optimality within the time and memory limit. The optimality gap of a solution is defined as $(UB - LB)/|LB|$, where LB and UB are the value of the best solution and the lowest upper bound as identified by the branch-and-cut algorithm, respectively. Of the 29 instances that are not solved to optimality, 22 have a gap smaller than 5%.

Any feasible solution for a conservation ratio of α is also a solution for any $\alpha' \leq \alpha$. We indeed see in Fig. 7.3 that this property holds, the solution values decrease monotonically with increasing α. Also the solutions for $\alpha = 0$ (no conservation constraints) are identical to the solutions obtained by running the single species method

Heinz [63] separately on the two networks.

There is a sharp decrease in module size for $\alpha = 1$. Indeed, this is the most restrictive setting since it enforces that all the nodes in a module must be conserved. We also observe that as α increases, both positive and negative *conserved* nodes are added, indicating that we manage to retrieve informative nodes in a gradual manner. See also Supplementary Text A.8 for a detailed analysis of module overlap for all combinations of α values.

When we compare solutions across time points, we see that the conserved active modules capture two phases of the differentiation process. We observe high activity at 2 h as well as at the late time points. Several authors reported such biphasic behavior during early Th17 differentiation, both in mouse [46, 224] and human [198]. The low activity score observed at the 4 h time point is in line with previous mouse studies, which suggest that after the initial induction sustained by Stat3 and Stat1 in the first four hours, a phase of Rorc induction takes place and lasts until the 20 h time point, after which the effective protein level of Rorc starts to increase and to trigger the cytokine production phase [224]. Our model and the solutions obtained suggest that these dynamics are conserved between the two organisms.

7.4.2 Early regulation of Th17 differentiation is conserved between human and mouse

In the following, we study the two phases of the Th17 differentiation process in more detail. We focus on the 2 h and 48 h time points. We selected for this evaluation $\alpha = 0.8$ for both time points, as this value provides a balance between conservation and activity and produces modules of interpretable size. All results at all time points are available on the accompanying website. Fig. 7.4 reports the resulting human and mouse modules for the two time points.

We assess statistical significance of the resulting modules by performing 100 runs on randomized networks for each value of α, and additional 400 runs for the selected $\alpha = 0.8$. We do this using two randomization methods: (1) permuting the node weights while keeping the graph fixed, and (2) permuting the network topology while keeping the node weights and the node degrees fixed as described in [88]. With the exception of a few extreme cases at the 48 h time point, all modules were found to be highly significant. For details see Supplementary Text A.8.

At the 2 h time point, xHeinz identifies a conserved module consisting of 58 human and 50 mouse proteins. Interestingly, both the human and mouse modules are centered around STAT3/Stat3. STAT3 is a signal transducer having transcription factor activity and was shown to play a key role in the differentiation process of Th17 [96]. Once activated by Th17 polarizing cytokines (such as IL6 in our case), it eventually binds to the promoter regions of IL17A/Il17a and IL17F/Il17f cytokines and activates transcription. These cytokines are the hallmark cytokines produced by activated Th17 cells. It is worth noting that IL17/Il17 cytokines and associated receptors are not in the 2 h modules, as these proteins have been shown to be expressed only at later time points [198]. Moreover, STAT1/Stat1, another member of the STAT family, is part of the solution and belongs to the central core of the human

and mouse modules, which is consistent with its major role during the early phases of Th17 differentiation [224].

We also observe that the STAT3/BATF/IL6ST/SOCS3 region of the 2 h module is well-conserved. Batf has been shown to directly control Th17 differentiation in mouse [172] and BATF proteins are detected as early as after 12 h of polarization in human [198]. Similarly, SOCS3 is a known IL6 and IL21-induced negative regulator of Th17 polarization, that is eventually down-regulated by TGFB and IL6ST at a later phase in order to prolong STAT3 activation [163, 227]. Overall, these modules show highly conserved and significant enrichment for response to cytokine stimulus (Benjamini-Hochberg (BH) FDR 5.6e-4), JAK-STAT (BH FDR 4.8e-4) cascade and transcription regulator activity (BH FDR 2.3e-4), computed using the DAVID functional annotation chart [104]. This indicates that the identified module matches expected biological mechanisms observed at early phases [46]. Furthermore, comparison of the dynamics of expression shows that genes differentially expressed in both species change expression in the same direction (cf. Supplementary Text A.3).

We also applied xHeinz to find a conserved module at a later time point (48 h). Kinetics analysis of Th17 differentiation showed that the effective secretion of Th17 hallmark cytokines only happens after several days of polarization [198, 224] and we do observe in these modules a significant enrichment for interleukin related proteins present in both species, which was absent for the 2 h modules, such as up-regulation of IL9/Il9. Secretion of IL9 by Th17 cells have been demonstrated both in mouse and human cells [26], Il9 is know to be induced by Bcl3 [167], and Bcl3 inhibition has been recently shown to affect the function of Th17 cells in mouse [170]. We also observe the conserved down-regulation of GATA3/Gata3, which is known to be the master regulator of Th2 cells [226], and is likely to constrain the Th17 regulation program [203]. Similarly to the modules found at 2 h, the 48 h modules are centered around STAT3, although at the 48 h time point this gene is not differentially expressed anymore neither in human or mouse (resp. logFC of 0.17, score of -4.59 for human, and logFC 0.52, score of -3.21 for mouse). This observation is in line with the major role of STAT3 along the differentiation process at all time points [224]. To the contrary, STAT1 has been indicated as an exclusively early regulator [224] in mouse and is indeed not present anymore in the 48 h modules. We also observe the presence of the RORA/RORC/Rora/Rorc members of the RORs family of intracellular transcription factors, which are considered to be the master regulators of the Th17 lineage [221], and have been implicated in both species [53]. Interestingly, these regulators are linked to the up-regulation of the vitamin-D receptor (VDR/Vdr), whose role in Th17 differentiation and several human auto-immune related disease have been recently studied [42].

In summary, our findings show the relevance of the identified conserved active modules with regard to the biological process of interest. By requiring the active modules to contain a certain fraction of conserved nodes, xHeinz identifies the main core proteins involved in the differentiation of Th17. Our analysis confirms that these proteins are very likely to have similar roles in both species.

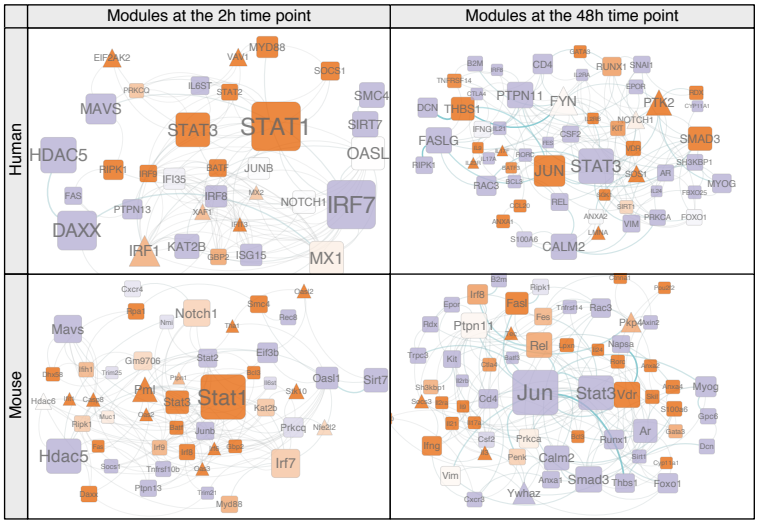


Figure 7.4: **Conserved active Th17 differentiation modules in human and mouse at** 2 h **and** 48 h**.** We obtained node activity scores capturing the significance of differential gene expression between the Th17 and Th0 conditions in human and mouse using the BUM model with FDR = 0.1. xHeinz uses these scores to search for conserved active modules in the STRING protein action network. The first row shows the human counterparts of the best scoring conserved modules for the 2 h (left) and 48 h (right) samples. The second row depicts the mouse counterparts. Rounded squares depict genes for which a homolog—as defined by Ensembl—is present in the counterpart, whereas triangles denote non-conserved genes. Node color gradually indicates activity scores. Orange: larger than 2; white: between −2 and 2; violet: smaller than −2. Node labels and sizes are proportional to betweenness centrality and edge width to edge-betweenness—both centralities are with respect to the sub-network module. Only nodes having a degree larger than 2 (resp. 3) are displayed for the 2 h (resp. 48 h) module. The full networks are available on the accompanying website and in Supplementary Text A.3.

7.4.3 Comparison to neXus

We compare the 48 h xHeinz modules (*cf.* Fig. 7.4) with subnetworks computed by neXus version 3 [59]. neXus uses a heuristic technique to grow subnetworks from seed nodes simultaneously in two species. This is done in an iterative fashion. Neighborhoods of the two current modules are determined using a depth-first search. This search is restricted to only consider nodes that have a path to the seed node with a confidence larger than the user-specified parameter `dfscutoff`. The confidence of a path is defined as the product of the confidences of the edges comprising that path. The modules are extended to include the most active pair of orthologous nodes in the neighborhoods—where activity is defined as normalized log fold change and thus differs from the definition of activity used in xHeinz. This whole procedure is repeated until either the cluster coefficient drops below the user-specified parameter `cc`, or the average activity scores of one of the two modules drops below parameter `scorecutoff`. We ran neXus with the default parameters `cc` = 0.1, 0.2, `scorecutoff` = 0.15 and `dfscutoff` = 0.3, 0.8 for mouse and human respectively for all time points. Table 7.1 gives the resulting module sizes for human and mouse.

Table 7.1: **Modules calculated with neXus for all time points.** Shown are the sizes in number of nodes of the first 15 representative solutions and the average sizes for the human subnetwork and for the mouse subnetwork in brackets. The last column lists the number of solutions for each time point. No solutions were obtained for time points 24 h and 72 h.

solution	1	2	3	4	5	6	7	8	9	10	11	12	13
0.5h	7 (6)	4 (4)	7 (6)	3 (3)									
1h	15 (10)	10 (9)	12 (12)	13 (13)	7 (7)	5 (5)	15 (13)	10 (11)	9 (10)	18 (16)	25 (24)	14 (14)	6 (7)
2h	15 (17)	6 (5)	12 (10)	12 (11)	10 (10)	13 (13)	17 (15)	12 (12)	8 (9)	5 (5)	11 (12)	19 (18)	3 (3)
4h	6 (9)	4 (4)	6 (5)	4 (4)	4 (4)	3 (3)	7 (8)	9 (10)	4 (4)	3 (3)			
48h	5 (5)												

solution	14	15	avg.	#sols
0.5h			5.25 (4.75)	4
1h	5 (5)	6 (6)	9.95 (9.58)	19
2h	9 (8)	23 (21)	10 (9.83)	30
4h			5 (5.40)	10
48h			5 (5)	1

neXus finds 1 module for time point 48 h which is shown in Fig. 7.5 for human (A) and mouse (B). In total 5 genes are contained in the module, which are identical for human and mouse, but the number of edges differs. Only one of the genes is significantly differentially expressed, CCL20, which has an absolute log fold change bigger than 1 and a BH FDR smaller than 0.1. Since neXus does not use p-values as an input, but log fold-changes which are normalized to activity values, the genes CCL20 and CXCR3 are considered as active nodes with a value above 0.15. These genes show changes in expression, but only two of these changes are statistically significant. The low number of active nodes points to a drawback in the neXus algorithm: due to the locality of the greedy search strategy it may happen that the average activity of the subnetwork in construction keeps on degrading without reaching the next active node. The effects of this issue can be seen, for example, in Fig. 7.5, where CCL20

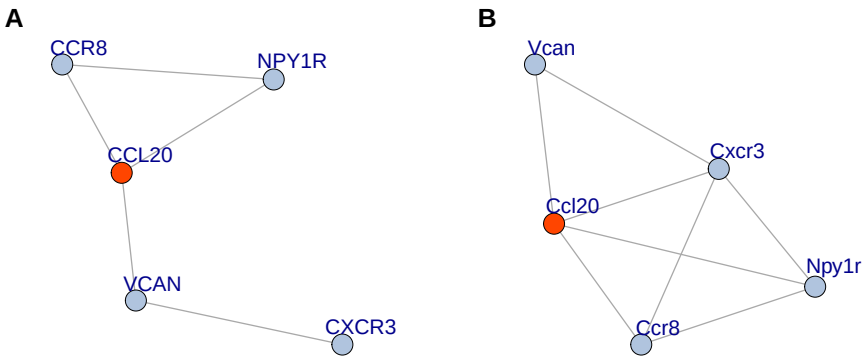


Figure 7.5: **neXus module for the time point 48 hours for human (A) and mouse (B).** Orange coloring indicates genes with significant differential expression (BH FDR ≤ 0.1, $|\log \mathrm{FC}| \geq 1$). Here only one gene is significantly differentially expressed (CCL20).

is the seed node and the majority of other neighboring nodes are not differentially expressed.

Another consequence of the neXus search strategy is that the module sizes are small (*cf.* Tab. 7.1) and thus only give a limited view of the molecular mechanisms at play. Theoretically, the parameter `dfscutoff` can be decreased to increase the module size. Doing so, however, produces only slightly larger modules, but drastically increases the running time (Supplementary Table 1). Changes in the clustering coefficient parameter `cc` only reduce the module size with increasing `cc` (Supplementary Table 2).

Conservation in neXus is enforced stringently by only allowing pairs of orthologous genes or genes that are only present in one of the networks to be included in the subnetworks (see Fig. 7.5). This is too restrictive if the underlying mechanisms in the two species differ. For instance, for time point 48 hours and all but $\alpha = 1$ values, xHeinz finds the non-conserved gene IL23R (BH FDR 3.52e-8, score 14.50, logFC 1.38) in human, which is involved in Th17 autocrine signaling [213] but which is not differentially expressed in mouse. xHeinz also finds JUNB, which at the 2 hour time point is up-regulated in human data (BH FDR 1e-2, score 0.02, logFC 1.3) and not detected as differentially expressed in the mouse data (BH FDR 0.48, score -4.01, logFC 0.65). JUNB is a known partner of BATF with which it heterodimerizes preferentially during Th17 differentiation [172], indicating its relevance. Both important genes would have been missed by a more restrictive conservation setting. Indeed, both neXus and xHeinz at $\alpha = 1$ fail to find these genes showing that a more flexible view on conservation is required to adequately deal with transferability.

7.5 Conclusion

We introduce a mathematical model for the problem of finding active subnetwork modules that are conserved between two species and thus contribute to formalizing the notion of conserved active modules. A key feature of our model is a flexible notion of conservation, which is controlled by a parameter $\alpha \in [0,1]$: We require that at least a fraction α of the nodes are conserved between the species-specific modules of a solution. Note that in case of more distantly-related species a smaller α value may be more appropriate. We have translated our model into an integer linear programming formulation and have devised and implemented an exact branch-and-cut algorithm that computes provably optimal or near-optimal conserved active modules in our model.

Our computational experiments for understanding the mechanisms underlying Th17 T cell differentiation in both mouse and human demonstrate that the flexibility in the definition of conservation is crucial for the computation of meaningful conserved active modules. We have found two conserved Th17 modules at time points 2 h ($\alpha = 0.8$) and 48 h ($\alpha = 0.8$) that thoroughly encompass the biphasic Th17 differentiation process. This result can not be revealed by requiring full conservation ($\alpha = 1$) or by independent modules without requiring conservation ($\alpha = 0$). Likewise, neXus, an alternative approach based on a stringent conservation model, is not able to capture the key regulatory program of the differentiation process.

A key characteristics of our model is its flexibility. This allows its extension to multiple species and time points, which we will address in future work. In this case, however, realistic instances will be harder to compute to optimality and will require the development of powerful algorithm engineering techniques.

Acknowledgements. We thank the three anonymous referees for their constructive comments.

7.6 Supplementary material

7.6.1 Integer linear programming approach

The integer programming problem is formulated in the main text as follows.

$$\max \sum_{v \in V_1 \cup V_2} w_v x_v \tag{7.7}$$

$$\text{s.t. } m_u = \max_{uv \in R} \{x_u x_v\} \qquad u \in V_1 \tag{7.8}$$

$$m_v = \max_{uv \in R} \{x_u x_v\} \qquad v \in V_2 \tag{7.9}$$

$$\sum_{v \in V_1 \cup V_2} m_v \geq \alpha \sum_{v \in V_1 \cup V_2} x_v \tag{7.10}$$

$$G_1[\mathbf{x}] \text{ and } G_2[\mathbf{x}] \text{ are connected} \tag{7.11}$$

$$x_v, m_v \in \{0,1\} \qquad v \in V_1 \cup V_2 \tag{7.12}$$

This formulation satisfies the properties of activity, conservation and modularity.

Activity. Variables $\mathbf{x} \in \{0,1\}^{V_1 \cup V_2}$ encode the presence of nodes in the solution, i.e., for all $v \in V_1 \cup V_2$ we want $x_v = 1$ if $v \in V^*$ and $x_v = 0$ otherwise. The objective function (7.7) uses these variables to express the activity of the solution, which we aim to maximize.

Conservation. Variables $\mathbf{m} \in \{0,1\}^{V_1 \cup V_2}$ encode the presence of conserved nodes in the solution. Recall that a node $u \in V_1^*$ ($u \in V_2^*$) that is present in the solution is conserved if there is another node $v \in V_2^*$ ($v \in V_1^*$) in the solution such that the two nodes form a conserved node pair $uv \in R$ ($vu \in R$). This corresponds to constraints (7.8) and (7.9). We linearize $x_u x_v$, in a standard way, by introducing binary variables $\mathbf{z} \in \{0,1\}^R$ such that $z_{uv} = x_u x_v$ for all $uv \in R$:

$$z_{uv} \leq x_u \qquad uv \in R \tag{7.13}$$
$$z_{uv} \leq x_v \qquad uv \in R \tag{7.14}$$
$$z_{uv} \geq x_u + x_v - 1 \qquad uv \in R \tag{7.15}$$
$$z_{uv} \in \{0,1\} \qquad uv \in R \tag{7.16}$$

Subsequently, we model the max function in (7.8) and (7.9) as follows.

$$m_u \geq z_{uv} \qquad uv \in R \tag{7.17}$$
$$m_v \geq z_{uv} \qquad uv \in R \tag{7.18}$$
$$m_u \leq \sum_{uv \in R} z_{uv} \qquad u \in V_1 \tag{7.19}$$
$$m_v \leq \sum_{uv \in R} z_{uv} \qquad v \in V_2 \tag{7.20}$$

We model the required degree of conservation by constraint (7.10).

Modularity. Constraint (7.11) states that the nodes encoded in the solution $\mathbf{x}$ induce a connected subgraph in both G_1 and G_2. There are many ways to model connectivity, *e.g.*, using flows or cuts [143]. Cut-based formulations perform better in practice [61]. Recently, Álvarez-Miranda et al. [11] have introduced a cut-based formulation that only uses node variables. In an empirical study, the authors show that their formulation outperforms other cut-based formulations. We model connectivity along the same lines. Since the constraints that we will describe are similar for both graphs, we introduce them only for graph $G_1 = (V_1, E_1)$.

$$\sum_{v \in V_1} y_v \leq 1 \tag{7.21}$$
$$y_v \leq x_v \qquad v \in V_1 \tag{7.22}$$
$$x_v \leq \sum_{u \in \delta(S)} x_u + \sum_{u \in S} y_u \qquad v \in V_1, \{v\} \subseteq S \subseteq V_1 \tag{7.23}$$
$$y_v \in \{0,1\} \qquad v \in V_1 \cup V_2 \tag{7.24}$$

where $\delta(S) = \{v \in V_1 \setminus S \mid \exists u \in S : uv \in E_1\}$ denotes the *neighbors* of S. The modularity property states that $\mathbf{x}$ should induce a connected subgraph in G_1. We model this by

introducing binary variables $\mathbf{y} \in \{0,1\}^{V_1}$ that determine the root node. Constraints (7.21) and (7.22) state that at most one node $v \in V_1^*$ is the root node—in which case $y_v = 1$. Constraints (7.23) state that x_v can only be 1 if for all sets $S \subseteq V$ containing v it holds that either the root node is in S or there is a neighbor u of S in the solution. There is an exponential number of such constraints. We therefore do not add all these constraints to our initial formulation. Instead, we use a branch-and-cut approach, that is, at every node of the branch-and-bound tree we identify all violated constraints and add them to the formulation. Finding violated inequalities corresponds to solving a minimum cut problem, which we do using the algorithm by Boykov and Kolmogorov [35].

To further improve the performance, we have strengthened our model with the following constraints.

$$y_v = 0 \qquad v \in V, w_v < 0 \tag{7.25}$$

$$\sum_{u \in V} y_u \geq x_v \qquad v \in V, w_v \geq 0 \tag{7.26}$$

$$y_v \leq 1 - x_u \qquad u, v \in V, u < v, w_u \geq 0, w_v \geq 0 \tag{7.27}$$

$$x_v \leq \sum_{u \in \delta(\{v\})} x_u + y_v \qquad v \in V \tag{7.28}$$

As an optimization, we require that the root node must be a non-negatively weighted node in (7.25). Constraints (7.26) state that if a non-negatively weighted node v is present in the solution then there must be a root node. Constraints (7.27) are symmetry breaking constraints, they require that among all non-negatively weighted nodes in the solution, the root node is the smallest one—according to some arbitrary order. Finally, constraints (7.28) correspond to the cases where the set S in (7.23) is a singleton.

7.6.2 The beta-uniform mixture model

The method proceeds as follows. First, similarly to Pounds and Morris [161], the distribution of the gene-wise p-values $x = x_1, \ldots, x_n$ is described as a beta-uniform mixture (BUM) model, which is a mixture of a $B(a,1)$ beta distribution (signal) and a uniform distribution (noise): $\lambda + (1-\lambda)ax^{a-1}$, for $0 < a < 1$, with mixture parameter λ and shape parameter a of the beta distribution. The log likelihood is defined as $\log \mathscr{L}(\lambda, a; x) = \sum_{i=1}^{n} \log(\lambda + (1-\lambda)ax_i^{a-1})$, and consequently the maximum-likelihood estimations of the unknown parameters are given by $[\hat{\lambda}, \hat{a}] = \arg\max_{\lambda,a} \mathscr{L}(\lambda, a; x)$. The parameter estimates have been obtained using numerical optimization. As detailed in [161] the BUM model allows the estimation of a false discovery rate (FDR) that can be controlled via a p-value threshold $\tau(\text{FDR})$. The adjusted log likelihood ratio score is then defined as

$$s(x, \text{FDR}) = \log \frac{\hat{a}x^{\hat{a}-1}}{\hat{a}\tau(\text{FDR})^{\hat{a}-1}} = (\hat{a}-1)(\log(x) - \log(\tau(\text{FDR}))) \,.$$

Genes whose differential expression is considered significant given the FDR threshold obtain a positive score while genes showing no differential expression will receive a negative score. The size of the resulting module can be regulated with the FDR parameter.

7.6.3 Data processing pipeline

The full pipeline (implemented using Snakemake) from data downloading to running xheinz goes as follows:

1. Retrieve human and mouse ENSEMBL orthologs, STRING species specific network
2. Retrieve human dataset from GEO (all time points, all conditions)
3. Annotate human probes with ENSEMBL
4. Select human probes based on variance filter
5. Perform linear modeling of the whole human dataset
6. Retrieve mouse dataset (all time points, all conditions)
7. Annotate mouse probes with ENSEMBL
8. Select mouse probes based on variance filter
9. Perform linear modeling of the whole mouse dataset
10. For each time point of interest:
    a) Call differentially expressed human genes by contrasting the Th17 with the Th0 condition ⇒ p-value for each gene at this time point
    b) Call differentially expressed mouse genes by contrasting the Th17 with the Th0 condition ⇒ p-value for each gene at this time point
    c) For an FDR threshold of 0.1, fit a BUM model for the human genes ⇒ positive and negative scores for human genes
    d) For an FDR threshold of 0.1, fit a BUM model for the mouse genes ⇒ positive and negative score for mouse genes
    e) Rank normalize the mouse score based on the human scores ⇒ update the scores of the mouse genes
    f) Map human and mouse genes to proteins on the STRING network
    g) For each value of interest for the conservation threshold α:
        i. Run xHeinz with the following inputs:
            A. the human STRING network
            B. the mouse STRING network
            C. the BUM scored human proteins
            D. the BUM scored mouse proteins
            E. the human and mouse orthologs

7.6.4 Module details

Full Th17 module figures. Figure 7.6 and Figure 7.7 show the full, unfiltered module contents of the Th17 modules described in Sect. 4.2 of the main text.

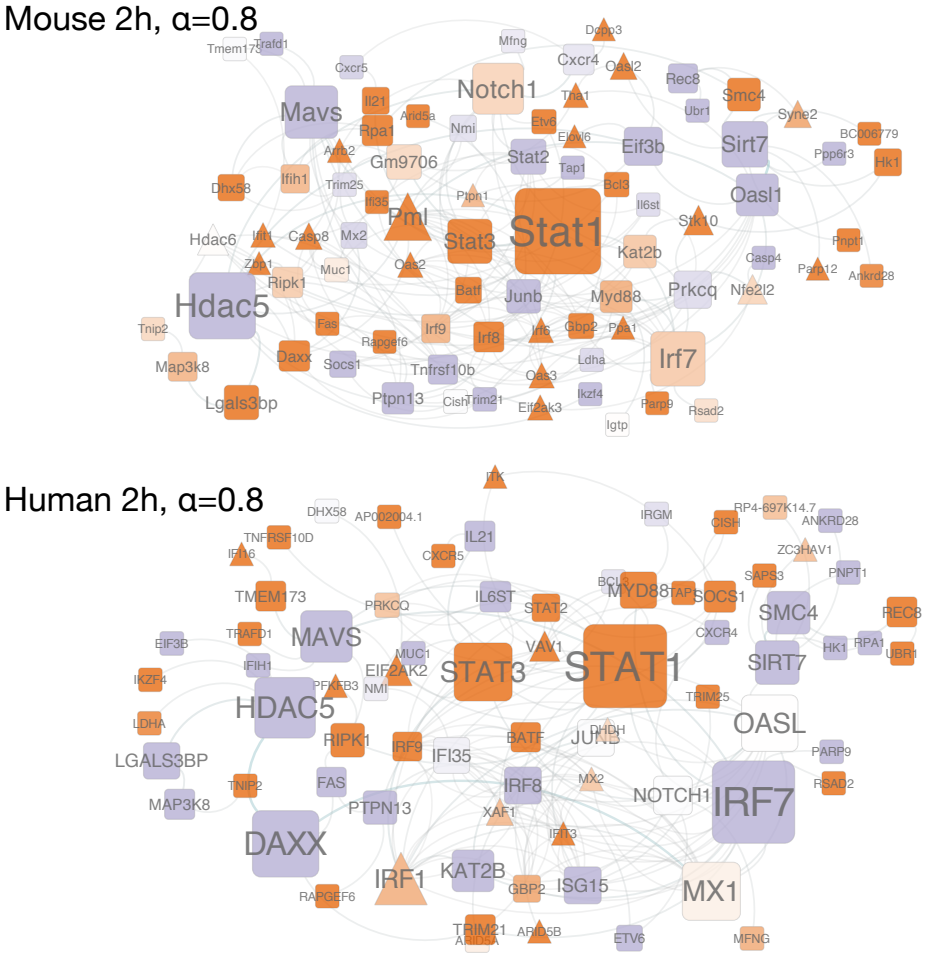


Figure 7.6: Full module at 2 h for $\alpha = 0.8$ conservation ratio.

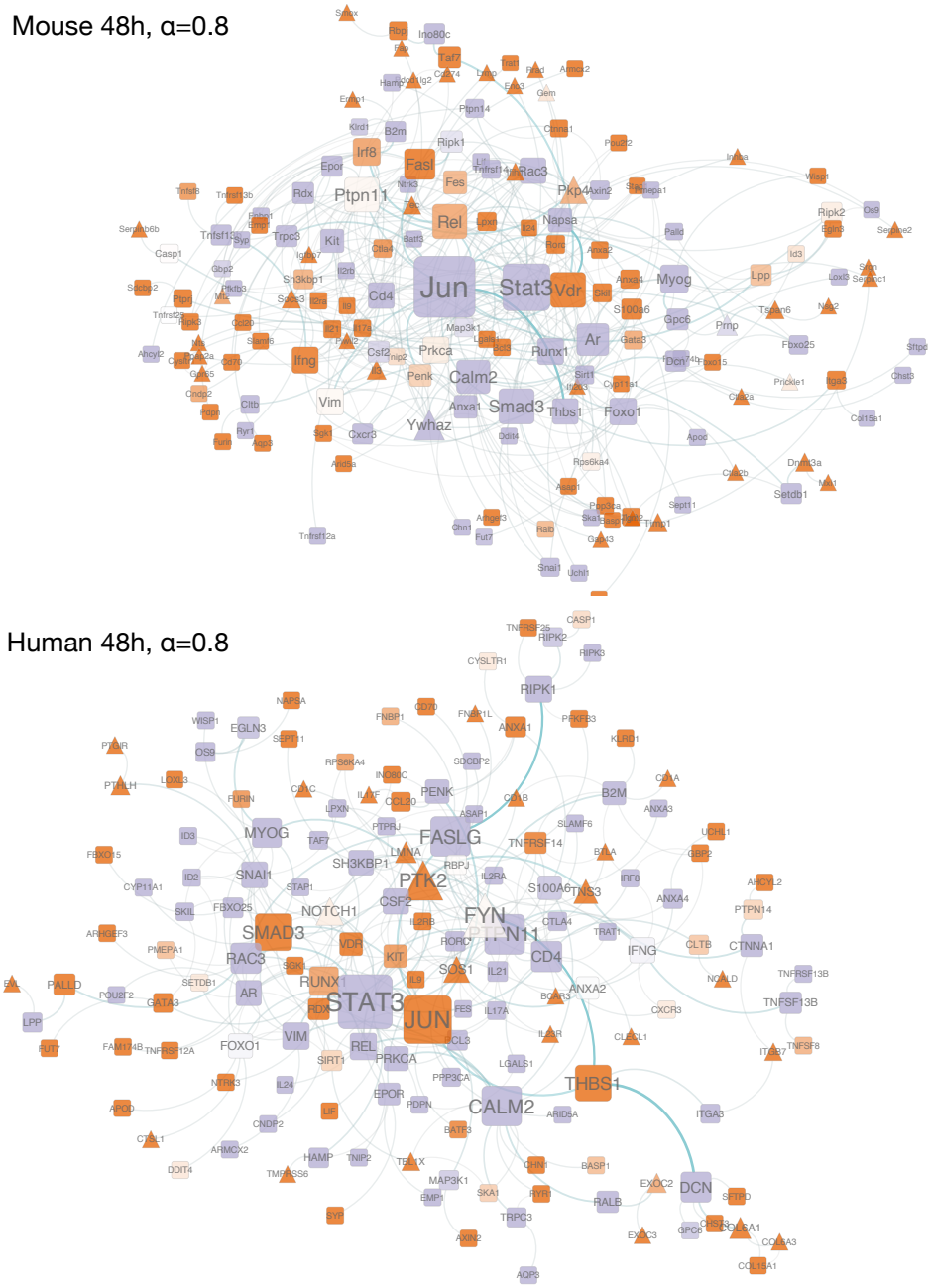


Figure 7.7: Full module at 48 h for $\alpha = 0.8$ conservation ratio.

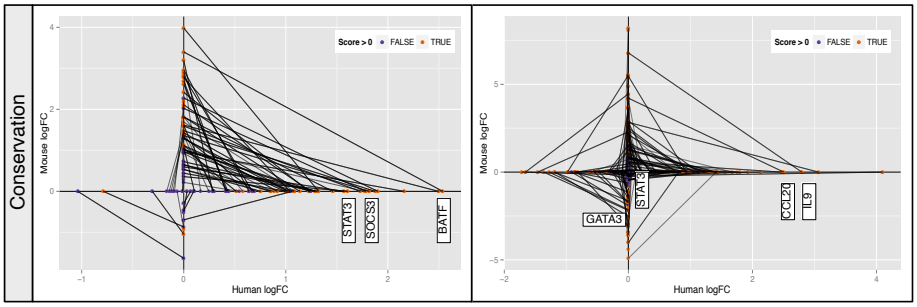


Figure 7.8: **Comparison of log fold change expression in mouse and human for conserved gene pairs at** 2 h **(left) and** 48 h **(right)** Each panel shows the log fold change correlation between conserved gene pairs: For each pair, a line segments connects the human logFC (x-axis) to the mouse logFC (y-axis). Point color indicates whether the human or mouse gene has a positive score. A line segment in the 1st or 3rd quadrant signifies positively correlated logFC values whereas a link in the 2nd and 4th quadrant corresponds to negative correlation. The sign of the activity score is indicated by the coloring. Genes discussed in the main text are indicated with white boxes.

Full Th17 module tables. In addition we provide a tabular overview of the module contents (Figures 7.3 and 7.5). In the tables, each row lists unmatched genes or homologuous gene pairs with their activity scores. On the left side are the mouse genes and on the right side the human genes. The gene activity scores are in parentheses. Names in bold represent genes for which no homologous counterpart exist in the underlying networks.

Note that very few (overall only 2 in mouse for 48 h) genes with negative activity score and without conserved counterpart are selected. Some of them even have a strong positive activity score. These genes would be missed in a strict conservation model.

7.6.5 Conservation of dynamics

The overall dynamics in human and mouse at 2 h are well conserved, as illustrated in Fig. 7.8. The plots show that in our modules all but two conserved gene pairs change expression in the same direction. It is worth noting that we do not enforce any conservation of directionality in the xHeinz model.

7.6.6 Effect of the `dfscutoff` on neXus solutions

Table 7.6 shows the neXus solutions for time point 48 h with different values for parameter `dfscutoff`.

Mouse	Human
Batf (8.29)	BATF (13.90)
Stat3 (4.22)	STAT3 (8.81)
Rapgef6 (5.67)	RAPGEF6 (6.68)
Stat1 (4.07)	STAT1 (7.95)
Rec8 (-2.85)	REC8 (11.56)
Rsad2 (0.44)	RSAD2 (7.99)
Cish (-0.08)	CISH (8.04)
Tnip2 (0.53)	TNIP2 (6.04)
Ifi35 (6.68)	IFI35 (-0.44)
Irf9 (1.07)	IRF9 (4.89)
Gbp2 (3.85)	GBP2 (1.21)
Parp9 (7.95)	PARP9 (-2.98)
Tnfrsf10b (-3.47)	TNFRSF10D (7.99)
Trim21 (-3.08)	TRIM21 (7.58)
Etv6 (7.90)	ETV6 (-3.54)
Stat2 (-1.88)	STAT2 (6.21)
Bcl3 (4.82)	BCL3 (-0.88)
Tmem173 (-0.15)	TMEM173 (3.96)
Dhx58 (3.96)	DHX58 (-0.20)
Fas (7.58)	FAS (-4.21)
Myd88 (1.11)	MYD88 (2.15)
Trim25 (-0.96)	TRIM25 (4.22)
Socs1 (-2.82)	SOCS1 (5.96)
Ubr1 (-3.27)	UBR1 (6.15)
Lgals3bp (6.52)	LGALS3BP (-3.64)
Trafd1 (-2.87)	TRAFD1 (5.67)
Ikzf4 (-3.34)	IKZF4 (6.14)
BC006779 (1.85)	RP4-697K14.7 (0.87)
Ripk1 (0.71)	RIPK1 (1.93)
Tap1 (-4.38)	TAP1 (6.91)
Arid5a (2.06)	ARID5A (0.23)
Saps3 (-3.51)	SAPS3 (5.62)
Daxx (5.36)	DAXX (-3.31)
Casp4 (-2.56)	AP002004.1 (4.14)
Cxcr5 (-1.36)	CXCR5 (2.63)
Notch1 (0.60)	NOTCH1 (-0.06)
Ldha (-1.10)	LDHA (1.61)
Mfng (-0.63)	MFNG (1.10)
Hk1 (4.89)	HK1 (-4.47)
Prkcq (-1.03)	PRKCQ (0.84)
Irf8 (2.70)	IRF8 (-3.31)
Igtp (0.02)	IRGM (-0.92)
Mx2 (-1.23)	MX1 (0.20)
Pnpt1 (3.12)	PNPT1 (-4.46)
Nmi (-0.87)	NMI (-0.55)
Il21 (2.68)	IL21 (-4.69)
Ifih1 (1.00)	IFIH1 (-3.19)
Oasl1 (-2.29)	OASL (0.01)
Rpa1 (1.82)	RPA1 (-4.36)
Ankrd28 (1.66)	ANKRD28 (-4.34)
Smc4 (2.15)	SMC4 (-5.12)
Muc1 (0.27)	MUC1 (-3.25)
Isg15 (0.53)	ISG15 (-3.57)
Kat2b (0.79)	KAT2B (-4.23)
Junb (-4.02)	JUNB (-0.08)
Map3k8 (1.02)	MAP3K8 (-5.33)
Cxcr4 (-0.74)	CXCR4 (-3.63)
Il6st (-1.06)	IL6ST (-3.37)
Irf7 (0.74)	IRF7 (-5.39)
Mavs (-1.97)	MAVS (-5.26)
Eif3b (-4.46)	EIF3B (-3.51)
Hdac5 (-3.20)	HDAC5 (-4.84)
Sirt7 (-4.06)	SIRT7 (-5.13)
Ptpn13 (-3.87)	PTPN13 (-5.40)
Oas2 (15.42)	
Elovl6 (13.90)	
Ppa1 (11.56)	
Pml (8.81)	
Zbp1 (8.55)	
Ifit1 (8.02)	
Parp12 (7.99)	
Oas3 (6.15)	
Arrb2 (5.67)	
Casp8 (3.69)	
Eif2ak3 (2.80)	
Irf6 (2.63)	
Stk10 (2.30)	
Syne2 (1.23)	
Ptpn1 (1.03)	
Nfe2l2 (0.59)	
Hdac6 (0.04)	
Dcpp3 (6.14)	
Tha1 (4.61)	
Oasl2 (2.19)	
	ITK (6.49)
	ARID5B (4.61)
	PFKFB3 (3.78)
	VAV1 (3.00)
	IFIT3 (2.70)
	EIF2AK2 (2.29)
	IRF1 (1.07)
	XAF1 (0.97)
	ZC3HAV1 (0.90)
	DHDH (0.55)
	IFI16 (6.52)
	MX2 (0.78)

Table 7.3: Timepoint 2 h, $\alpha = 0.8$, FDR = 0.1

Mouse	Human
Napsa (-3.86)	NAPSA (32.07)
Il9 (5.03)	IL9 (22.36)
Il21 (32.07)	IL21 (-4.97)
Cd70 (7.24)	CD70 (19.85)
Cysltr1 (22.36)	CYSLTR1 (0.30)
Il17a (27.88)	IL17A (-5.46)
Fbxo15 (3.11)	FBXO15 (18.26)
Wisp1 (24.44)	WISP1 (-4.63)
Slamf6 (22.16)	SLAMF6 (-5.38)
Rbpj (14.79)	RBPJ (0.04)
Il24 (17.69)	IL24 (-3.66)
Emp1 (16.64)	EMP1 (-2.83)
Chn1 (-5.02)	CHN1 (18.50)
Egln3 (14.50)	EGLN3 (-1.79)
Ccl20 (9.84)	CCL20 (2.50)
Aqp3 (17.54)	AQP3 (-5.38)
Vdr (2.22)	VDR (9.74)
Apod (-4.84)	APOD (16.41)
Pdpn (16.86)	PDPN (-5.29)
Ahcyl2 (-3.39)	AHCYL2 (14.79)
Sgk1 (4.04)	SGK1 (5.55)
Lgals1 (12.94)	LGALS1 (-3.52)
Lpxn (14.49)	LPXN (-5.30)
Anxa2 (8.87)	ANXA2 (-0.12)
Loxl3 (-4.22)	LOXL3 (12.94)
Tnfrsf25 (-0.02)	TNFRSF25 (7.75)
Arhgef3 (4.10)	ARHGEF3 (2.72)
Basp1 (5.83)	BASP1 (0.89)
Gata3 (1.19)	GATA3 (5.24)
Fut7 (-1.83)	FUT7 (7.58)
Furin (4.28)	FURIN (1.46)
Ryr1 (-4.25)	RYR1 (8.54)
Axin2 (-4.95)	AXIN2 (7.83)
Tnfrsf13b (6.13)	TNFRSF13B (-3.57)
Tnfsf8 (1.41)	TNFSF8 (1.04)
Il2rb (-3.52)	IL2RB (5.91)
Fasl (7.75)	FASLG (-5.37)
Anxa3 (7.83)	ANXA3 (-5.48)
Jun (-2.95)	JUN (5.14)
Sdcbp2 (5.24)	SDCBP2 (-3.21)
Klrd1 (-1.64)	KLRD1 (3.51)
Uchl1 (-5.07)	UCHL1 (6.91)
Rps6ka4 (0.22)	RPS6KA4 (1.34)
Ifng (1.77)	IFNG (-0.32)
S100a6 (2.72)	S100A6 (-1.60)
Pfkfb3 (-5.16)	PFKFB3 (6.07)
Rdx (-3.21)	RDX (4.10)
Palld (-4.63)	PALLD (5.31)
Gbp2 (-1.37)	GBP2 (2.04)
Sftpd (-5.12)	SFTPD (5.71)
Asap1 (4.97)	ASAP1 (-4.42)
Col15a1 (-4.82)	COL15A1 (5.34)
Casp1 (0.03)	CASP1 (0.48)
Chst3 (-5.14)	CHST3 (5.61)
Arid5a (5.34)	ARID5A (-4.90)
Lif (-2.49)	LIF (2.92)
Thbs1 (-4.55)	THBS1 (4.87)
Fam174b (-3.95)	FAM174B (4.09)
Ptprj (5.47)	PTPRJ (-5.43)
Syp (-3.59)	SYP (3.45)
Trat1 (5.14)	TRAT1 (-5.35)
Tnfrsf12a (-4.23)	TNFRSF12A (3.90)
Id2 (5.02)	ID2 (-5.38)
Armcx2 (4.41)	ARMCX2 (-4.95)
Cyp11a1 (2.69)	CYP11A1 (-3.46)
Itga3 (2.81)	ITGA3 (-3.60)
Batf3 (-2.57)	BATF3 (1.64)
Pmepa1 (-2.08)	PMEPA1 (0.89)
Ctla4 (1.49)	CTLA4 (-2.69)
Anxa1 (-4.66)	ANXA1 (3.43)
Skil (2.30)	SKIL (-3.67)
Ripk2 (0.25)	RIPK2 (-1.64)
Ino80c (-3.62)	INO80C (2.22)
Stap1 (3.19)	STAP1 (-4.67)
Tnip2 (0.48)	TNIP2 (-1.99)
Smad3 (-3.58)	SMAD3 (1.95)
Sept11 (-5.00)	SEPT11 (3.21)
Tnfrsf14 (-3.50)	TNFRSF14 (1.70)
Ntrk3 (-5.01)	NTRK3 (3.19)
Ctnna1 (3.21)	CTNNA1 (-5.04)
Taf7 (3.41)	TAF7 (-5.28)
Ripk3 (1.93)	RIPK3 (-4.05)
Ddit4 (-2.48)	DDIT4 (0.31)
Anxa4 (3.28)	ANXA4 (-5.53)
Id3 (0.38)	ID3 (-2.65)
Foxo1 (-2.09)	FOXO1 (-0.28)
Map3k1 (-0.93)	MAP3K1 (-1.47)
Rorc (2.25)	RORC (-4.79)
Cndp2 (1.21)	CNDP2 (-3.76)
Il2ra (1.95)	IL2RA (-4.53)
Cxcr3 (-2.88)	CXCR3 (0.29)
Ppp3ca (1.84)	PPP3CA (-4.64)
Setdb1 (-3.16)	SETDB1 (0.31)
Pou2f2 (2.50)	POU2F2 (-5.48)
Lpp (0.89)	LPP (-4.05)
Sh3kbp1 (1.09)	SH3KBP1 (-4.29)
Bcl3 (2.19)	BCL3 (-5.51)
Irf8 (1.29)	IRF8 (-5.04)
Fes (1.03)	FES (-4.82)
Runx1 (-5.09)	RUNX1 (1.29)
Kit (-5.24)	KIT (1.41)
Ptpn14 (-4.47)	PTPN14 (0.58)
Fnbp1 (-5.10)	FNBP1 (1.09)
Rel (1.20)	REL (-5.37)
Ralb (0.98)	RALB (-5.21)
Sirt1 (-4.83)	SIRT1 (0.60)
Ska1 (-5.08)	SKA1 (0.66)
Cltb (-5.23)	CLTB (0.76)
Penk (0.66)	PENK (-5.23)
Csf2 (-1.22)	CSF2 (-3.46)
Ptpn11 (0.12)	PTPN11 (-4.91)
Vim (0.10)	VIM (-5.10)
Prkca (0.24)	PRKCA (-5.36)
Gpc6 (-4.26)	GPC6 (-1.59)
Ripk1 (-0.81)	RIPK1 (-5.60)
Calm2 (-2.76)	CALM2 (-4.28)
Snai1 (-5.08)	SNAI1 (-2.21)
Fbxo25 (-4.90)	FBXO25 (-2.49)
Stat3 (-3.22)	STAT3 (-4.60)
Epor (-5.17)	EPOR (-3.07)
Rac3 (-4.44)	RAC3 (-3.86)
B2m (-4.57)	B2M (-4.45)
Os9 (-5.20)	OS9 (-4.11)
Tnfsf13b (-4.10)	TNFSF13B (-5.24)
Trpc3 (-4.28)	TRPC3 (-5.21)
Dcn (-4.39)	DCN (-5.27)
Ar (-4.20)	AR (-5.67)
Myog (-5.25)	MYOG (-5.01)
Cd4 (-4.94)	CD4 (-5.53)
Hamp (-5.21)	HAMP (-5.53)
Gap43 (22.44)	
Nts (18.50)	
Inhba (13.50)	
Tec (10.10)	
Serpine2 (9.94)	
Lrmp (9.92)	
Eno3 (9.74)	
Serpinc1 (9.69)	
Fap (9.64)	
Ifih1 (8.54)	
Tspan6 (8.41)	
Smox (6.91)	
Ppap2a (6.35)	
Piwil2 (6.11)	
Cd274 (5.61)	
Ermp1 (5.31)	
Tgm2 (5.21)	
Srgn (4.72)	
Serpinb6b (4.53)	
Mxi1 (4.39)	
Dnmt3a (4.32)	
Nsg2 (3.71)	
Igfbp7 (3.45)	
Gpr65 (3.30)	
Pdcd1lg2 (3.01)	
Rrad (2.97)	
Socs3 (1.90)	
Mt2 (1.31)	
Pkp4 (1.12)	
Gem (0.37)	
Prickle1 (0.30)	
Prnp (-1.10)	
Ywhaz (-3.30)	
Timp1 (18.48)	
Ifi203 (18.26)	
Ctla2b (10.18)	
Ctla2a (6.54)	
Il3 (1.86)	
	PTHLH (24.44)
	TMPRSS6 (16.86)
	COL6A3 (16.64)
	IL23R (14.50)
	PTK2 (11.74)
	EXOC3 (10.18)
	BCAR3 (9.84)
	IL17F (8.67)
	PTGIR (6.95)
	ITGB7 (6.54)
	COL6A1 (5.47)
	TNS3 (5.20)
	LMNA (4.70)
	NCALD (4.41)
	EVL (4.28)
	BTLA (3.41)
	FNBP1L (1.93)
	SOS1 (1.84)
	EXOC2 (1.19)
	NOTCH1 (0.23)
	FYN (0.08)
	CD1A (27.88)
	CLECL1 (22.16)
	CD1C (18.48)
	CD1B (14.49)
	CTSL1 (13.50)
	TBL1X (4.04)

Table 7.5: Timepoint 48 h, $\alpha = 0.8$, FDR = 0.1

Table 7.6: **neXus solutions for time point** 48 h **with different values for parameter** `dfscutoff`**.** Shown are the number of solutions, the average and maximum number of nodes and running times for the human network.

dfscutoff	no. sols.	avg. size	max. size	CPU time [s]
0.1	2	5.00	5	176039.91
0.2	3	7.00	9	110282.61
0.3	5	6.60	9	77502.13
0.4	4	6.62	10	54972.92
0.5	6	5.58	9	35360.42
0.6	4	5.38	6	20538.4
0.7	3	5.33	6	11164.19
0.8	60	4.04	6	4639.59
0.9	0	0.00	0	1359.87

7.6.7 Effect of the `clustering coefficient` on neXus solutions

Table 7.7 shows the neXus solutions for time point 48 h with different values for parameter `cc`.

Table 7.7: **neXus solutions for time point** 48 h **with different values for parameter** `cc`**.** Shown are the number of solutions, the average and maximum number of nodes and running times for the human network.

cc	no. sols.	avg. size	max. size	CPU time [s]
0.1	1	5	5	23367.68
0.2	1	5	5	23243.83
0.3	1	5	5	23903.09
0.4	1	5	5	24285.30
0.5	1	3	3	24318.78
0.6	1	3	3	24059.59
0.7	1	3	3	24518.68
0.8	1	3	3	24321.92
0.9	1	3	3	24246.56

7.6.8 Significance of results

We computed empirical p-values to assess the significance of the obtained scores at 2 h and 48 h, for each value of $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$, with the following two procedures:

1. *Weights permutation:* We shuffled the node weights by generating random permutations of the activity scores of all genes.

2. *Topology permutation:* We repeated a million times the following operation: given two randomly selected edges A1–A2 and B1–B2, if the edges A1–B2 and B1–A2 are not present in the network we add them and remove the original edges, thus generating a random network with the same node weights and degree distribution.

For each resulting permuted network we applied a modified version of xHeinz a fixed number of times (500 times for $\alpha = 0.8$, 100 times otherwise). This modified version allows to check whether the optimal score on the new random network would exceed the best score we found on the original network.

To speed up these computations, we used the observation that solving a relaxed version of the conserved modules problem is sufficient, since we only need a sufficiently low upper bound (less than the score we obtained with unshuffled data) on the optimal values for those shuffled instances to make this decision.

We therefore consider the following ILP for the shuffled instances:

$$\max \sum_{v \in V_1 \cup V_2} w_v x_v \tag{7.29}$$

$$\text{s.t. } (7.8), (7.9), (7.10), (7.12) \tag{7.30}$$

that is, we drop the connectivity constraints. Note that the objective value is an upper bound of the optimal score of the original problem since this new ILP is a relaxation of the original one. In addition, we can stop the branch-and-cut algorithm as soon as the upper bound on (7.29) is lower than the best found solution of the original ILP on the unshuffled data. These two observations help to speed up the significance computations tremendously.

Table 7.8 shows the results of these runs, which we performed using a cumulative time limit of 500 h. It can be seen that only at extreme values of α for the 48 h time point the upper bound on (7.29) was not good enough to prove significance. This result was actually expected, since the original run was the only one with a very high gap (33.36%) at the end of the allocated timelimit.

Note that the p-values $\hat{p}$ that we demonstrate here are actually upper bounds to the underlying p-values p that we would obtain without the relaxation to the ILP.

For all other combinations, including the ones we chose to compute the Th17 modules and where we computed even more permutations, our procedure demonstrates that the signal in the real network is useful to obtain a statistically significant score.

7.6.9 Robustness of modules for varying α

Given two modules $V_1' \subseteq V_1$ and $V_2' \subseteq V_2$, the Jaccard index is defined as $(V_1' \cap V_2')/(V_1' \cup V_2')$. Fig. 7.9 shows for each time point the Jaccard index for all pairs of conservation ratios. For consecutive values of α we can see that the module contents do not change much.

Fig. 7.10 shows the human gene module contents for time point 2 h for varying values of α, one gene per line. Genes in the left panel (FALSE) are negatively scored,

Timepoint	α	FDR	k	k'	$\hat{p}$
2 h	0.1	0.1	100	0	0.0
2 h	0.2	0.1	100	0	0.0
2 h	0.3	0.1	100	0	0.0
2 h	0.4	0.1	100	0	0.0
2 h	0.5	0.1	100	0	0.0
2 h	0.6	0.1	100	0	0.0
2 h	0.7	0.1	100	0	0.0
2 h	**0.8**	**0.1**	**500**	**0**	**0.0**
2 h	0.9	0.1	100	0	0.0
2 h	1.0	0.1	100	0	0.0
48 h	0.1	0.1	100	7	**0.07**
48 h	0.2	0.1	100	4	**0.04**
48 h	0.3	0.1	100	0	0.0
48 h	0.4	0.1	100	0	0.0
48 h	0.5	0.1	100	0	0.0
48 h	0.6	0.1	100	0	0.0
48 h	0.7	0.1	100	0	0.0
48 h	**0.8**	**0.1**	**500**	**0**	**0.0**
48 h	0.9	0.1	100	0	0.0
48 h	1.0	0.1	100	100	**1.0**

Timepoint	α	FDR	k	k'	$\hat{p}$
2 h	0.1	0.1	100	0	0.0
2 h	0.2	0.1	100	0	0.0
2 h	0.3	0.1	100	0	0.0
2 h	0.4	0.1	100	0	0.0
2 h	0.5	0.1	100	0	0.0
2 h	0.6	0.1	100	0	0.0
2 h	0.7	0.1	100	0	0.0
2 h	**0.8**	**0.1**	**500**	**0**	**0.0**
2 h	0.9	0.1	100	0	0.0
2 h	1.0	0.1	100	0	0.0
48 h	0.1	0.1	100	7	0.0
48 h	0.2	0.1	100	4	0.0
48 h	0.3	0.1	100	0	0.0
48 h	0.4	0.1	100	0	0.0
48 h	0.5	0.1	100	0	0.0
48 h	0.6	0.1	100	0	0.0
48 h	0.7	0.1	100	0	0.0
48 h	**0.8**	**0.1**	**500**	**0**	**0.0**
48 h	0.9	0.1	100	0	0.0
48 h	1.0	0.1	100	20	**0.2**

Table 7.8: Results of significance experiments. For $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$ at time points 2 h and 48 h we computed upper bounds on k instances with permuted scores using the procedures described above (weights permutation on the left, topology permutation on the right). In k' of these cases, resulting upper bound was not lower than the score of the best found conserved active module on the original network, resulting in a p-value $p \leq \hat{p} = k'/k$. Values at the threshold $\alpha = 0.8$ we used to compute the Th17 modules and non-zero p-values are highlighted.

that is, they are not differentially expressed at an FDR of 0.1. Genes in the right panel (TRUE) have a positive score. Squares represent conserved genes, whereas triangles represent non-conserved genes. An xHeinz module solution is thus the set of genes marked with a square or triangle at a given value of α. The coloring is as follows:

- Genes that are selected at all values of α or are coloured red. These are the core genes.
- Genes that are selected at $\alpha = 0$ (no conservation) but not at $\alpha = 1$ are green.
- Genes that are selected by $\alpha = 1$ (strict conservation) but not at $\alpha = 0$ are turquoise.
- Genes that can only be picked by xHeinz (intermediate α values) are orchid.

This figure shows that a large fraction of genes only occur in an intermediate α regime (orchid color). Furthermore, the module content smoothly changes as α

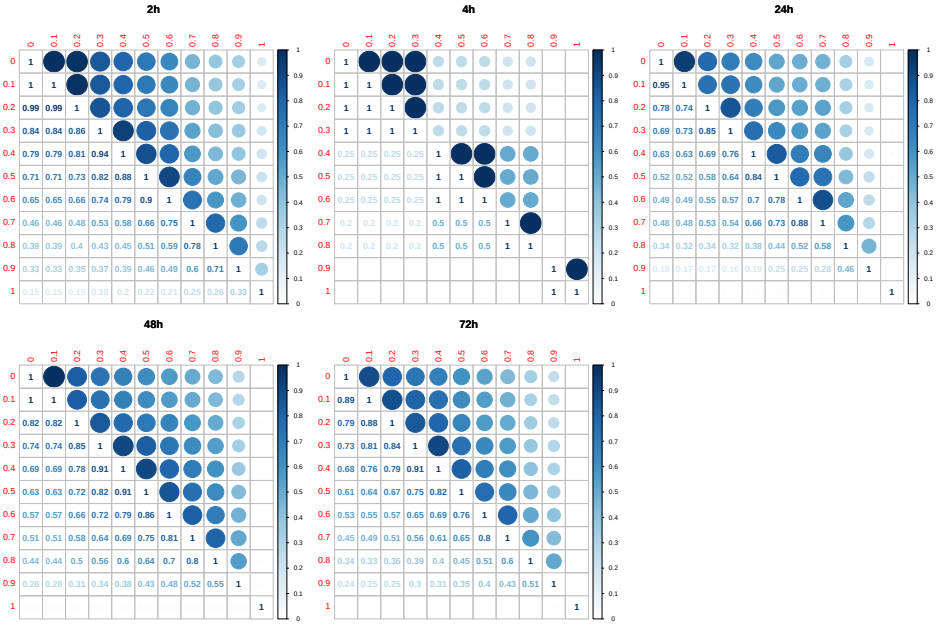


Figure 7.9: Jaccard index evolution over changes of α

varies, which illustrates the robustness of the approach. xHeinz thus allows the investigator to make an informed choice on the conservation of genes in the modules, which is not possible with methods assuming no or strict conservation.

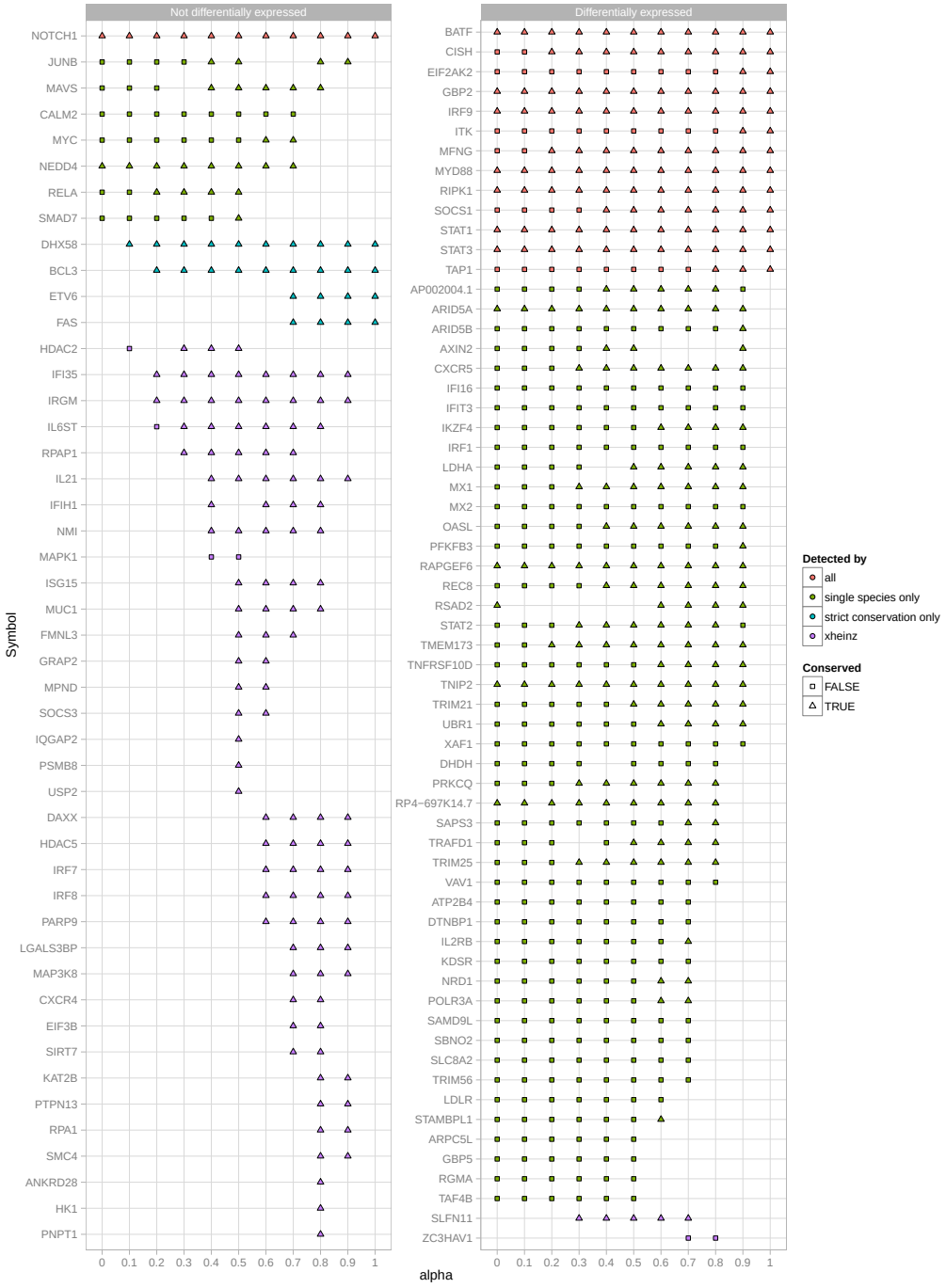


Figure 7.10: Human module contents of the 2 h time point with varying α.

Chapter 8

Charge group partitioning

Published as:

S. Canzar[†], M. El-Kebir[†], R. Pool, K. M. Elbassioni, A. K. Malde, A. E. Mark, D. P. Geerke, L. Stougie, and G. W. Klau. Charge group partitioning in biomolecular simulation. In *Research in Computational Molecular Biology, RECOMB 2012, Barcelona, Spain, April 21–24, 2012*, pages 29–43, 2012.

S. Canzar[†], M. El-Kebir[†], R. Pool, K. Elbassioni, A. K. Malde, A. E. Mark, D. P. Geerke, L. Stougie, and G. W. Klau. Charge Group Partitioning in Biomolecular Simulation. *Journal of Computational Biology*, 20(3):188–198, Mar. 2013.

[†]joint first authorship

Abstract

Molecular simulation techniques are increasingly being used to study biomolecular systems at an atomic level. Such simulations rely on empirical force fields to represent the intermolecular interactions. There are many different force fields available—each based on a different set of assumptions and thus requiring different parametrization procedures. Recently, efforts have been made to fully automate the assignment of force-field parameters, including atomic partial charges, for novel molecules. In this work, we focus on a problem arising in the automated parametrization of molecules for use in combination with the GROMOS family of force fields: namely, the assignment of atoms to charge groups such that for every charge group the sum of the partial charges is ideally equal to its formal charge. In addition, charge groups are required to have size at most k. We show NP-hardness and give an exact algorithm that solves practical problem instances to provable optimality in a fraction of a second.

Keywords: charge groups, atomic force fields, GROMOS, biomolecular simulation, tree decomposition, dynamic programming

8.1 Introduction

In the context of drug development, biomolecular systems such as protein-peptide [220], protein-ligand [179] and protein-lipid interactions [34] can be studied with the use of molecular simulations [6, 202] using a force field model that describes the interatomic interactions. Many biomolecular force fields are available, including AMBER [52], CHARMM [37], OPLS [115] and GROMOS [155, 171, 175]. These force fields have in common that the non-bonded intermolecular interactions are represented in terms of interatomic pair potentials.

Typically, the number of atoms in biomolecular systems are in the range of 10^4 to 10^6. To observe relevant biological phenomena, time scales in the order of nano- to milliseconds need to be simulated. For such large-scale systems, evaluating all atom-atom interactions is practically infeasible. One way of dealing with this is to only consider interactions of atoms whose distance is within a pre-specified cut-off radius. Since not all interactions are considered, an error is introduced. The magnitude of the *error* due to omitting atom-atom interactions is inversely proportional to the distance between the atoms. More problematically, there are *discontinuities* as atoms move in and out of the cut-off radius.

Errors and discontinuities are reduced by combining atoms into *charge groups*, for which individual centers of geometry are determined. If the distance between two centers of geometry lies within the cut-off distance then all interactions between the atoms of the involved charge groups are considered. Ideally, charge groups should be neutral as interactions are then reduced to dipole-dipole interactions that scale inversely proportional to the cubed interatomic distance. Charge groups should not be too large. This is because the effective cut-off distance of an individual atom in a given charge group is given by the cut-off distance minus the distance to the center of geometry of the charge group. If the distance of an atom to the center of geometry becomes large, the effective cutoff becomes small, leading to errors and discontinuities as described above. For the same reason, charge groups should be connected as interatomic bonds impose spatial proximity.

To simulate a molecule, a force field requires a specific *topology*, which includes the atom types, bonds and angles, the atomic charges and the charge group assignment. Most biomolecular force fields come with a set of topologies for frequently simulated molecules such as amino acids, lipids, nucleotides and cofactors. Unparametrized molecules, however, require the construction of their topologies. Such a situation occurs, for instance, when assessing the binding affinity of a novel drug-like compound to a certain protein.

Manually building topologies for new compounds is a tedious and time-consuming task especially when a large chemical library needs to be screened, for example when determining binding affinities for large sets of potential drug compounds to a newly discovered protein target. Therefore, automated approaches are needed.

Here, we focus on the GROMOS family of force fields, which has been specifically tailored to simulate biochemical processes, including protein-drug binding and peptide folding. A widely used topology generator for the GROMOS force field is PRODRG [174]. However, the charge group assignment by PRODRG for amino acid topologies contained several large charge groups comprising disconnected atoms, which is in-

consistent with GROMOS [139]. The Automated Topology Builder (ATB) is a recent method for automated generation of GROMOS topologies [144]. The assignment of atomic charges and charge groups by the ATB proceeds in three consecutive stages. Firstly, partial charges are computed using quantum calculations. Subsequently, the symmetry of the molecule is exploited to ensure that symmetric atoms have identical charges. Finally, the molecule is partitioned into charge groups using a greedy algorithm. The ATB method was experimentally verified for a set of biologically relevant molecules [144]. For some large molecules, such as the cofactor Adenosine-5'-triphosphate (ATP), however, the ATB assigns too large charge groups, which leads to instabilities during simulation as described above.

As existing automated procedures such as PRODRG and the ATB fail in assigning appropriate charge groups, we have investigated the problem in detail. Our contribution is threefold: (1) We introduce the charge group partitioning problem and give a sound mathematical problem definition resulting in charge groups of small size and zero charge. We prove NP-hardness of the problem and identify important special cases, for which we give polynomial time algorithms. (2) Exploiting the properties of molecular structures enables us to present a tree decomposition-based algorithm that solves typical practical problem instances to optimality within fractions of a second. (3) We evaluate the performance of our method by running simulations using the resulting charge group assignments of amino acid side chains, which yield results consistent with experimentally known values. Moreover, for large, highly charged, molecules such as ATP we obtain charge groups which are both suitable for use in simulations as well reasonable from a chemical perspective.

8.2 Problem statement and complexity

In this section we give a formal definition of the problem associated with assigning appropriate charge groups within a molecule. Our aim is to capture the two important aspects of chemical intuition discussed above: (1) the number of atoms in a charge group should not exceed a given integer k and (2) the sum of partial and formal charges of a charge group is ideally equal. Mathematically, the latter condition is equivalent to requiring the sum of differences of formal and partial charges in a charge group to be close to zero. We prove NP-hardness of the problem even if we take into account special characteristics of graphs representing a molecular structure. For the special case $k = 2$ we obtain a polynomial-time algorithm by reducing the problem to a minimum cost perfect matching problem.

A molecular structure can be modeled as a degree-bounded graph $G = (V,E)$, where the nodes correspond to atoms and the edges to chemical bonds. In addition, we consider node weights $\delta : V \to \mathbb{R}$, where $\delta(v)$ corresponds to the difference between formal and partial charge of the atom v. A formal definition of the *charge group partitioning problem* is as follows:

Definition 8.1 (Charge group partitioning, CGP) *Given a graph $G = (V,E)$, node weights $\delta : V \to \mathbb{R}$, and an integer $2 \le k \le |V| - 1$, find a partition $\mathcal{V}$ of V such that for all $V' \in \mathcal{V}$ it holds $|V'| \le k$, the subgraph $G[V']$ induced by V' is connected, and which*

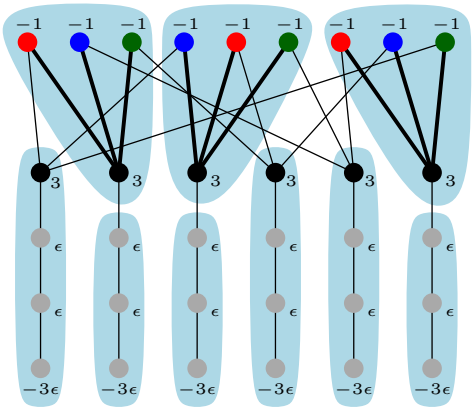


Figure 8.1: Reduction from (see Definition 8.2): every $x \in X$ corresponds to a node with weight $\delta(x) = -1$, whereas every $T \in \mathcal{T}$ corresponds to a node with weight $\delta(T) = 3$. There is an edge between nodes $x \in X$ and $T \in \mathcal{T}$ if and only if $x \in T$. In addition to every $T \in \mathcal{T}$ a path (T, s_1^T, s_2^T, s_3^T) is attached with weights $\delta(s_1^T) = \delta(s_2^T) = \epsilon$ and $\delta(s_3^T) = -3\epsilon$

has minimal total error

$$c(\mathcal{V}) := \sum_{V' \in \mathcal{V}} \left| \sum_{v \in V'} \delta(v) \right| .$$

Each subset $V' \in \mathcal{V}$ of the nodes in the partition corresponds to a charge group. The following theorem shows NP-hardness of the problem, even for the restricted case where G is planar. As we will discuss in Section 8.3, most molecular graphs are planar.

Theorem 8.1 CGP *is NP-hard, even in the restricted case where G is planar, $k = 4$, the maximum degree of a node in the graph is 4, and the node weights are $\mathcal{O}(1)$.*

Proof Clearly, the problem belongs to NP. Consider the following problem.

Definition 8.2 (Planar 3-dimensional matching, PLANAR 3DM) *Given disjoint sets X_1, X_2, X_3 with $|X_1| = |X_2| = |X_3| = m$ and a set of n triples $\mathcal{T} \subset X_1 \times X_2 \times X_3$. The bipartite graph B, with $\mathcal{T}$ as its one color class and $X = X_1 \cup X_2 \cup X_3$ as its other color class and an edge between $T \in \mathcal{T}$ and $x \in X$ if and only if $x \in T$, is planar. Each element of X appears in 2 or 3 triples only. Does there exist a* perfect matching *in $\mathcal{T}$; i.e. a subset $M \subset \mathcal{T}$ of m triples such that each element of X occurs uniquely in a triple in M?*

This problem has been shown NP-complete by Dyer and Frieze [68]. We reduce it to CGP in polynomial time. Take the bipartite graph B in the definition of PLANAR 3DM with $\mathcal{T}$ and X as color classes. Give each $x \in X$ a weight $\delta(x) = -1$ and each $T \in \mathcal{T}$ a weight $\delta(T) = 3$. For each $T \in \mathcal{T}$ we introduce three extra vertices s_1^T, s_2^T, s_3^T with

weights $\delta(s_1^T) = \epsilon$, $\delta(s_2^T) = \epsilon$, $\delta(s_3^T) = -3\epsilon$, for an arbitrary $0 < \epsilon < 1$, and connect them by the path (T, s_1^T, s_2^T, s_3^T), which we call the *tail* of T. See Figure 8.1 for an example. Clearly, the resulting graph G remains planar (and bipartite). Since each $x \in X$ is in at most three triples it is easy to see that G has bounded degree 4.

Given a feasible partition to the CGP-instance, we say a group is of type i if it contains exactly i nodes from X, $i \in \{0,1,2,3\}$ and exactly one node from $\mathcal{T}$. Notice that, for $i = 1,2,3$, each type i group contributes error $(3-i)$ by itself, and because it covers a $\mathcal{T}$-node and therefore leaves a tail-path it contributes indirectly an extra error ϵ (the alternative of including one of the tail nodes into the group with the triple node does not decrease the sum of the two errors). A type 0 group consists of a $\mathcal{T}$-node only and therefore will be combined with its tail to yield an error of $3-\epsilon$. Let y_i denote the number of type i groups, $i \in \{0,1,2,3\}$. Let y denote the number of X-vertices that form a group on their own. Then the feasible solution has total error

$$W = y_0(3-\epsilon) + y_1(2+\epsilon) + y_2(1+\epsilon) + y_3\epsilon + y. \tag{8.1}$$

We show that there exists a perfect matching if and only if G admits a partition with total error

$$W = m\epsilon + (n-m)(3-\epsilon).$$

Suppose $M \subset \mathcal{T}$ is a perfect matching. For every triple $T_i \in M$ we create a type 3 group consisting of the corresponding vertex T_i in G and the three vertices corresponding to its three elements. Hence $y_3 = m$. By the properties of the matching all X-vertices of G are now covered, and $n-m$ triple-vertices of G remain uncovered. The latter necessarily form $n-m$ type 0 groups: $y_0 = n-m$. Insertion in (8.1) yields $W = m\epsilon + (n-m)(3-\epsilon)$.

Now assume that no perfect matching exists. First, note that in any optimal solution to the CGP-instance $y = 0$. Assume $y > 0$ and let $x \in X$ be such a vertex. Then every neighbor of x in $\mathcal{T}$ is contained in a group of type i, with $i \leq 2$. Therefore, adding x to any such group would decrease the cost of the solution by at least $2(1-\epsilon)$. Furthermore, every group that contains two nodes from $\mathcal{T}$ can be split into two groups without increasing the cost of the solution. Now, since there exists no perfect matching, we need $m+c$ groups of type 1,2, or 3, for some $c \geq 1$, to cover all vertices in X. Using equations

$$y_1 + y_2 + y_3 = m + c \tag{8.2}$$

$$y_1 + 2y_2 + 3y_3 = 3m \tag{8.3}$$

we get

$$y_3 = m - 2c + y_1 \tag{8.4}$$

$$y_2 = 3c - 2y_1 \tag{8.5}$$

and the cost contributed to (8.1) by type 1, 2, and 3 groups becomes equal to $m\epsilon + c(3+\epsilon)$. Together with the remaining $n-m-c$ groups of type 0 the total weight becomes

$$m\epsilon + (n-m)(3-\epsilon) + 2c\epsilon.$$

□

Using the same reduction, but extending the tails to length $k-1$ paths with ϵ weight on the internal vertices and $-(k-1)\epsilon$ weight on the leaf, proves the problem to be hard for any $k \geq 4$.

CGP with $k = 2$ can be solved by formulating a minimum cost perfect matching problem. Starting from $G = (V, E)$, we assign a weight to the edges that is equal to the error that the pair of vertices will contribute if chosen as a group of the partition. For each vertex $v \in V$ create a shadow vertex v' with $\delta(v') = 0$. The weight on the edge $\{v, v'\}$ is then $|\delta(v)|$, the error if v is chosen as a single vertex group. Additionally we insert an edge $\{u', v'\}$ of weight 0 if and only if $\{u, v\} \in E$. It is not difficult to see that a minimum cost perfect matching in this graph corresponds to an optimal partition, where an edge in the matching between a vertex and its shadow vertex signifies a single vertex group in the partition.

For $k = 3$ and for general, non-planar graphs CGP is NP-hard by reduction from ordinary 3DM. Intriguingly, for planar graphs and $k = 3$ the complexity is still unknown.

8.3 Dynamic programming for bounded treewidth

While problem CGP is NP-hard in general as shown in the previous section, we can solve it by a dynamic program in polynomial time if the molecule graph is a tree. Starting from the leaves we proceed towards an arbitrarily chosen root node. At a given node i we guess the group V' that contains i in the optimal solution to the subproblem induced by the subtree rooted at i and recurse on the subtrees obtained when removing V'. Due to the size restriction $|V'| \leq k$ we only have to consider a polynomial number of groups.

Although the structural formula of biomolecules is not always a tree, as we will see later, it is usually still tree-like, which has already been exploited in [57]. Formally, this property is captured by the *treewidth* of a graph [169]. The definition is as follows.

Definition 8.3 *A* tree decomposition (T, X) *of a graph* $G = (V, E)$ *consists of a tree* T *and sets* X_i *for all* $i \in V(T)$*, called* bags*, satisfying the three following properties:*

1. *Every vertex in* G *is associated with at least one node in* T*:* $\bigcup_{i \in V(T)} X_i = V$.
2. *For every edge* $\{u, v\} \in E$*, there is an* $i \in V(T)$ *such that* $\{u, v\} \subseteq X_i$.
3. *The nodes in* T *associated with any vertex in* G *define a subtree of* T.

The width *of a tree decomposition is* $\max_i |X_i| - 1$*. The* treewidth *of* G *is the minimum width of any tree decomposition of* G.

In this section, we propose a tree decomposition-based dynamic program for problem CGP whose running time grows exponentially with the treewidth of G. Therefore, a tree decomposition of small width is crucial for the efficiency of our approach. Unfortunately, computing a tree decomposition of minimum width is NP-hard [14]. However, for the class of r-outerplanar graphs an optimal tree decomposition can be

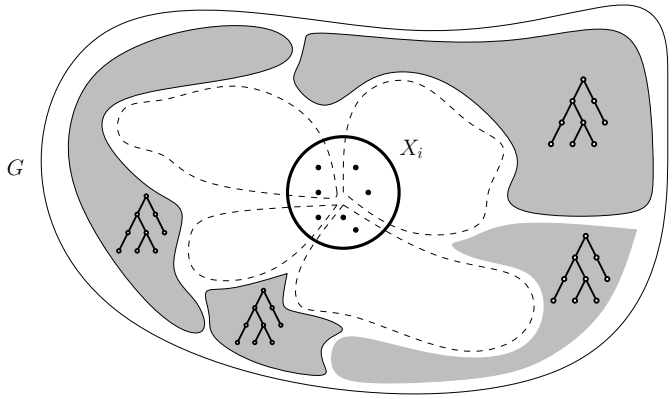


Figure 8.2: Illustration of the tree decomposition-based dynamic programming algorithm. A graph G falls apart into connected components (grey regions) by removing the groups (dashed lines) that intersect bag X_i.

determined in time $\mathcal{O}(r \cdot n)$ [2]. A graph is r-outerplanar if, after removing all vertices on the boundary face, the remaining graph is $(r-1)$-outerplanar. A graph is 1-outerplanar if it is outerplanar, that is, if it admits a crossing-free embedding in the plane such that all vertices are on the same face. Interestingly enough, most molecule graphs of biomolecules are r-outerplanar for some small integer r. For example, Horváth *et al.* [103] have observed that 94.3% of the molecules in the NCI database[1] are 1-outerplanar. Even more, every r-outerplanar graph has treewidth at most $3r-1$ [33]. Therefore, not surprisingly, Yamaguchi et al. [218] observed that out of 9,712 chemical compounds in the KEGG LIGAND database [90], all but one had treewidth between 1 and 3, with a single molecule having treewidth 4. In fact, among the molecules considered here the maximal treewidth was 2. As a result, our tree decomposition-based dynamic program found an optimal charge group partitioning in well under one second.

Let (T,X) be a tree decomposition of width ℓ for graph $G=(V,E)$. The high-level idea of the algorithm is as follows, see also Figure 8.2. For an arbitrarily chosen root i of the tree decomposition we guess the groups that intersect X_i, denoted by the dashed lines in the figure. After removing these groups, G falls apart into connected components, denoted by the filled regions in the figure. By the properties of a tree decomposition, these connected components will correspond one-to-one to the subtrees of the tree decomposition obtained by removing bags that became empty. Recursing on the roots of these new subtrees yields the overall optimal solution.

Without loss of generality we assume that T has at most $n := |V|$ vertices and depth $\mathcal{O}(\log n)$ [32], with r being the root of T. In the following, we let $V_i = \bigcup_{j \in T_i} X_j$, where T_i denotes the subtree rooted at i, and write $V(T_i)$ for the set of nodes in T_i. We define an extension of a partition of a vertex set $V_1 \subseteq V$ with nodes in $V_2 \setminus V_1$ into connected subgraphs of G of size at most k:

[1] National Cancer Institute (http://cactus.nci.nih.gov/)

Definition 8.4 *For vertex sets $V_1 \subseteq V_2 \subseteq V$, set $\mathscr{L}_G(V_1, V_2)$ contains all sets $\mathscr{V} \in 2^{V_2}$ with $V_1 \subseteq \bigcup_{V' \in \mathscr{V}} V'$, all sets in $\mathscr{V}$ being disjoint, and all $V' \in \mathscr{V}$ satisfying: (i) $G[V']$ is connected, (ii) $|V'| \leq k$ and (iii) $V_1 \cap V' \neq \emptyset$.*

Furthermore, by $r(S)$ we denote the root of a subtree S of T, and for any node i in T and any vertex set $A \subseteq V$ we denote by $\mathscr{S}(i, A)$ the set of trees, corresponding to the connected components of $T_i[j \in V(T_i) \mid X_j \setminus A \neq \emptyset]$ whose roots are not a descendant of another subtree in $\mathscr{S}$ (i.e. there are no $S_i, S_j \in \mathscr{S}$ for which $V_{r(S_i)} \subseteq V_{r(S_j)}$). With a slight abuse of notation, for sets $A \subseteq V$ and $\mathscr{V} \subseteq 2^V$ we will write $A \cup \mathscr{V}$ instead of $\bigcup_{V' \in \mathscr{V}} V' \cup A$, when the meaning is clear from the context. Then for any node i of T and any subset $A \subseteq V$ the cost of an optimal solution to CGP on graph $G[V_i \setminus A]$, denoted by $cgp(i, A)$, can be described by the recurrence

$$cgp(i, A) = \min_{\mathscr{V} \in \mathscr{L}_G(X_i \setminus A, V_i \setminus A)} \left\{ c(\mathscr{V}) + \sum_{S \in \mathscr{S}(i, A \cup \mathscr{V})} cgp(r(S), A \cup \mathscr{V}) \right\}, \tag{8.6}$$

which also holds in the base case where $\mathscr{S}(i, A \cup \mathscr{V}) = \emptyset$, in particular when i is a leaf of T. The optimal partition has cost $cgp(r, \emptyset)$. We can solve the recurrence relation (8.6) using dynamic programming.

Theorem 8.2 *The cost of an optimal solution to* CGP *on a graph of treewidth ℓ and maximum degree d can be computed in time $n \cdot \mathcal{O}(e^{2k} \ell^4 d^{4k-2} \cdot \log n)^\ell$.*

Proof Let (T, X) be a tree decomposition of G of width k and depth $\mathcal{O}(\log n)$. Consider an arbitrary node i in T and a subset $A \subseteq V$, for which $X_i \setminus A \neq \emptyset$. We first observe that

$$|\mathscr{L}_G(X_i \setminus A, V_i \setminus A)| \leq \left(\frac{e^k d^{2k-1} (\ell + 1)^2}{(d-1)k} \right)^{\ell+1}. \tag{8.7}$$

Indeed, for each partition $\mathscr{Y} = \{Y_1, \ldots, Y_h\}$ of $X_i \setminus A$, the number of possible extensions in $\mathscr{L}_G(X_i \setminus A, V_i \setminus A)$ can be bounded as follows. For $j = 1, \ldots, h$, let B_j be the set of vertices at distance at most $k-1$ from Y_j in the graph $G_j = G[V_i \setminus (A \cup X_i) \cup Y_j]$ (this set can be found by contracting Y_j to a single vertex y_j and performing BFS in G_j starting from y_j). Each possible extension is then given by a family of pairwise-disjoint sets $Z_1, \ldots, Z_h$, where $Z_j \subseteq B_j$, $G[Z_j \cup Y_j]$ is connected and $|Y_j \cup Z_j| \leq k$. Since the degree of each vertex is at most d, it follows that $|B_j| \leq |Y_j| d^{k-1}$. Consequently, the total number of choices of sets Z_j is at most $(\ell+1)e^k d^{2k-1}/(k(d-1))$ (and all these choices can be enumerated in time $\mathcal{O}(d^{2k} k^2 (\ell+1)^2)$ and space $\mathcal{O}(d^{2k}(\ell+1)^2)$); see [199]. Since $h \leq \ell + 1$, the overall number of choices we consider is bounded by (8.7).

Since every $\mathscr{V}$ considered in (8.6) intersects $X_i \setminus A$ (requirement 3), and due to the properties of a tree decomposition and the connectivity of all parts $V' \in \mathscr{V}$ (in G), the induced subgraph $T_i[j \in V(T_i) \mid X_j \cap V' \neq \emptyset]$, for all $V' \in \mathscr{V}$, is a subtree of T_i rooted at i. Keeping this crucial observation in mind, let us focus our attention on a particular node i in T, and bound the number of sets A that we need to consider on the left hand side of equation (8.6). To this end, it is convenient to consider the computation tree $\mathbf{T}$ for (8.6) (that is, the recursion tree obtained when solving (8.6)) in a *top-down* fashion. We can label each node in this tree by (j, A), where j is a node

in T and A is a subset of V. The root of $\mathbf{T}$ is $(r,\emptyset)$ and the children of node (j,A) are labeled by the elements of the set $\{(r(S),A\cup\mathcal{V}) : S\in\mathscr{S}(i,\mathcal{V}),\mathcal{V}\in\mathscr{L}_G(X_i\setminus A,V_i\setminus A)\}$.

Consider node (i,A) in $\mathbf{T}$, and let $(j_1,A_1),\ldots,(j_h,A_h)$ be its ancestors. It is clear that every vertex $v\in A$ belongs to *exactly one* connected component (group) V' that originated at some ancestor (j_r,A_r), i.e, $v\in V'\in\mathcal{V}\in\mathscr{L}_G(X_{j_r}\setminus A_r,V_{j_r}\setminus A_r)$; we say in this case that ancestor (j_r,A_r) contributes to (i,A). Since $X_i\setminus A\neq\emptyset$ (by our assumption that (i,A) appears in the computation tree), it follows by our observation above that the number of ancestors that contribute to (i,A) is at most ℓ (since each such ancestor contributes at least one component that has a non-empty intersection with X_i). In other words, A can be partitioned into at most ℓ parts, such that each part belongs to a connected component that originated at some ancestor of (i,A), and hence, $|A|\leq k\ell$. The number of choices for the contributing ancestors is at most $\mathrm{depth}(T)^\ell$. Using an argument similar to the one used to derive (8.7), we can conclude that for each vertex v in one of the chosen ancestors, the number of connected components originating at v is at most $e^k d^{2k-1}/(k(d-1))$ and thus we obtain $\left(e^k d^{2k-1}\ell\cdot\mathrm{depth(T)}/(k(d-1))\right)^\ell$ for the total number of choices for A. For each such choice we have to evaluate a number of sets $\mathcal{V}$ bounded by (8.7), whose properties 1-3 can be verified in time $\mathcal{O}(n)$. Determining the roots of subtrees in $\mathscr{S}(i,A\cup\mathcal{V})$ takes time $\mathcal{O}(n\ell)$. □

Additionally storing, along with each entry $cgp(i,A)$, the partition $\mathcal{V}\in\mathscr{L}_G(X_i\setminus A,V_i\setminus A)$ minimizing the right hand side in (8.6), allows us to finally reconstruct a charge group partition that gives the optimal cost.

8.4 Experimental evaluation

We implemented the dynamic programming method for bounded treewidth in C++ using the LEMON graph library (http://lemon.cs.elte.hu). We used `libtw` (http://www.treewidth.com/) to obtain bounded treewidth decompositions of the input molecules. In our implementation we solve the dynamic programming recurrence (8.6) in a top-down fashion by employing memoization.

8.4.1 Hydration free energy of amino acid side chains

We tested the quality of charge group assignments by comparing the calculated free energies of solvation in water of a set of 14 charge-neutral amino acid side chain analogs to experimental values, which are denoted by $\Delta G_{\mathrm{hyd,exp}}$ [87, 155]. For each analog, we used the GROMOS 53A6 covalent and van der Waals parameters [155] and partial atomic charges symmetrized by the ATB [144]. A united-atom representation is used for aliphatic carbon groups. For comparison, we also include the manually parametrized solution that the GROMOS 53A6 force field provides [155]. The topologies are derived from the amino acid structures by truncating at the C_α–C_β bond. For simplicity, we refer to these analogs by their parent amino acid.

Using the GROMACS 4.5.1 package [25], we computed the free energy of hydration $\Delta G_{\mathrm{hyd,calc}}$ using the thermodynamic integration method [29]. A series of simulations were performed at a constant pressure of $p=1$ bar and a constant temperature

Table 8.1: Comparison of hydration free energies ΔG_{hyd} of amino acid (AA) analogs. All free energy values are given in kJ/mol. When two values separated by a semicolon are given, two experimental values were found. The absolute free energy differences between simulation outcomes and the experimental values are given in parentheses. The average values of these differences are given in the bottom line. "ffG53A6" denotes results using the default GROMOS force field parameters for the analog, "ATB" denotes those using the ATB charge group assignment, "$k = 5$" denotes those using our method. We performed a two-tailed paired Student's t-test between the distributions given in column 6 (ATB) and column 8 ($k = 5$) resulting in a p-value of 0.2867. The difference in hydration free energy differences is thus not statistically significant.

AA analog	$\Delta G_{\text{hyd,exp}}$	$\Delta G_{\text{hyd,calc}}$					
		ffG53A6		ATB		$k = 5$	
Asn	-40.6	-42.7	(2.1)	-40.5	(0.1)	-47.0	(6.4)
Asp	-28.0	-30.1	(2.1)	-29.1	(1.1)	-28.6	(0.6)
Cys	-5.2	-4.9	(0.3)	-7.0	(1.8)	-7.1	(1.9)
Gln	-39.4	-40.4	(1.0)	-35.9	(3.5)	-35.9	(3.5)
Glu	-27.0	-27.0	(0.0)	-28.2	(1.2)	-32.1	(5.1)
His	-42.9	-44.8	(1.9)	-43.7	(0.8)	-40.9	(2.0)
Ile	8.7; 8.8	9.1	(0.3)	6.3	(2.5)	6.7	(2.1)
Leu	9.4; 9.7	10.8	(1.2)	7.4	(2.2)	7.1	(2.5)
Lys	-18.3	-18.1	(0.2)	-7.2	(11.1)	-7.2	(11.1)
Met	-6.2	-7.4	(1.2)	2.5	(8.7)	2.6	(8.8)
Phe	-3.1	-1.3	(1.8)	1.8	(4.9)	0.6	(3.7)
Trp	-24.7	-25.9	(1.2)	-20.9	(3.8)	-19.7	(5.0)
Tyr	-26.6	-26.9	(0.3)	-30.1	(3.5)	-39.5	(12.9)
Val	8.2	8.5	(0.3)	8.0	(0.2)	8.0	(0.2)
average			(1.1)		(3.2)		(4.7)

$T = 298.15$ K. The free energy was calculated for the process $A \rightarrow B$ which involved switching off all non-bonded interactions of the solute in water and in the gas phase. The hydration free energy is calculated as $\Delta G_{\text{hyd,calc}} = \Delta G_{\text{AB,solution}} - \Delta G_{\text{AB,gas}}$ [209]. The simulations were performed in cubic periodic boxes of length $L \approx 3$ nm. Depending on the analog, the solvated system contained approximately 900 SPC [209] water molecules.

As described in the introduction, neutral charge groups lead to more accurate simulation results. In our problem definition we aim to identify a charge group assignment where the constituent charge groups have small residual error, which is the absolute difference between the sum of the formal charges and the sum of the partial charges of the atoms in the charge group. To ensure neutral charge groups where possible, we adjust the partial charges slightly by redistributing the residual error of every charge group over its atoms.

The results are presented in Table 8.1 and Figure 8.3. The GROMOS 53A6 simulation results (ffG53A6 in Table 8.1) for the studied analogs show good agreement with experiment, which is not surprising as the force field has been parametrized to reproduce the hydration free energy [155]. Using the ATB charge group assignment solution (ATB in Table 8.1) leads to slightly larger deviations from experiment, but

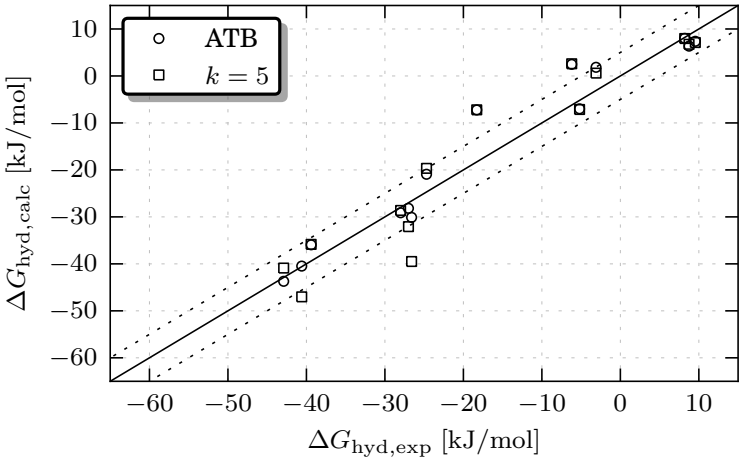


Figure 8.3: Calculated ΔG_{hyd} values versus experimental ones, showing the effect of the charge group assignment on the simulated hydration free energy. The labels in the legend are the same as in Table 8.1. The solid line represents perfect agreement with experiment, dotted lines indicate the ±5 kJ/mol approximate experimental error.

the average deviation is also within the experimental error of approximately 5 kJ/mol [144]. Although the current method leads to values close to those obtained experimentally, they deviate slightly more from experiment than the ATB values.

8.4.2 Adenosine Tri-Phosphate (ATP)

Although showing good performance on the amino acid side chains, the ATB method may lead to unacceptably large charge groups, in particular for large highly-charged molecules. An example is the cofactor Adenosine-5'-triphosphate (ATP), for which the ATB combined all phosphate groups and part of the ribose and nucleotide ring systems into a single charge group, see Figure 8.4(c). In Figure 8.4(b), the GROMOS 53A6 charge group assignment is given. For comparison, our solution is presented in Figure 8.4(a), and shows that the phosphate groups have been sorted in separate charge groups, in agreement with the 53A6 assignment and in line with chemical intuition where one expects functional group such as phosphate, amino and hydroxyl moieties to form separate charge groups.

8.5 Discussion

In this work we have formally introduced the charge group partitioning problem which arises in the development of atomic force fields, and more generally, in the identification of functional groups in molecules. The problem is to assign atoms to charge groups of size at most k and such that for every charge group the sum of its

partial charges is close to the sum of its formal charges. We showed NP-hardness for $k \geq 4$ and proposed and implemented an exact algorithm capable of solving practical problem instances to provable optimality. With this combination of rigorous definition and exact solution approach, we have made a first step towards formalizing and quantifying some of the aspects that make up "chemical intuition".

Algorithmically, we showed that the case $k = 2$ is solvable in polynomial time. In addition, we have presented a polynomial-time algorithm for bounded charge group size in cases where the molecular graph is a tree. Based on the observation that molecular graphs have bounded treewidth in practice and exploiting further properties such as outerplanarity and bounded degree, we developed a practical dynamic programming algorithm, which is based on a tree decomposition of the graph corresponding to the chemical structure of interest. An interesting open question is to settle the complexity status for the case $k = 3$.

Since our method relies on point charges obtained from quantum mechanical calculations, the quality of charge group assignments and subsequently of simulation outcomes depends on the accuracy of these calculations. However, our experiments have shown that taking into account charge group size and neutrality already gives good results, especially for large highly-charged molecules such as ATP, where other methods fail to produce meaningful solutions. Still, the greedy partitioning algorithm built into the ATB performs better on the set of smaller amino acid side chain molecules, which is due to the fact that this method exploits additional chemical knowledge. It is thus able, for instance, to deal with a symmetric molecule such as the Tyrosine side chain, where the charge group assignment of our new method resulted in a large deviation because we do not consider symmetry in our problem definition. We will therefore investigate how to incorporate symmetry into our approach, which is not trivial as symmetry may interfere with the optimal substructures required by the dynamic program. In addition to symmetry we plan to integrate other aspects of chemical intuition. For example, we will investigate the effect of bounding the error per charge group. Additionally, we plan to integrate constraints that take spatial geometry into account rather than using the number of atoms as a measure for charge group size. We would like to stress that only through a proper problem definition together with a method capable of obtaining provably optimal solutions, one is able to make progress in answering the question how a good charge group partition should look like.

Acknowledgments. We thank SARA Computing and Networking Services (www.sara.nl) for their support in using the Lisa Compute Cluster. In addition, we are grateful to the referees for helpful comments. The research leading to these results has received support from the Tinbergen Institute as well as from the Innovative Medicines Initiative Joint Undertaking under grant agreement no. 115002 (eTOX), resources of which are composed of financial contribution from the European Union's Seventh Framework Programme (FP7/2007âĂŘ2013) and EFPIA companies' in kind contribution.

Author Disclosure Statement. No competing financial interests exist.

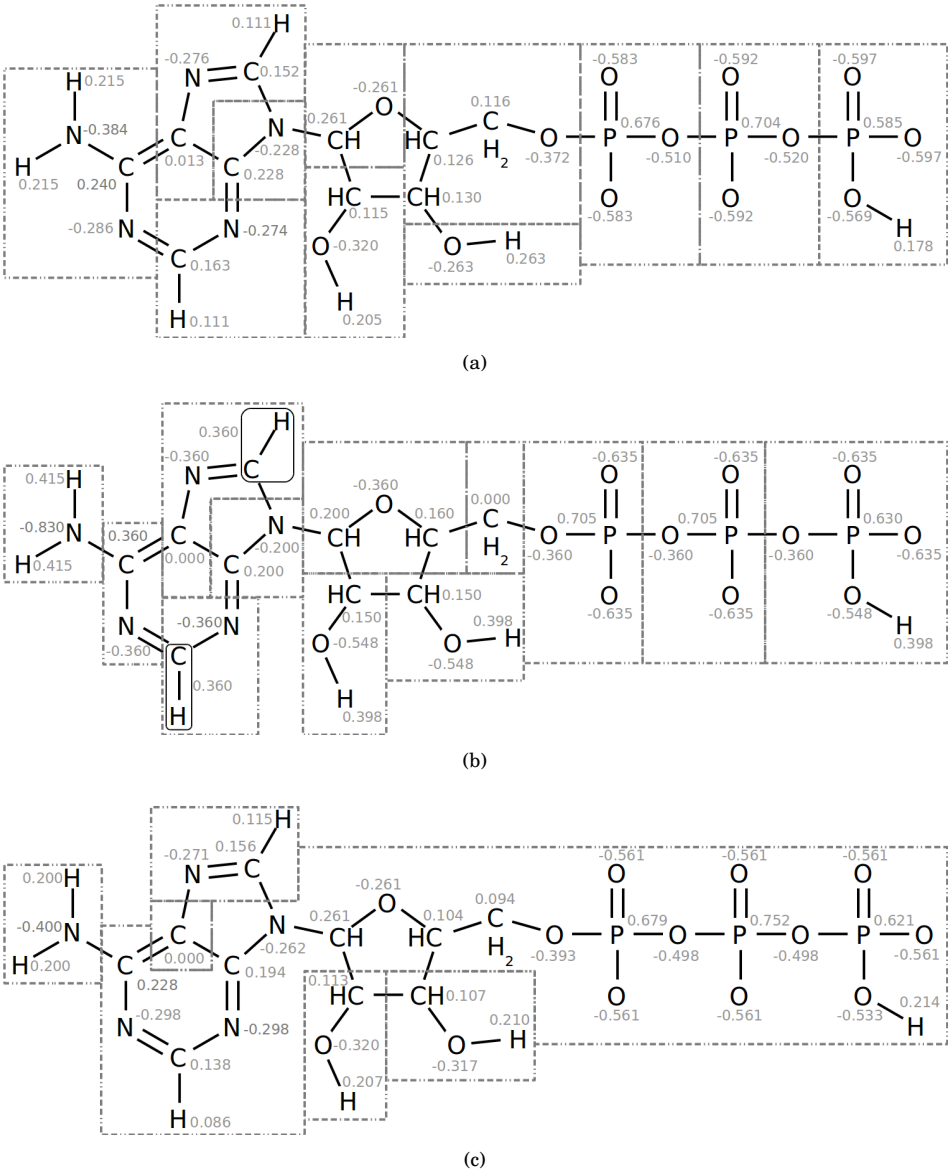


Figure 8.4: Charge group assignments for Adenosine Tri-Phosphate (ATP) at pH 5.0. The total molecular charge is -3. The partial charges are shown in grey. (a) Our optimal assignment according to Def. 8.1 obtained with $k = 5$, (b) GROMOS 53A6 assignment, and (c) assignment by the ATB. Note that the C–H segments indicated by the rounded boxes are considered as single atom types in the GROMOS assignment, whereas they comprise two atoms in the other assignments.

Part III

Breeding schedules

Chapter 9

Crossing schedule optimization

Adapted from:

S. Canzar[†] and M. El-Kebir[†]. A mathematical programming approach to marker-assisted gene pyramiding. In T. M. Przytycka and M.-F. Sagot, editors, *WABI*, volume 6833 of *Lecture Notes in Computer Science*, pages 26–38. Springer, 2011.

[†]joint first authorship

Abstract

In the crossing schedule optimization problem we are given an initial set of parental genotypes and a desired genotype, the ideotype. The task is to schedule crossings of individuals such that the number of generations, the number of crossings, and the required populations size are minimized. We present for the first time a mathematical model for the general problem variant and show that the problem is NP-hard and even hard to approximate. On the positive side, we present a mixed integer programming formulation that exploits the intrinsic combinatorial structure of the problem. We are able to solve a real-world instance to provable optimality in less than 2 seconds, which was not possible with earlier methods.

9.1 Introduction

Plant breeding is the practice of creating improved varieties of cultivated crops with for instance a higher yield, better appearance or enhanced disease resistance [38]. Up to recently, selection of favorable traits has been solely on the basis of observable *phenotype* [58]. With the availability of *genetic maps*, containing the exact locations on the genome of genetic markers associated with desirable traits, selection at the *genotypic* level has become possible [151]. This knowledge allows to design a schedule of crossings of individuals resulting ultimately in an individual with all alleles corresponding to desired favorable traits present. In the plant breeding literature this process is called *marker-assisted gene-pyramiding* and the resulting plan a *gene-pyramiding scheme* or a *crossing schedule* [50, 176, 222]. In this work we consider a

mathematical programming approach to the problem that asks to identify given (1) a genetic map, (2) an initial set of parental genotypes and (3) the desired genotype—the so called *ideotype*—a crossing schedule that results most cost-efficiently in the ideotype with respect to the following three criteria. Firstly, it takes time for the progeny to mature such that a next crossing can be performed. So the *number of generations* is a measure on the time it takes to execute the crossing schedule. Secondly, every crossing between two individual plants requires an effort from the breeder, e.g. plants have to be treated such that they flower at the same time. So typically the *number of crossings* is also to be minimized. Thirdly, in order to obtain the genotypes required by the schedule, for every crossing a specific number of offspring need to be generated among which the desired genotype is expected to be present. Simply speaking, the more difficult it is to obtain the desired genotype out of its parental genotypes, the larger the required number of offspring will be. Since every individual in the offspring has to be screened for having the desired genotype, the *total population size* is also to be minimized.

Related work. Most work on gene pyramiding lacks a formal framework; instead only an overview of guidelines and rules of thumb is given [109, 110, 222]. A notable exception, however, is the work by Servin et al. [176] who were the first to introduce a special case of the problem considered in this paper in a formal way. The authors show how to make use of the genetic map in determining the population sizes needed for all crossings. Contrary to our formulation, they allow a genotype to only participate in one crossing. In addition, very restrictive assumptions about the genotypes of the initial parents were made. These restrictions allowed the authors to exhaustively enumerate all crossing schedules and compare them in terms of population size needed. By introducing a heuristic, which partially alleviates the restriction on reuse of genotypes, the authors could compute smaller population sizes for the instances considered. Later papers by Ishii and Yonezawa [109, 110] assume that target genes are always unlinked, which imposes a lower bound on the genetic distance of pairs of target genes. Similar to our work, in [109, 110] the number of generations, number of crossings and the total population size are identified as important attributes. An experimental evaluation is performed on manually obtained crossing schedules having different topologies for a fixed number of parents.

Our contribution. In this work we lift the restrictions imposed by Servin et al. and consider a more general variant of the problem where genotypes are allowed to be re-used and no assumption about the initial parental genotypes is made. For the first time we formulate a mathematical model of the general problem. We show NP-hardness using an approximation-factor preserving reduction from which an inapproximability result follows. We introduce a mixed integer linear program (MIP) formulation which exploits various aspects of the inherent combinatorial structure of the problem and which approximates the non-linear objective by a piecewise linear curve. Finally, we show that our approach is capable of solving real-world instances to provable optimality within a precise mathematical model, which was not possible with earlier methods.

The rest of the paper is organized as follows. We start by formally defining the problem. In Section 9.3 we show hardness of the problem. In Section 9.4 we introduce our method and state a MIP formulation for the problem. A thorough experimental evaluation of our algorithm on a real-word instance and on randomly generated instances is presented in Section 9.5. We conclude with a discussion on our results in Section 9.6.

9.2 Problem definition

A *genotype* C is a $2 \times m$ matrix whose elements are called *alleles*. The two rows, $C_{1,\cdot}$ and $C_{2,\cdot}$, are called the lower and upper *chromosome*, respectively. Each column in C corresponds to a *locus*. So at a locus p two alleles are present, which we denote by $c_{1,p}$ and $c_{2,p}$. A locus is said to be *homozygous* if its two alleles are identical, otherwise it is *heterozygous*. Likewise, a genotype is homozygous if all its loci are homozygous, otherwise the genotype is said to be heterozygous. The desired genotype is called the *ideotype*, which we denote by C^*. In plant breeding often pure lines are desired, as they allow for instance for the production of F1 hybrids [38]. Therefore for the remainder of the paper we assume the ideotype to be homozygous. In this case, actual alleles can be classified as being present in the ideotype or not. Hence, the alleles in any genotype C are binary.

We represent a *crossing schedule* as a *connected directed acyclic graph* (DAG) whose nodes are labeled by genotypes. Specifically, the source nodes correspond to the initial parental genotypes. A non-source node, which we refer to as an *inner node*, corresponds to a crossing. The single target node is labeled by the ideotype. The arcs are directed towards the ideotype and relate a parent with its child. Since a genotype is obtained from two parents, the in-degree of an inner node is exactly 2. The two parents of a node need not be distinct. We say that a genotype is obtained via *selfing* if its two parents are identical. From the topology of a crossing schedule the number of generations and the number of crossings can be inferred. The number of generations is the length of the longest path from a source node to the target node. On the other hand, the number of crossings corresponds to the number of inner nodes. In Figure 9.1 an example crossing schedule is given.

The third attribute of a crossing schedule, the *total population size*, is the sum of the population sizes implied by the crossings represented by inner nodes. Let C be the genotype of an inner node and let D and E be the genotypes of the two parents of C. Later, we will show what the probability $\Pr[D,E \to C]$ of obtaining C out of D and E is. For now we denote this probability with ρ. The population size $N(\rho,\gamma)$ corresponding to ρ is the number of offspring one needs to generate in order to find with a given *probability of success* γ an individual with genotype C among the offspring. Since ρ is the probability of success in a Bernoulli trial, the probability that none of the $N(\rho,\gamma)$ offspring have genotype C is $(1-\rho)^{N(\rho,\gamma)} = 1-\gamma$. Therefore we have that

$$N(\rho,\gamma) = \frac{\log(1-\gamma)}{\log(1-\rho)}. \tag{9.1}$$

As also remarked in [176], it is sensible to have an upper bound on every population size in the schedule, as depending on the plant species only a limited number

of offspring can be generated. For that purpose we define $N_{\max}$ to be the *maximal population size* to which every crossing in a crossing schedule has to adhere.

In diploid organisms, the genotype of a zygote is obtained by the fusion of two haploid gametes originating from one parent each. So one of the chromosomes of the resulting genotype C, say $C_{1,\cdot}$, corresponds to a gamete given rise to by D and the other chromosome corresponds to a gamete produced by E. A gamete is the result of a biological process called *meiosis* where in pairs of homologous chromosomes crossover events may occur. In our setting, this means that an allele $c_{1,p}$ corresponds to either $d_{1,p}$ or $d_{2,p}$ (where $1 \leq p \leq m$). In case a pair of alleles at loci p and q of $C_{1,\cdot}$ do not correspond to the same chromosome of D, we say that a *crossover* has occurred between loci p and q (see Figure 9.1). From the genetic map, the probability of having a crossover between any pair of loci can be inferred using for instance Haldane's mapping function [94]. Let R be a $m \times m$ matrix containing all crossover probabilities. Due to the nature of meiosis, we have that $r_{p,q} \leq 0.5$ for $1 \leq p < q \leq m$. Let $s = (\nu(1), \ldots, \nu(k))$ be an ordered sequence of heterozygous loci in D. The probability of obtaining $C_{1,\cdot}$ out of D, i.e. $\Pr[D \to C_{1,\cdot}]$, is then as follows [176]. If there is an allele in $C_{1,\cdot}$ that does not occur in D at the same locus then $\Pr[D \to C_{1,\cdot}] = 0$. Otherwise, if s is empty then $\Pr[D \to C_{1,\cdot}] = 1$. Otherwise

$$\Pr[D \to C_{1,\cdot}] = \frac{1}{2} \prod_{i=1}^{k-1} \begin{cases} r_{\nu(i),\nu(i+1)} & \text{if } c_{1,\nu(i)} = d_{1,\nu(i)} \wedge c_{1,\nu(i+1)} = d_{2,\nu(i+1)} \\ & \text{or } c_{1,\nu(i)} = d_{2,\nu(i)} \wedge c_{1,\nu(i+1)} = d_{1,\nu(i+1)} \\ 1 - r_{\nu(i),\nu(i+1)} & \text{otherwise.} \end{cases} \tag{9.2}$$

We can now compute $\Pr[D, E \to C]$ using the following lemma.

Lemma 9.1 *The probability of obtaining C out of genotypes D and E is*

$$\Pr[D, E \to C] = \begin{cases} \Pr[D \to C_{1,\cdot}] \cdot \Pr[E \to C_{2,\cdot}] & \text{if } C_{1,\cdot} = C_{2,\cdot} \\ \Pr[D \to C_{1,\cdot}] \cdot \Pr[E \to C_{2,\cdot}] & \\ \quad + \Pr[E \to C_{1,\cdot}] \cdot \Pr[D \to C_{2,\cdot}] & \text{if } C_{1,\cdot} \neq C_{2,\cdot} \end{cases} \tag{9.3}$$

Proof It holds that either $C_{1,\cdot}$ is obtained from D and $C_{2,\cdot}$ is obtained from E, or vice versa. Thus, we have

$$\Pr[D, E \to C] = \Pr[((D \to C_{1,\cdot}) \cap (E \to C_{2,\cdot})) \cup ((E \to C_{1,\cdot}) \cap (D \to C_{2,\cdot}))].$$

Due to independence, it holds that

$$\Pr[D \to C_{1,\cdot} \cap E \to C_{2,\cdot}] = \Pr[D \to C_{1,\cdot}] \cdot \Pr[E \to C_{2,\cdot}],$$

and

$$\Pr[D \to C_{2,\cdot} \cap E \to C_{1,\cdot}] = \Pr[D \to C_{2,\cdot}] \cdot \Pr[E \to C_{1,\cdot}].$$

Recall that

$$\Pr[A \cup B] = \Pr[A] + \Pr[B] - \Pr[A \cap B].$$

Let $A = (D \to C_{1,\cdot}) \cap (E \to C_{2,\cdot})$ and $B = (E \to C_{1,\cdot}) \cap (D \to C_{2,\cdot})$. If $C_{1,\cdot} \neq C_{2,\cdot}$ then $\Pr[A \cap B]$ is 0. Otherwise, we have that $\Pr[A] = \Pr[B] = \Pr[A \cap B]$. □

A common way to deal with multiple objectives is to consider a convex combination of the objective criteria involved [189]. Given a crossing schedule, let crs, gen and pop denote the number of crossings, number of generations and the total population size, respectively. For $\lambda_{\text{crs}}, \lambda_{\text{gen}}, \lambda_{\text{pop}} \geq 0$ and $\lambda_{\text{crs}} + \lambda_{\text{gen}} + \lambda_{\text{pop}} = 1$, the *cost* of that crossing schedule is given by the convex combination $\lambda_{\text{crs}} \cdot \text{crs} + \lambda_{\text{gen}} \cdot \text{gen} + \lambda_{\text{pop}} \cdot \text{pop}$.

Problem 9.1 (CrossingSchedule) *Given $\mathscr{P} = \{C^1, \ldots, C^n\}$, the set of parental genotypes we start with, the homozygous ideotype $C^* \notin \mathscr{P}$, the recombination matrix R, the desired probability of success $\gamma \in (0,1)$, the maximal population size $N_{\max} \in \mathbb{N}$ allowed per crossing, and a vector λ of the cost efficients, problem* CrossingSchedule *asks for a crossing schedule of minimum cost.*

9.3 Complexity of the problem

We show that the problem is NP-hard even for the case where we are only minimizing the number of crossings. We do this by giving a polynomial-time reduction from the decision problem SetCover, which asks, given a universe U and a collection of subsets $\mathscr{S} \subseteq 2^U$, whether there exists a cover $\mathscr{C} \subseteq \mathscr{S}$ of cardinality at most k whose union is U [120]. Let $(U, \mathscr{S}, k)$, where $U = \{e_1, \ldots, e_n\}$ and $\mathscr{S} = \{S_1, \ldots, S_l\}$, be a problem instance of SetCover. The corresponding problem instance of CrossingSchedule is obtained by letting the loci correspond to the elements in U and the initial set of parents to the subsets in $\mathscr{S}$. The first chromosome of a parent C^i has a 1 at locus p if $p \in S_i$. The second chromosomes of all parental genotypes consists of only zeros. The ideotype has 1-alleles at every locus. In the cost function we only consider the number of crossings, i.e. $\lambda_{\text{crs}} = 1$ and $\lambda_{\text{gen}} = \lambda_{\text{pop}} = 0$. The reduction can be done in polynomial time (in fact in $\mathcal{O}(nl)$ time). In order to show that the reduction works, we have to show that the following holds.

Lemma 9.2 *There is a cover of cardinality at most k if and only if the cost of the optimum schedule is at most k.*

Proof ($\Rightarrow$) Let $\mathscr{C} \subseteq \mathscr{S}$ be a cover such that $|\mathscr{C}| \leq k$. Recall that every subset $S \in \mathscr{S}$ corresponds to one parent. So with $|\mathscr{C}| - 1$ crossings we can obtain an individual whose genotype contains at least one 1 at every locus. The ideotype can then be obtained by a selfing step. The corresponding schedule G contains thus $|\mathscr{C}|$ crossings. Therefore, the optimum schedule G^* contains at most $|\mathscr{C}| \leq k$ crossings.

($\Leftarrow$) Let G^* be the optimum crossing schedule, and let m be the number of crossings in G^*. We have that $m \leq k$. We claim that the final crossing in G^* is a selfing step. Assume for a contradiction that this is not the case. Let D and E be the two distinct genotypes whose crossing resulted in C^*. We either have $D \notin \mathscr{P}$ or $E \notin \mathscr{P}$, as if both C and D were to be in $\mathscr{P}$ then they would be equal (recall that the second chromosome of genotypes in $\mathscr{P}$ has only zeros). Now assume that $E \notin \mathscr{P}$. The genotype C^* can also be obtained by selfing D. Since $E \notin \mathscr{P}$, we require a crossing to obtain E from its parents. So by obtaining C^* via a selfing of D, the number of crossings is at least one less than originally the case. This contradicts our assumption that G^* is optimal. Hence, the final crossing in G^* is a selfing.

All other $m-1$ crossings involve two distinct individuals (i.e. no selfing); the argument for this is as follows. Suppose for a contradiction that there is a non-final selfing step involving an individual C in G^*. Because the selfing is non-final, the resulting individual C' participates in another crossing. Instead of using C' in this crossing, we could have used C and ended up with a schedule with fewer crossing than G^*. This would be a contradiction however.

So now we have $m-1$ crossings, each involving two distinct individuals. Let G' be the subgraph of G^* that contains these crossings. We have that G' is connected as G^* was connected. We now want to bound the number of parents in G'. There can be at most m parents in G', as the largest number of crossings is achieved when we have a binary tree with $m-1$ inner nodes rooted at the pre-final individual. Since parents correspond to subsets in $\mathscr{S}$ and since crossing them together resulted in the ideotype, we have that there is a cover of at most m subsets. As $m \leq k$, there is a cover of cardinality at most k. □

Note that the reduction preserves the approximation factor, as the number of crossings equals the number of subsets in the cover. There is an inapproximability result for SetCover: it cannot be approximated within $\mathscr{O}(\log n)$ unless P = NP [165]. Because of the approximation factor preserving reduction, this also holds for CrossingSchedule. In sum, the following theorems hold.

Theorem 9.3 CrossingSchedule *is NP-hard.*

Theorem 9.4 *Approximating* CrossingSchedule *within* $\mathscr{O}(\log n)$ *is NP-hard.*

9.4 Method

After exploring the combinatorial structure of the problem, we present an algorithm in which iteratively an MIP is solved. Details on the MIP formulation are given in Section 9.4.1.

Since we are considering homozygous ideotypes, we can assume without loss of generality that C^* has only 1-alleles and derive a lower bound based on the minimum set cover as follows. The universe corresponds to the loci, i.e. $U = \{1,\ldots,m\}$, and the subsets $\mathscr{S} = \{S_1,\ldots,S_n\}$ correspond to $\mathscr{P} = \{C^1,\ldots,C^n\}$. We define $p \in S_i$ if either $c^i_{1,p} = 1$ or $c^i_{2,p} = 1$ where $1 \leq i \leq n$ and $1 \leq p \leq m$. The following lemma now follows.

Lemma 9.5 *The cardinality of a minimum set cover is a lower bound on the number of crossings of any feasible crossing schedule.*

Proof Let $\mathscr{C} = \{S_1,\ldots,S_k\}$ be a minimal cover with cardinality k. Assume without loss of generality that C^* contains only 1-alleles. The subsets $\{S_1,\ldots,S_k\}$ that comprise $\mathscr{C}$ each correspond to an initial parental genotype, say that they correspond to $\{C^1,\ldots,C^k\}$. To obtain a crossing schedule with exactly k source nodes, we need at least $k-1$ inner nodes/crossings. Since $\mathscr{C}$ is a minimum cover and since the ideotype is not an initial parental genotype, we need at least one more crossing for obtaining C^*. □

Computing the minimum set cover is NP-hard. However, since in our experiments the number of loci and parents are relatively small, we are able to obtain the lower bound by solving a corresponding ILP [217] in a fraction of a second.

We can derive a lower bound for the total population size when considering consecutive pairs of loci for which there is no genotype in $\mathscr{P}$ containing 1-alleles at the same chromosome at both loci. Let $(p, p+1)$ be such a pair of loci (where $1 \leq p < m$). For the ideotype to be obtained, there must be a genotype C in the crossing schedule that contains 1-alleles at loci p and $p+1$ on the same chromosome, while this condition does not hold for its parents D and E. Clearly, $\Pr[D, E \to C] \leq r_{p,p+1}$. Plugging this into (9.1) yields a lower bound on the population size for that crossing and hence for the total population size. A tighter lower bound can be obtained when considering *all* pairs of consecutive loci for which there are no genotypes in $\mathscr{P}$ containing 1-alleles at the respective loci on the same chromosome. Let $\mathscr{L}$ be the set of such pairs of loci. One can easily verify that

$$LB_{\text{pop}} := \sum_{(p,p+1) \in \mathscr{L}} N(r_{p,p+1}, \gamma) \tag{9.4}$$

is a lower bound on the total population size, as (i) for every $(p, p+1) \in \mathscr{L}$, $r_{p,p+1}$ is an upper bound on the probability of joining the two 1-alleles at loci p and $p+1$, and (ii) the following lemma holds:

Lemma 9.6 *Let* $p, q \in [0, 0.5]$. *Then*

$$N(p, \gamma) + N(q, \gamma) \leq N(pq, \gamma)$$

Proof From (9.1) we have

$$\frac{\log(1-\gamma)}{\log(1-p)} + \frac{\log(1-\gamma)}{\log(1-q)} \leq \frac{\log(1-\gamma)}{\log(1-pq)}.$$

Since $\log(1-\gamma) < 0$, multiplying by $1/\log(1-\gamma)$ yields

$$\frac{1}{\log(1-p)} + \frac{1}{\log(1-q)} \geq \frac{1}{\log(1-pq)}.$$

Without loss of generality we can assume that $p \geq q$ and therefore it suffices to show that

$$\frac{2}{\log(1-p)} \geq \frac{1}{\log(1-pq)},$$

which amounts to

$$p(pq^2 - 2q + 1) \geq 0.$$

In case $p = 0$ the lemma follows, otherwise we can divide by p and use $p^3 \geq pq^2$ yielding

$$p^3 + 1 \geq 2q$$

which holds since $q \leq 0.5$ and $p \geq 0$. □

Using (9.3) one can show that there is an optimal crossing schedule where all homozygous genotypes are obtained via selfings.

Lemma 9.7 *There is an optimal schedule in which the (inner) homozygous genotypes are obtained via selfings.*

Proof We prove this by contradiction. Let C be a homozygous genotype obtained optimally but not via a selfing. Let D and E be the two parents of C. Since C is homozygous, i.e. $C_{1,\cdot} = C_{2,\cdot}$, we have that $\Pr[D \to C_{1,\cdot}] > 0$ and $\Pr[E \to C_{1,\cdot}] > 0$. In other words, both D and E can give rise to $C_{1,\cdot}$.

Let's look at what happens when we self either D or E. The number of generations and the number of crossings would not change for the worse. We claim that if we self the genotype that has the highest probability of giving rise to $C_{1,\cdot}$, we get that the probability of obtaining C is at most the probability of obtaining C via a crossing of D and E. Since C is homozygous, we have by Lemma 9.1 that

$$\Pr[D,E \to C] = \Pr[D \to C_{1,\cdot}] \cdot \Pr[E \to C_{1,\cdot}].$$

We assume without loss of generality that $\Pr[E \to C_{1,\cdot}] \leq \Pr[D \to C_{1,\cdot}]$. Therefore we have

$$\Pr[D,E \to C] \leq \Pr[D \to C_{1,\cdot}]^2.$$

So the population size needed for obtaining C by selfing D is at most the population size that we needed to obtain C out of D and E. □

Finally, parental genotypes that contain a 1-allele at a locus at which all other parental genotypes contain all 0 have to be used by any feasible schedule. To reduce the search space explored by the MIP solver we fix these *compulsory* parental genotypes to be contained in any solution.

We present a MIP formulation for the problem variant where the number of crossings and the number of generations is fixed to F, respectively G. The reason for this is to be able to introduce cuts that ensure monotonically better solutions. In order to solve a problem instance, we iteratively consider combinations of (F,G) starting from $F = LB_{\text{crs}}$ and $G = 1 + \lceil \log_2 F \rceil$. In addition we enforce that the objective value of any feasible solution must be better than the currently best one. We do this by computing an upper bound UB_{pop} on the total population size, based on the best objective value found so far and the current values of (F,G) (see Algorithm 5, line 4). If at some point, say (F',G'), $LB_{\text{pop}} \geq UB_{\text{pop}}$ then we know that none of the combinations of $F'' \geq F'$, $G'' \geq G'$ will lead to a better solution. Therefore if $G = 1 + \lceil \log_2 F \rceil$ and $LB_{\text{pop}} \geq UB_{\text{pop}}$, we have found the optimal solution (see Algorithm 5, line 7). To guarantee termination for the case where $\lambda_{\text{crs}} = \lambda_{\text{gen}} = 0$, we stop incrementing F as soon as it reaches a pre-specified parameter UB_{crs}. Similarly, UB_{gen} is a pre-specified parameter bounding G. In Algorithm 5 the pseudo code is given.

9.4.1 MIP formulation

Given an instance to CrossingSchedule with initial parental genotypes $\mathscr{P} = \{C^1,\ldots,C^n\}$, a feasible solution with G generations and F crossings can be characterized by the following five conditions: (i) The topology of the schedule is represented by a directed acyclic graph with n source nodes $s_1,\ldots,s_n$, one target node t, and $F-1$ additional nodes, where every non-source node has in-degree two. Parallel arcs are

Algorithm 5: OPTCROSSINGSCHEDULE($UB_{\text{crs}}, UB_{\text{gen}}$)

Input: UB_{crs} and UB_{gen} are the maximum number of crossings and generations considered.

```
OPT ← ∞
for F ← LB_crs to UB_crs do
    for G ← 1 + ⌈log_2 F⌉ to min(F, UB_gen) do
        UB_pop ← (1/λ_pop)(OPT − F·λ_crs − G·λ_gen)
        if LB_pop < UB_pop then OPT ← min(OPT, MIP(F, G, UB_pop))
        else UB_gen ← G − 1
    if UB_gen ≤ 1 + ⌈log_2 F⌉ then return OPT
return OPT
```

allowed and represent selfings. (ii) The longest path from a source node to the target node has length G. (iii) The alleles of each non-source node are derived from either the upper or lower chromosome of the node's respective predecessors. (iv) The genotype of a source node s_i is C^i, the genotype of t is C^*. (v) The probability of obtaining the genotype of an inner node v is at least $1-(1-\gamma)^{\frac{1}{N_{\max}}}$ such that its corresponding population size is at most $N_{\max}$.

In the following we show how these conditions can be formulated as linear constraints. Throughout our formulation, we let $L := F + n$ be the total number of nodes. Indices $1 \le i, j \le L$ correspond to genotypes, loci are indexed by $1 \le p, q \le m$ and chromosomes are referred to by $1 \le k, l \le 2L$. For brevity's sake, in the remainder of the paper we will omit the linearization of products of binary variables. Unless otherwise stated, we applied a standard transformation [36]: for a product $x \cdot y$ of binary variables x and y we introduce a new binary variable z and require $z \le x$, $z \le y$, and $z \ge x + y - 1$. Similarly, we omit the details of the implementation of absolute differences of binary variables.

Feasibility constraints. The first set of constraints encodes the structure of the underlying directed acyclic graph $D = (V, A)$. We assume a numbering of the vertices according to their topological order. In particular, arcs always go from vertices $j < i$ to a vertex i for $i, j \in V$. Based on the node numbering, the lower and upper chromosomes of a node $i \in V$ are respectively $2i - 1$ and $2i$. For convenience we introduce a mapping function $\delta(k)$ that returns the node a chromosome k corresponds to. Then binary variables $x_{k,i} \in \{0, 1\}$, $2n < k \le 2L$, $1 \le i < \delta(k)$, denote whether chromosome k originates from genotype i, that is, they indicate an arc $(i, \delta(k))$. Since a chromosome originates from exactly one genotype, we have

$$\sum_{j=1}^{\delta(k)-1} x_{k,j} = 1 \qquad 2n < k \le 2L \tag{9.5}$$

We capture the second condition by fixing a path of length G using the x variables and by restricting the depth of all remaining nodes, represented by additional integer variables, to be at most $G - 1$. Implementing this condition requires the assumed

ordering on vertices to be extended. For the sake of a clear discussion, we omit the details.

To model the third condition, we introduce binary variables $a_{k,p}$, $1 \le k \le 2L$, $1 \le p \le m$, which indicate the allele at locus p of chromosome k. In addition to knowing from which genotype a chromosome originates, we also need to know from which of the two chromosomes of that parental genotype an allele comes. Therefore we define binary variable $y_{k,p}$, $2n < k \le 2L$, $1 \le p \le m$, to be 1 if the allele at locus p of chromosome k comes from the lower chromosome of its originating genotype; conversely $y_{k,p}$ is 0 if the allele originates from the upper chromosome. Now we can relate alleles to originating chromosomes. We do this by introducing binary variables $g_{k,p,l}$, for $2n < k \le 2L$, $1 \le p \le m$, and $1 \le l < 2\delta(k) - 1$. We define $g_{k,p,l} = 1$ if and only if the allele at locus p of chromosome k originates from chromosome l and has value 1. This is established through constraints

$$g_{k,p,2i} - a_{2i,p} \cdot x_{k,i} \cdot (1 - y_{k,p}) = 0 \qquad 2n < k \le 2L, 1 \le p \le m, i < \delta(k) \tag{9.6}$$

$$g_{k,p,2i-1} - a_{2i-1,p} \cdot x_{k,i} \cdot y_{k,p} = 0 \qquad 2n < k \le 2L, 1 \le p \le m, i < \delta(k) \tag{9.7}$$

Finally, an allele is 1 if and only if it originates from exactly one 1-allele:

$$a_{k,p} - \sum_{i=1}^{\delta(k)-1} (g_{k,p,2i-1} + g_{k,p,2i}) = 0 \qquad 2n < k \le 2L, 1 \le p \le m \tag{9.8}$$

The fourth property can be ensured by simply forcing the variables representing the alleles of the parental genotypes, i.e. the source nodes, and the alleles of the desired ideotype, that is, the target node, to the actual value of the respective allele. Thus for the parental genotypes we have $a_{2i-1,p} = c^i_{1,p}$ and $a_{2i,p} = c^i_{2,p}$ for $1 \le i \le n, 1 \le p \le m$ and for the ideotype $a_{2L-1,p} = a_{2L,p} = c^*_{1,p}$ for $1 \le p \le m$. The last property is enforced implicitly by the objective function, which is described in the following.

Objective function. The probability of a given genotype i giving rise to a specific chromosome k determines the required population size (see (9.1)). This probability in turn depends on the exact set of crossovers necessary to generate chromosome k and on the sequence s of heterozygous loci (see (9.2)). We define binary variable $\tilde{a}_{i,p} = 1$ if and only if locus p of genotype i is heterozygous: $\tilde{a}_{i,p} = |a_{2i-1,p} - a_{2i,p}|$ for $1 \le i \le L, 1 \le p \le m$. Now a genotype i is heterozygous, indicated by $h_i = 1$, if at least one of its loci is heterozygous: $h_i \ge \tilde{a}_{i,p}$ for $1 \le i \le L, 1 \le p \le m$. It is ensured that $h_i = 0$ whenever $\tilde{a}_{i,p} = 0$, $\forall 1 \le p \le m$, as $h_i = 1$ would increase the required population size.

Concerning the dependence on the set of crossovers, we let integer variables $z_{k,p,q}$ indicate whether the segment between locus p and q of chromosome k results from a crossover event:

$$z_{k,p,q} = \sum_{r=p+1}^{q} |y_{k,r} - y_{k,r-1}| \qquad 2n < k \le 2L, 2 \le p < q \le m \tag{9.9}$$

The distinction between the two different cases in (9.2) is based on crossover events between two successive heterozygous loci, i.e. $v(i)$ and $v(i+1)$. We capture the sequence s of heterozygous loci used in (9.2) by binary variables $b_{i,p,q}$, which indicate a

maximal block of homozygous loci between heterozygous loci p and q, $1 \le p < q \le m$, in genotype i, $1 \le i \le L$:

$$b_{i,p,q} = \tilde{a}_{i,p} \cdot \tilde{a}_{i,q} \cdot \prod_{r=p+1}^{q-1} (1 - \tilde{a}_{i,r}) \qquad 1 \le i \le L, 1 \le p < q \le m \tag{9.10}$$

We are now able to formulate the probability given in (9.2) in terms of the binary variables h, b and z. For that, let ξ_k^j denote the event of obtaining a chromosome k from a genotype j. Since $(1 - r_{p,q}) \ge r_{p,q}$ for all $1 \le p < q < m$, we may define $\Delta r_{p,q} := (1 - r_{p,q}) - r_{p,q}$. We can express $\Pr[\xi_k^j]$, assuming $x_{k,j} = 1$, as follows.

$$\begin{aligned}
\Pr[\xi_k^j] &= (1 - \frac{h_j}{2}) \prod_{p=1}^{m-1} \prod_{q=p+1}^{m} \Big((1 - b_{j,p,q}) \\
&\qquad + b_{j,p,q} \cdot (r_{p,q} \cdot z_{k,p,q} + (1 - r_{p,q}) \cdot (1 - z_{k,p,q})) \Big) \\
&= (1 - \frac{h_j}{2}) \prod_{p=1}^{m-1} \prod_{q=p+1}^{m} \left((1 - b_{j,p,q}) + b_{j,p,q} \cdot (1 - r_{p,q} - \Delta r_{p,q} z_{k,p,q}) \right)
\end{aligned}$$

Indeed, for a homozygous genotype j all elements in the upper product evaluate to 1 as desired. In the heterozygous case, every maximal homozygous block contributes $r_{p,q}$ if it contains at least one crossover (first case in (9.2)), and $(1 - r_{p,q})$ otherwise (second case in (9.2)). The corresponding log-probability of event ξ_k^j is as follows.

$$\begin{aligned}
\ln(\Pr[\xi_k^j]) = h_j \ln(\frac{1}{2}) &+ \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} b_{j,p,q} \cdot \ln(1 - r_{p,q}) \\
&+ \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} b_{j,p,q} \cdot z_{k,p,q} \cdot \ln(\frac{r_{p,q}}{1 - r_{p,q}})
\end{aligned}$$

If j_1 and j_2 are the two parental genotypes of chromosomes $2i-1$ and $2i$ forming genotype i, we compute in variable p_i the log probability of event $\xi_{2i-1}^{j_1} \cap \xi_{2i}^{j_2}$ as $\ln(\Pr[\xi_{2i-1}^{j_1}]) + \ln(\Pr[\xi_{2i}^{j_2}])$. For that we have to sum over all possible $1 \le j < i$ to identify j_1 and j_2:

$$p_i = \sum_{j=1}^{i-1} \sum_{k=2i-1}^{2i} x_{k,j} \cdot \ln(\Pr[\xi_k^j]) \tag{9.11}$$

$$= \sum_{j=1}^{i-1} \sum_{k=2i-1}^{2i} x_{k,j} \Big(h_j \ln(\frac{1}{2}) + \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} b_{j,p,q} \ln(1 - r_{p,q}) \tag{9.12}$$

$$+ \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} b_{j,p,q} \cdot z_{k,p,q} \ln(\frac{r_{p,q}}{1 - r_{p,q}}) \Big) \tag{9.13}$$

Notice that in case genotype i is heterozygous, its two chromosomes may swap their originating genotypes as accounted for in the second case of equation (9.3). We

model such an event $\xi_{2i-1}^{j_2} \cap \xi_{2i}^{j_1}$ with the shadow variables $\tilde{x}_{k,i} \in \{0,1\}$ and the following constraints.

$$\tilde{x}_{2i-1,j} \le x_{2i,j} \qquad n < i \le L, 1 \le j < i \tag{9.14}$$

$$\tilde{x}_{2i,j} \le x_{2i-1,j} \qquad n < i \le L, 1 \le j < i \tag{9.15}$$

$$\sum_{j=1}^{i-1} \tilde{x}_{2i-1,j} = \sum_{j=1}^{i-1} \tilde{x}_{2i,j} \qquad n < i \le L \tag{9.16}$$

$$\tilde{x}_{k,j} \le h_{\delta(k)} \qquad 2n < k \le 2L, 1 \le j < \delta(k) \tag{9.17}$$

In addition, we need to ensure that the swapped parental genotypes are actually able to give rise to the two chromosomes. We do this by introducing shadow variables $\tilde{y}_{k,p}$, $\tilde{g}_{k,p,l} \in \{0,1\}$ that have similar meaning as their non-shadow counterparts as expressed by the following constraints.

$$\tilde{g}_{k,p,2i} - a_{2i,p} \cdot \tilde{x}_{k,i} \cdot (1 - \tilde{y}_{k,p}) = 0 \qquad 2n < k \le 2L, 1 \le p \le m, 1 \le i < \delta(k) \tag{9.18}$$

$$\tilde{g}_{k,p,2i-1} - a_{2i-1,p} \cdot \tilde{x}_{k,i} \cdot \tilde{y}_{k,p} = 0 \qquad 2n < k \le 2L, 1 \le p \le m, 1 \le i < \delta(k) \tag{9.19}$$

$$\tilde{g}_{k,p,2i} + \tilde{g}_{k,p,2i-1} - a_{k,p} \cdot \tilde{x}_{k,i} = 0 \qquad 2n < k \le 2L, 1 \le p \le m, 1 \le i < \delta(k) \tag{9.20}$$

Now we can express the log-probability $\tilde{p}_i$ of observing $\xi_{2i-1}^{j_2} \cap \xi_{2i}^{j_1}$ as follows.

$$\begin{aligned} \tilde{p}_i = \sum_{j=1}^{i-1} \sum_{k=2i-1}^{2i} \tilde{x}_{k,j} \Big(h_j \ln(\frac{1}{2}) + \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} b_{j,p,q} \ln(1 - r_{p,q}) \\ + \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} b_{j,p,q} \cdot \tilde{z}_{k,p,q} \ln(\frac{r_{p,q}}{1 - r_{p,q}}) \Big) \end{aligned} \tag{9.21}$$

We develop an appropriate approximation of the nonlinear function $N(\rho, \gamma)$ defining the required population size so that integer linear programming techniques can be utilized. More precisely, we re-express the population size w.r.t. probability $p := \Pr[\, j_1, j_2 \to i \,]$ as:

$$N(p, \gamma) = \frac{\ln(1 - \gamma)}{\ln(1 - (\mathrm{e}^{p_i} + \mathrm{e}^{\tilde{p}_i}))}$$

We use ℓ segments to approximate the nonlinear function N by a piecewise-linear curve specified by the points $(d_j, N(d_j, \gamma))$ for $j = 1, \ldots, \ell + 1$. The idea of the *λ-method* [189] is to express any point $p \in [d_1, d_{\ell+1}]$ as a convex combination of two adjacent breakpoints d_j and d_{j+1}, where $p \in [d_j, d_{j+1}]$, and derive an approximation for N by weighing the function's values $N(d_j, \gamma)$ and $N(d_{j+1}, \gamma)$ accordingly. More precisely, we add for every i the following constraints (where $n < i \le L$):

$$\sum_{j=1}^{\ell+1} \lambda_j^i = 1 \tag{9.22}$$

$$\sum_{j=1}^{\ell+1} \lambda_j^i \cdot d_j = p_i \tag{9.23}$$

$$\lambda_j^i \ge 0 \qquad j = 1, \ldots, \ell + 1 \tag{9.24}$$

Note that the adjacency condition on the positive coefficients λ_j^i will always be enforced by the minimization of function N, which is *convex* in the interval $[-\infty, -0.23]$: The first and second derivatives of the separable population size function are

$$N'(x) = \frac{\ln(1-\gamma)e^x}{\ln^2(1-e^x)(1-e^x)}$$

$$N''(x) = \frac{\ln(1-\gamma)e^x(\ln(1-e^x)+2e^x)}{\ln^3(1-e^x)(1-e^x)^2}$$

Solving $N''(x) = 0$ yields

$$x = \log(W\left(-\frac{2}{e^2}+2\right)) - \log(2) \approx -0.227136$$

where $W(z)$ is the product log function. The log probabilities that occur are at most $\log(0.5) = -0.69$ or exactly $\log(1) = 0$.

Finally, we replace the populations size $N(e^{p_i}, \gamma)$ for each crossing i in the objective function by a convex combination of the respective breakpoint scores to derive $\lambda_{\text{pop}} \cdot \left(\sum_{i=n+1}^{L} \sum_{j=1}^{\ell+1} \lambda_j^i \cdot N(e^{d_j}, \gamma)\right) + \lambda_{\text{gen}} \cdot G + \lambda_{\text{crs}} \cdot F$.

Additional cuts. We consider three additional cuts. The first one is due to Lemma 9.7. The following constraints enforce that a homozygous genotype results via selfing: $|x_{2i-1,j} - x_{2i,j}| \leq h_j$ for $n < i \leq L, 1 \leq j < i$. In addition, the lower and upper bound on the population size correspond to $LB_{\text{pop}} \leq \sum_{i=n+1}^{L} \sum_{j=1}^{\ell+1} \lambda_j^i \cdot N(e^{d_j}, \gamma) \leq UB_{\text{pop}}$ for $n < i \leq L$. For the sake of simplicity we omit the additional constraints required to enforce compulsory parental genotypes to be contained in the solution.

To come back to condition five of our characterization of feasible solutions in the beginning of this section, we simply set $d_1 = \log(1-(1-\gamma)^{\frac{1}{N_{\max}}})$. Then any $p_i < d_1$ implying a population size larger than $N_{\max}$ cannot be expressed as a convex combination of break points d_j, $j = 1, \ldots, \ell+1$, and hence any feasible solution must satisfy the bound on the population size.

In total, our MIP formulation comprises $\mathcal{O}(L(Lm^2 + \ell))$ many variables and $\mathcal{O}(L^2 m)$ constraints.

9.5 Experimental results

We have implemented OptCrossingSchedule in C++ using CPLEX 12.2[1] (default settings) with Concert Technology. We ran the experiments on a compute cluster with Intel Quad Core 2.26 GHz processors with 24 GB of RAM, running 64 bit Linux. We applied a time limit of 10 hours. Computations exceeding this limit were aborted. As mentioned earlier, there exist no previous methods for the general problem formulation we are considering. However, our problem formulation subsumes the one given by Servin et al., therefore we consider the same instances as well. In addition,

[1] http://www.cplex.com

#loci	tree			PWC2			MIP		
	pop	crs	gen	pop	crs	gen	pop	crs	gen
4	374	5	5	359	7	5	350	5	5
5	551	6	6	516	8	6	482	9	8
6	770	7	7	691	9	6	624	9	7
7	1046	8	8	890	13	7	901	10	9
8	1394	9	9	1147	15	7	1329	10	10

Table 9.1: Results for the instances by Servin et al. First column are the results on the tree cases (as obtained by Servin et al's method and our MIP), the second column corresponds to PWC2 heuristic and the last column to our MIP for DAGs.

we study a real-world instance. We conclude by evaluating our method on automatically generated instances. Throughout this section, the term 'provably optimal solution' indicates that the objective value of any feasible solution with respect to the piecewise-linear approximation and the simplification of (9.3) is at most the objective value of the obtained solution.

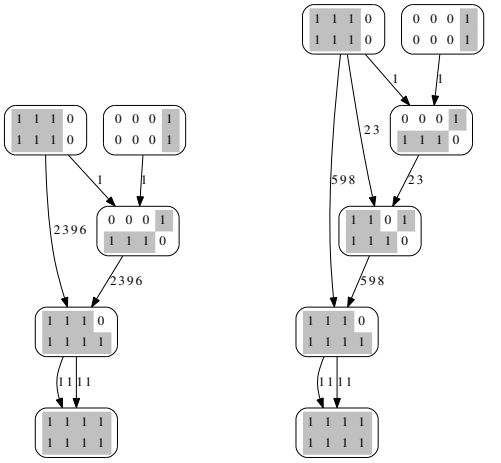
Instances by Servin et al. As opposed to our setting, in [176] a crossing schedule is required to be a tree. In addition, the number of initial parental genotypes $\mathscr{P} = \{C^0, C^1, \ldots, C^m\}$ is one more than the number of loci m. Parental genotypes are assumed to be homozygous. More specifically, C^0 consists of only 0-alleles, whereas for a genotype C^i, $1 \leq i \leq m$, the only 1-alleles are present at locus i. The ideotype is comprised entirely of 1-alleles and only the population size is considered, i.e. $\lambda_{\text{pop}} = 1, \lambda_{\text{gen}} = \lambda_{\text{crs}} = 0$. The desired probability of success is $\gamma = 0.999$ and the genetic distance between pairs of consecutive loci is 20 centimorgans (cM). By including constraints forcing a crossing schedule to be a tree (i.e. the out-degree of a node is forced to be 1), we obtained the same optimal results (see Table 9.1).

In their paper Servin et al. realize that better crossing schedules can be obtained when dropping the tree restriction. Rather than considering general DAGs, the authors consider a heuristic (PWC2) that transforms every enumerated tree into a DAG with smaller total population size. As opposed to the tree case, our method does not guarantee the solutions found in the DAG case of Servin's instances to be optimal. This is because the objective function does neither include the number of crossings nor the number of generations. In addition, we put a time limit of 10 hours in place. In Table 9.1 we can see that we obtain better solutions w.r.t. the population size for the instances up to six loci. Due to the time limit, the best *feasible* solutions found for the instances with 7 and 8 loci are worse than the ones computed by Servin et al. Since PWC2 solutions are also feasible to our general model, a higher time limit would result in solutions that are at least as good as Servin's solutions. We expect our approach to be less competitive with PWC2 on larger instances of this specific class. This comes at no surprise since PWC2 is specifically tailored toward these restricted instances.

Real-world instance. We consider a real-world case that deals with a disease in pepper called powdery mildew. This disease is caused by the fungus *Leveillula*

Taurica. In severe cases of the disease the infected pepper plant may lose a significant amount of its leaves, which in turn results in crop loss. The fungus is resistant to fungicides, so host-plant resistance is desired. There is a wild-type pepper line that is resistant to the fungus. For this wild-type, three dominant quantitative trait loci (QTLs), numbered 1,2 and 3, that explain the resistance have been identified [180]. In addition to resistance, we also look at pungency, which is a dominant monogenic trait whose locus we assign number 4. The pungency gene is closely linked with one of the resistance QTLs, say the one of locus 3, with a genetic distance of 0.01 cM, i.e. $r_{3,4} = 0.01$ [94]. The resistant line is pungent. On the other hand, the elite line used for production is sweet but susceptible to the disease. Both lines are pure lines, i.e. they are homozygous at all loci. The goal now is to come up with a crossing schedule that results in a homozygous individual that is both resistant and sweet. We do this by using 1-alleles to indicate desired alleles. Therefore the parent set is $\mathscr{P} = \left\{\begin{pmatrix}1&1&1&0\\1&1&1&0\end{pmatrix}, \begin{pmatrix}0&0&0&1\\0&0&0&1\end{pmatrix}\right\}$, and the ideotype is $C^* = \begin{pmatrix}1&1&1&1\\1&1&1&1\end{pmatrix}$. Unlinked loci by definition have a crossover probability of 1/2. So except for $r_{3,4}$, $r_{p,q} = 1/2$ for all $1 \le p < q \le 4$. We set $N_{\max} = 5000$ and $\gamma = 0.95$. Setting $\lambda_{\text{pop}} = 1/201, \lambda_{\text{gen}} = \lambda_{\text{crs}} = 100/201$ is a good trade off between the three criteria. In a practical setting, the λ-s are to be chosen such that they reflect the actual costs. Since there is a cost associated with the number of crossings and the number of generations, we are able to obtain a provably optimal solution in 1.5 seconds which is depicted in Figure 9.1(b). It is important to note that this problem instance cannot be expressed in the restricted framework of Servin et al.[176]: treating the resistance loci as a single locus does not result in the best crossing schedule (see Figure 9.1(a)), as the second genotype is obtained via a crossover between the second and third locus. To the best of our knowledge, such a real-world instance is solved for the first time to provable optimality within a precise mathematical model.

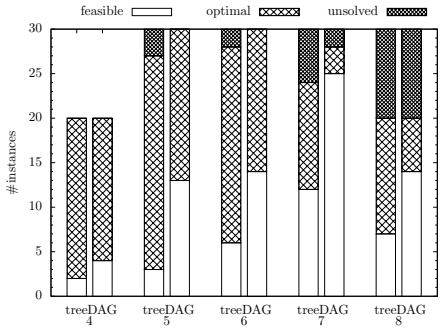
Generated instances. Due to the lack of further real-world instances, we generate random instances on which we evaluate the performance of our method. The generated instances either have 5 or 10 parents and concern 4 up to 8 loci. The number of correct alleles per parental genotype affects the difficulty of the instances, we vary this number depending on the number of loci. In total 140 instances are generated, among which 20 concern instances of 4 loci; the classes of 5-8 loci are comprised by 30 instances each. We run both the DAG and the tree version of the MIP on all instances. For the DAG case, we were able to obtain solutions to 128 instances compared to 119 instances (see Figure 9.2(a)) for the tree version. Among the unsolved instances for the tree case, there are also instances that are infeasible due to the value of $N_{\max}$ which requires re-use of genotypes. The number of instances that were solved to provable optimality in the DAG case is 58; for the tree case this number is 89. According to Figure 9.2(b), DAGs provide a gain in solution quality of up to 5% on average compared to the tree. Note that none of the instances is of the nature that is captured by Servin's model. Not surprisingly, trees are easier to solve as can be seen in Figure 9.2(b).



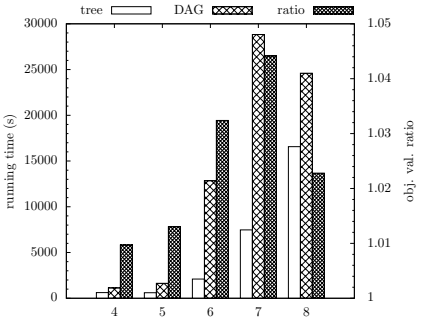
(a) $F = 3$, $G = 3$, pop = 2408 and obj = 14.69

(b) $F = 4$, $G = 4$, pop = 633 and obj = 7.13

Figure 9.1: Crossing schedules for the pepper instance. Inner nodes are obtained via crossings requiring a population size shown on the arcs, in both schedules the final crossing is a selfing. Chromosomes of an inner node are obtained via crossovers in their parents. Schedule (b) is provably optimal.



(a) Optimality of solutions



(b) Running times (instances exceeding the time limit were not considered), objective value ratio (right y-axis)

Figure 9.2: Results for generated instances

9.6 Conclusions

For the first time we have described a mathematical model capturing the problem of marker-assisted gene pyramiding to its full extent. We show that our approach is capable of solving a real-world instance and generated instances, often to provable optimality. As mentioned earlier, our method is not exact due to (i) the piecewise-linear approximation of the population size function and (ii) a simplification in (9.11) of neglecting the possibility that the two chromosomes may swap their originating genotypes. However, in our experiments we have not observed any crossing where this could have happened. The NP-hardness proof involves only the number of crossings; as for the number of generations, the same reduction can be applied. The hardness with respect to the population size remains open. Possible extensions to our problem definition include considering heterozygous ideotypes. This requires an extension to tertiary alleles. Another extension would be to consider so called 'don't care' alleles, which are alleles that are not preserved due to crossover events, and as such do not need to be considered in the probability function.

Acknowledgments. We would like to thank Bertrand Servin for kindly providing us the source code of his method. In addition we are very grateful for the constructive comments of the anonymous referees.

Chapter 10

Discussion

> Biology easily has 500 years of exciting problems to work on.
>
> Donald E. Knuth (1938)

In this thesis we have studied several combinatorial optimization problems in computational biology. These problems were based on biological questions. The approach we have taken in answering these questions consisted of four successive steps. First, we formulated the respective combinatorial problems, followed by an analysis of their complexity and combinatorial structure. Using these analyses, we then proceeded to design practically efficient algorithms. Finally, we assessed their performance using either benchmark data sets or alternative biological objective functions. In the following, we discuss the main results and list future work.

Networks. The question that we considered in Chapters 2 and 3 is rooted in the field of comparative network analysis. Here, the goal is to identify commonalities between biological networks from different strains or species, or derived from different conditions. Given two protein-protein interaction networks, we formulated the pairwise global network alignment problem as finding a partial mapping of nodes from one network to the other network with maximum score. We defined the score of an alignment in terms of aligned node and interaction pairs. We presented new algorithmic ideas in order to make a Lagrangian relaxation approach practically useful and competitive. The experimental evaluation showed that our approach outperforms competing methods in terms of biological quality of the results and running time. A recent study by Clark and Kalita [47] in which several—including more recent—network alignment methods were compared, shows that our method is among the top three performing methods. As for future work, we plan to embed our Lagrangian relaxation approach within a branch-and-bound framework that would allow (to use the bounds of the Lagrangian approach) to solve the problem to provable optimality. In addition, we want to consider the problem of local network alignment where the goal is to find highly-similar pairs of connected subnetworks in each of the two input networks. Similarly to local sequence alignment, local network alignment requires a

notion of locality as well as a graph edit distance function. The former may be captured by requiring connectivity. In sequence alignment, the scoring function takes only one entity into account: columns in the alignment in which residues and/or gaps are paired. On the other hand, the scoring function for network alignment needs to deal with two entities: nodes and edges. We plan to use a graph edit distance function to score the operations that are needed to make the two subnetworks isomorphic. We will study this problem further and apply it to topology-aware network querying where, as opposed to the method described in Chapter 3, inexact matches are allowed. Such a *relaxed isomorphism* approach is more geared toward dealing with presently available biological networks that are noisy and incomplete.

In Chapter 4 we studied the paralog mapping problem, which occurs as a subproblem in computational methods for protein-protein interaction prediction that are based on coevolution. Such methods make the following assumption: Evidence of coevolution of the protein families of two proteins indicates an evolutionary preserved interaction between the two proteins. The introduced method, CUPID, takes the multiple sequence alignments of two protein families as input and outputs a pairing of paralogs across the two families that maximizes the likelihood of coevolution. We plan to apply our approach to predict yeast protein-protein interactions using protein families described in Pfam [74]. As a benchmark data set, we want to use an experimentally determined yeast protein-protein interaction network [225]. In addition, we wish to relax the maximal matching constraint to allow for arbitrarily-sized matchings based on a parameter k. For setting k, we plan to use a permutation test to assess statistical significance.

Modules. The first two chapters of Part II concerned the active modules problem. In this problem we are given differential gene expression data overlaid on a biological network and are asked to find a connected subnetwork that is significantly differentially expressed. The corresponding combinatorial formulation is the maximum node-weighted connected subgraph problem, which we solved using integer linear programming. To facilitate the interpretation of active modules, we developed a visual analysis approach that displays set-based biological annotations as contours on top of a node-link layout. Ongoing work includes the integration of these individual components into an integrative network analysis pipeline that retrieves the network and molecular profile data, computes active modules, assesses their significance in terms of overrepresented categories, and visualizes the results. In addition, we plan to increase the performance of the integer linear program by considering an edge-based formulation.

In Chapter 7 we considered the conserved active modules problem, which is a cross-species generalization of the active modules problem. Conserved active modules are sets of genes, one for each species, which (i) induce a connected subnetwork in a species-specific interaction network, (ii) show overall differential behavior and (iii) contain a large number of orthologous genes. In contrast to existing methods, we proposed a flexible notion of conservation—controlled by the parameter $\alpha \in [0,1]$, which enforces that at least a fraction α of the genes in the conserved active modules are accompanied by an ortholog. Only for intermediate values of α we obtained biologically-interpretable modules, showing that, in our model, a flexible notion of

conservation is essential. For future work, we plan to generalize our approach beyond two networks.

Chapter 8 considered the charge group partitioning problem, which occurs in the automated parameterization of molecular compounds for use in molecular dynamics simulations. We introduced a dynamic programming formulation that exploits properties of practical input data, including low treewidth and bounded degree. A future direction is to make use of a repository of pre-parametrized compounds. Given such a repository and a novel compound, the task is to partition the input compound into connected subgraphs that occur in different compounds in the repository. These common fragments can then be used to parameterize the input compound. We can phrase this as a maximal connected common subgraph problem [128]. Interestingly, the $\alpha = 1$ case of the conserved active modules problem can also be cast into a maximal connected common subgraph problem. We plan to investigate this relation further.

Breeding schedules. In Chapter 9 we considered a combinatorial problem in the field of plant breeding. In the crossing schedule problem, we are given a set of parental genotypes and are asked to find a sequence of crossings that ultimately results in a specified, desired genotype. We solved this problem using mixed integer linear programming. Avenues for future research include a sensitivity analysis on the specified recombination frequencies. Moreover, we wish to incorporate the selection of relevant and cost-efficient loci into the crossing schedule problem by additionally considering the expected effect of individual loci on the desired phenotypic traits.

10.1 Closing remarks

As illustrated throughout this thesis, combinatorial optimization and computational biology are a good fit. It is important not to lose sight of the main biological question. To do so, it is essential to formulate a mathematical problem definition prior to designing an algorithm. If the solutions identified by the designed algorithm are unsatisfactory then it could mean that there is a large gap between the objective values of the solutions and their respective optima, or there is a discrepancy between the biological question and the formulated mathematical problem definition, or even both. The availability of a problem statement allows oneself and other researchers to design better algorithms for the same mathematical problem—better in terms of running time or objective value. In case an algorithm is exact, i.e. it only returns optimal solutions, we can directly assess the validity of the problem formulation and revise it if necessary. Unfortunately, a lot of work in computational biology presents algorithms without stating the mathematical problem formulation, thereby hampering future research.

The emergence of new technologies, as well as improvements in current technologies, presents many exciting opportunities for applications of combinatorial optimization to computational biology—as stated by Donald E. Knuth [127]:

> 'Biology easily has 500 years of exciting problems to work on.'

Bibliography

[1] W. P. Adams and T. Johnson. Improved linear programming-based lower bounds for the quadratic assignment problem. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 1994.

[2] J. Alber, F. Dorn, and R. Niedermeier. Experimental evaluation of a tree decomposition-based algorithm for vertex cover on planar graphs. *Discrete Appl Math*, 145(2):219–231, 2005.

[3] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter. *Molecular biology of the cell*. Garland Science Taylor & Francis Group, 4th edition, 2002. ISBN 0815332181.

[4] A. Alexa, J. Rahnenführer, and T. Lengauer. Improved scoring of functional groups from gene expression data by decorrelating GO graph structure. *Bioinformatics*, 22(13): 1600–1607, June 2006.

[5] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, et al. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769):503–511, Feb. 2000.

[6] M. Allen and D. Tildesley. *Computer Simulation of Liquids*. Oxford University Press, New York, 1987.

[7] U. Alon. Network motifs: theory and experimental approaches. *Nat Rev Genet*, 8(6): 450–461, 2007.

[8] B. Alper, N. Riche, G. Ramos, and M. Czerwinski. Design study of LineSets, a novel set visualization technique. *IEEE T Vis Comput Gr*, 17(12):2259–2267, 2011.

[9] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J Mol Bio*, 215(3):403–410, 1990.

[10] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. H. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*, 25(17):3389–3402, Sept. 1997.

[11] E. Álvarez-Miranda, I. Ljubić, and P. Mutzel. The maximum weight connected subgraph problem. In M. Jünger and G. Reinelt, editors, *Facets of Combinatorial Optimization*, pages 245–270. Springer Berlin Heidelberg, 2013.

[12] F. Annunziato and S. Romagnani. Do studies in humans better depict Th17 cells? *Blood*, 114(11):2213–2219, 2009.

[13] F. Annunziato, L. Cosmi, F. Liotta, E. Maggi, and S. Romagnani. Human Th17 cells: are they different from murine Th17 cells? *Eur J Immunol*, 39(3):637–640, Mar. 2009.

[14] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J Algebr Discrete Meth*, 8(2):227–284, 1987.

[15] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet*, 25(1):25–29, May 2000.

[16] M. Ashburner, C. A. Ball, J. A. Blake, et al. Gene ontology: tool for the unification of biology. *Nat Genet*, 25, 2000.

[17] N. Atias and R. Sharan. Comparative analysis of protein networks: hard problems, practical solutions. *Commun ACM*, 55(5):88–97, 2012.

[18] F. Ay, M. Kellis, and T. Kahveci. SubMAP: Aligning Metabolic Pathways with Subnetwork Mappings. *J Comput Biol*, 18(3):219–35, 2011.

[19] C. Backes, A. Rurainski, G. Klau, O. Müller, D. Stöckel, A. Gerasch, J. Küntzer, D. Maisel, N. Ludwig, M. Hein, A. Keller, H. Burtscher, M. Kaufmann, E. Meese, and H. P. Lenhof. An integer linear programming approach for finding deregulated subgraphs in regulatory networks. *Nucleic Acids Res*, 40(6):e43, 2012.

[20] A. Barsky, T. Munzner, J. Gardy, and R. Kincaid. Cerebral: Visualizing multiple experimental conditions on a graph with biological context. *IEEE T Vis Comput Gr*, 14(6): 1253–1260, 2008.

[21] A. Bateman, L. Coin, R. Durbin, R. D. Finn, V. Hollich, S. Griffiths-Jones, A. Khanna, M. Marshall, S. Moxon, E. L. Sonnhammer, D. J. Studholme, C. Yeats, and S. R. Eddy. The pfam protein families database. *Nucleic Acids Res*, 32(D1):138–141, 2004.

[22] M. Bateni, C. Chekuri, A. Ene, M. T. Hajiaghayi, N. Korula, and D. Marx. Prize-collecting Steiner problems on planar graphs. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1028–1049, 2011.

[23] G. Battista, P. Eades, R. Tamassia, and I. Tollis. *Graph drawing: algorithms for the visualization of graphs*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[24] D. Beisser, G. W. Klau, T. Dandekar, T. Müller, and M. T. Dittrich. BioNet: an R-package for the functional analysis of biological networks. *Bioinformatics*, 26(8):1129–30, Apr. 2010.

[25] H. J. C. Berendsen, D. van der Spoel, and R. Van Drunen. GROMACS: a message-passing parallel molecular dynamics implementation. *Com Phy Comm*, 91(1-3):43–56, 1995.

[26] Beriou et al. TGF-beta induces IL-9 production from human Th17 cells. *J Immunol*, 185(1):46–54, July 2010.

[27] F. Bertault and P. Eades. Drawing hypergraphs in the subset standard. In *Proceedings of the 8th International Symposium on Graph Drawing*, volume 1984 of *Lecture Notes in Computer Science*, pages 164–169, Berlin, Heidelberg, 2001. Springer.

[28] C. C. Berthier et al. Cross-species transcriptional network analysis defines shared inflammatory responses in murine and human lupus nephritis. *J Immunol*, 189(2):988–1001, July 2012.

[29] D. L. Beveridge and F. M. DiCapua. Free energy via molecular simulation: applications to chemical and biomolecular systems. *Annu Rev Biophys Biophys Chem*, 18(1):431–492, 1989.

[30] J. Blazewicz, P. Formanowicz, and M. Kasprzak. Selected combinatorial problems of computational biology. *Eur J Oper Res*, 161(3):585–597, 2005.

[31] M. Blumenstock and E. Althaus. Algorithms for the maximum weight connected subgraph and prize-collecting steiner tree problems. Contribution to the 11th DIMACS Implementation Challenge on Steiner Tree Problems., 2014. URL http://dimacs11.cs.princeton.edu/workshop/AlthausBlumenstock.pdf.

[32] H. L. Bodlaender. NC-algorithms for graphs with small treewidth. In J. van Leeuwen, editor, *Proc. 14th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 1988)*, volume 344 of *Lecture Notes in Computer Science*, pages 1–10. Springer, 1989.

[33] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theor Comput Sci*, 209(1-2):1–45, 1998.

[34] M. B. B. Boggara, A. Faraone, and R. Krishnamoorti. Effect of pH and ibuprofen on the phospholipid bilayer bending modulus. *J Phys Chem B*, 114(24):8061–8066, June 2010.

[35] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE T Pattern Anal*, 26(9):1124–1137, 2004.

[36] S. P. Bradley, A. C. Hax, and T. L. Magnanti. *Applied Mathematical Programming*. Addison-Wesley, 1977.

[37] B. R. Brooks, C. L. Brooks, III, A. D. Mackerell, Jr., L. Nilsson, R. J. Petrella, B. Roux, Y. Won, G. Archontis, C. Bartels, S. Boresch, A. Caflisch, L. Caves, Q. Cui, and e. Dinner. CHARMM: The Biomolecular Simulation Program. *J Comput Chem*, 30(10):1545–1614, 2009.

[38] J. Brown and P. Caligari. *Introduction to Plant Breeding*. Wiley-Blackwell, 2008.

[39] A. Caprara, M. Fischetti, and P. Toth. A heuristic method for the set cover problem. *Oper Res*, 47:730–743, 1999.

[40] R. Carvajal, M. Constantino, M. Goycoolea, J. Pablo Vielma, and A. Weintraub. Imposing connectivity constraints in forest planning models. *Oper Res*, 61(4):824–836, 2013.

[41] P. Casarosa, R. A. Bakker, D. Verzijl, M. Navis, H. Timmerman, R. Leurs, and M. J. Smit. Constitutive signaling of the human cytomegalovirus-encoded chemokine receptor US28. *J Biol Chem*, 276(2):1133–1137, 2001.

[42] S. H. Chang, Y. Chung, and C. Dong. Vitamin D suppresses Th17 cytokine production by inducing C/EBP homologous protein (CHOP) expression. *J Biol Chem*, 285(50):38751–38755, Dec. 2010.

[43] C.-Y. Chen and K. Grauman. Efficient activity detection with max-subgraph search. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1274–1281. IEEE, 2012.

[44] M. Chimani, C. Gutwenger, M. Jünger, G. W. Klau, K. Klein, and P. Mutzel. The open graph drawing framework (OGDF). In R. Tamassia, editor, *Handbook of Graph Drawing and Visualization*. CRC, 2013.

[45] J. Cinatl J, V. JU, K. R, and W. D. H. Oncomodulatory signals by regulatory proteins encoded by human cytomegalovirus: a novel role for viral infection in tumor progression. *FEMS Microbiol Rev*, 28(1):59–77, 2004.

[46] M. Ciofani et al. A validated regulatory network for Th17 cell specification. *Cell*, 151(2): 289–303, Oct. 2012.

[47] C. Clark and J. Kalita. A comparison of algorithms for the pairwise alignment of biological networks. *Bioinformatics*, 30(16):2351–2359, Aug. 2014.

[48] C. S. Cobbs, L. Harkins, M. Samanta, G. Y. Gillespie, S. Bharara, P. H. King, L. B. Nabors, C. G. Cobbs, and W. J. Britt. Human cytomegalovirus infection and expression in human malignant glioma. *Cancer Res*, 62(12):3347–3350, 2002.

[49] O. Cohen, H. Ashkenazy, D. Burstein, and T. Pupko. Uncovering the co-evolutionary network among prokaryotic genes. *Bioinformatics*, 28:i389–i394, 2012. ECCB 2012.

[50] B. C. Y. Collard and D. J. Mackill. Marker-assisted selection: an approach for precision plant breeding in the twenty-first century. *Phil Trans R Soc B*, 363(1491):557–572, 2008.

[51] C. Collins, G. Penn, and S. Carpendale. Bubble Sets: Revealing set relations with isocontours over existing visualizations. *IEEE T Vis Comput Gr*, 15(6):1009–1016, 2009.

[52] W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz, D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell, and P. A. Kollman. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *J Am Chem Soc*, 117:5179–5197, 1995.

[53] S. Q. Crome, A. Y. Wang, C. Y. Kang, and M. K. Levings. The role of retinoic acid-related orphan receptor variant 2 and IL-17 in the development and function of human CD4+ T cells. *Eur J Immunol*, 39(6):1480–1493, June 2009.

[54] P. Csermely, T. Korcsmáros, H. J. M. Kiss, G. London, and R. Nussinov. Structure and dynamics of molecular networks: a novel paradigm of drug discovery: a comprehensive review. *Pharmacol Therapeut*, 138(3):333–408, June 2013.

[55] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer, Berlin, Heidelberg, 2008.

[56] D. Dede and H. Oğul. TriClust: A Tool for Cross-Species Analysis of Gene Regulation. *Molecular Informatics*, 33(5):382–387, May 2014.

[57] A. K. Dehof, A. Rurainski, Q. B. A. Bui, S. Böcker, H.-P. Lenhof, and A. Hildebrandt. Automated bond order assignment as an optimization problem. *Bioinformatics*, 27(5): 619–625, 2011.

[58] J. C. M. Dekkers and F. Hospital. The use of molecular genetics in the improvement of agricultural populations. *Nat Rev Genet*, 3:22–32, 2002.

[59] R. Deshpande, S. Sharma, C. M. Verfaillie, W.-S. Hu, and C. L. Myers. A scalable approach for discovering conserved active subnetworks across species. *PLoS Comput Biol*, 6(12):e1001028, 2010.

[60] B. Dezső, A. Jüttner, and P. Kovács. LEMON–an open source C++ graph template library. *Electron Notes Theor Comput Sci*, 264(5):23–45, 2011.

[61] B. Dilkina and C. P. Gomes. Solving connected subgraph problems in wildlife conservation. In *CPAIOR'10: Proceedings of the 7th international conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Springer, June 2010.

[62] K. Dinkla, M. van Kreveld, B. Speckmann, and M. Westenberg. Kelp diagrams: Point set membership visualization. *Comput Graph Forum*, 31(3):875–884, 2012.

[63] M. T. Dittrich, G. W. Klau, A. Rosenwald, T. Dandekar, and T. Müller. Identifying functional modules in protein-protein interaction networks: an integrated exact approach. *Bioinformatics*, 24(13):i223–31, July 2008.

[64] C. W. Duin and A. Volgenant. Some generalizations of the Steiner problem in graphs. *Networks*, 17(3):353–364, 1987.

[65] M. J. Dunning, M. L. Smith, M. E. Ritchie, and S. Tavaré. beadarray: R classes and methods for Illumina bead-based data. *Bioinformatics*, 23(16):2183–4, 2007.

[66] T. Dwyer, K. Marriott, and P. Stuckey. Fast node overlap removal. In *Graph Drawing*, volume 3843 of *Lecture Notes in Computer Science*, pages 153–164. Springer, 2006.

[67] T. Dwyer, K. Marriott, F. Schreiber, P. Stuckey, M. Woodward, and M. Wybrow. Exploration of networks using overview+detail with constraint-based cooperative layout. *IEEE T Vis Comput Gr*, 14(6):1293–1300, 2008.

[68] M. Dyer and A. Frieze. On the complexity of partitioning graphs into connected subgraphs. *Discrete Appl Math*, 10(2):139–153, 1985.

[69] J. Edmonds. Path, trees, and flowers. *Canadian J Math*, 17:449–467, 1965.

[70] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *J ACM*, 19:248–264, 1972.

[71] D. Eisenberg, E. M. Marcotte, I. Xenarios, and T. O. Yeates. Protein function in the post-genomic era. *Nature*, 405(6788):823–826, 06 2000.

[72] M. El-Kebir, J. Heringa, and G. W. Klau. Lagrangian relaxation applied to sparse global network alignment. In *Pattern Recognition in Bioinformatics, PRIB 2011, Delft, The Netherlands, November 2–4, 2011*, pages 225–236, 2011.

[73] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of muliticast transmissions. In *The thirty-second annual ACM symposium*, pages 218–227, New York, New York, USA, 2000. ACM Press.

[74] R. D. Finn, A. Bateman, J. Clements, P. Coggill, R. Y. Eberhardt, S. R. Eddy, A. Heger, K. Hetherington, L. Holm, J. Mistry, E. L. L. Sonnhammer, J. Tate, and M. Punta. Pfam: the protein families database. *Nucleic Acids Res*, 42(D1):D222–30, Jan. 2014.

[75] J. Flannick, A. Novak, B. S. Srinivasan, H. H. McAdams, and S. Batzoglou. Graemlin: general and robust alignment of multiple large interaction networks. *Genome Res*, 16 (9):1169–1181, 2006.

[76] P. Flicek, I. Ahmed, M. R. Amode, D. Barrell, K. Beal, S. Brent, D. Carvalho-Silva, P. Clapham, G. Coates, S. Fairley, et al. Ensembl 2013. *Nucleic Acids Res*, 41(D1): D48–D55, 2013.

[77] A. Franceschini, D. Szklarczyk, S. Frankild, M. Kuhn, M. Simonovic, A. Roth, J. Lin, P. Minguez, P. Bork, C. von Mering, and L. J. Jensen. STRING v9.1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Res*, 41 (D1):D808–D815, Dec. 2013.

[78] A. Franceschini, D. Szklarczyk, S. Frankild, M. Kuhn, M. Simonovic, A. Roth, J. Lin, P. Minguez, P. Bork, C. von Mering, et al. STRING v9.1: protein-protein interaction networks, with increased coverage and integration. *Nucleic acids research*, 41(D1):D808–D815, 2013.

[79] Y. Frishman and A. Tal. Online dynamic graph drawing. *IEEE T Vis Comput Gr*, 14(4): 727–740, July 2008.

[80] T. Fruchterman and E. Reingold. Graph drawing by force-directed placement. *Software Pract Exper*, 21(11):1129–1164, 1991.

[81] K. J. Fryxell. The coevolution of gene family trees. *Trends Genet*, 12(9):364–369, 1996.

[82] M. K. Gandhi and R. Khanna. Human cytomegalovirus: clinical aspects, immune regulation, and emerging treatments. *Lancet Infect Dis*, 4(12):725–738, 2004.

[83] E. R. Gansner, Y. Hu, and S. Kobourov. Gmap: Visualizing graphs and clusters as maps. In *Pacific Visualization Symposium (PacificVis)*, pages 201–208. IEEE, 2010.

[84] M. Garey and D. Johnson. *Computers and Intractability*. Freeman, 1979.

[85] L. Gautier, L. Cope, B. M. Bolstad, and R. A. Irizarry. affy—analysis of affymetrix genechip data at the probe level. *Bioinformatics*, 20(3):307–315, 2004.

[86] N. Gehlenborg, S. I. O'Donoghue, N. S. Baliga, A. Goesmann, M. A. Hibbs, H. Kitano, O. Kohlbacher, H. Neuweger, R. Schneider, D. Tenenbaum, and A.-C. Gavin. Visualization of omics data for systems biology. *Nat Methods*, 7(3s):S56–S68, 2010.

[87] P. R. Gerber. Charge distribution from a simple molecular orbital type calculation and non-bonding interaction terms in the force field mab. *J Comput Aid Mol Des*, 12(1): 37–51, 1998.

[88] C. Gkantsidis, M. Mihail, and E. Zegura. The Markov chain simulation method for generating connected power law random graphs. In *Proceedings of the Fifth Workshop on Algorithm Engineering and Experiments*, volume 111, page 16. SIAM, 2003.

[89] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, and M. A. Caligiuri. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[90] S. Goto, Y. Okuno, M. Hattori, T. Nishioka, and M. Kanehisa. LIGAND: database of chemical compounds and reactions in biological pathways. *Nucleic Acids Res*, 30(1): 402–404, 2002.

[91] M. Guignard. Lagrangean relaxation. *Top*, 11:151–200, 2003.

[92] D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, New York, NY, USA, 1997. ISBN 0-521-58519-8.

[93] I. Hajirasouliha, A. Schönhuth, D. de Juan, A. Valencia, and S. C. Sahinalp. Mirroring co-evolving trees in the light of their topologies. *Bioinformatics*, 28(9):1202–1208, 2012.

[94] J. B. S. Haldane. The combination of linkage values and the calculation of distances between the loci of linked factors. *J Genet*, 8:299–309, 1919.

[95] L. Harkins, A. L. Volk, M. Samanta, I. Mikolaenko, W. J. Britt, K. I. Bland, and C. S. Cobbs. Specific localisation of human cytomegalovirus nucleic acids and proteins in human colorectal cancer. *Lancet*, 360(9345):1557–1563, 2002.

[96] T. J. Harris et al. Cutting edge: An in vivo requirement for STAT3 signaling in TH17 development and TH17-dependent autoimmunity. *J Immunol*, 179(7):4313–4317, Oct. 2007.

[97] M. Harrower and C. Brewer. *ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps*, pages 261–268. John Wiley & Sons, Ltd, Chichester, UK, 2011.

[98] M. Held and R. M. Karp. The traveling-salesman problem and minimum spanning trees: Part II. *Math Program*, 1:6–25, 1971.

[99] I. Herman, G. Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE T Vis Comput Gr*, 6(1):24–43, 2000.

[100] M. H. Herynk, R. Tsan, R. Radinsky, and G. E. Gallick. Activation of c-Met in colorectal carcinoma cells leads to constitutive association of tyrosine-phosphorylated β-catenin. *Clin Exp Metastas*, 20(4):291–300, 2003.

[101] M. Heuer and M. Smoot. Venn and Euler Diagrams. http://apps.cytoscape.org/apps/vennandeulerdiagrams, 2013.

[102] D. S. Hochbaum and A. Pathria. Node-optimal connected k-subgraphs. 1994.

[103] T. Horváth, J. Ramon, and S. Wrobel. Frequent subgraph mining in outerplanar graphs. *Data Min Knowl Discov*, 21(3):472–508, 2010.

[104] D. W. Huang, B. T. Sherman, and R. A. Lempicki. Systematic and integrative analysis of large gene lists using david bioinformatics resources. *Nat Protoc*, 4(1):44–57, 2009.

[105] F. Hüffner, N. Betzler, and R. Niedermeier. Separator-based data reduction for signed graph balancing. *J Comb Optim*, 20(4):335–360, 2010. ISSN 1382-6905. doi: 10.1007/s10878-009-9212-2.

[106] IBM Corp. IBM ILOG CPLEX optimization studio, 2014.

[107] T. Ideker and R. Sharan. Protein networks in disease. *Genome Res*, 18:644–652, 2008.

[108] T. Ideker, O. Ozier, B. Schwikowski, and A. F. Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics*, 18(Suppl 1):S233–S240, July 2002.

[109] T. Ishii and K. Yonezawa. Optimization of the marker-based procedures for pyramiding genes from multiple donor lines: I. Schedule of crossing between the donor lines. *Crop Sci*, 47:537–546, 2007.

[110] T. Ishii and K. Yonezawa. Optimization of the marker-based procedures for pyramiding genes from multiple donor lines: II. Strategies for selecting the objective homozygous plant. *Crop Sci*, 47:1878–1886, 2007.

[111] J. M. G. Izarzugaza, D. Juan, C. Pons, F. Pazos, and A. Valencia. Enhancing the prediction of protein pairings between interacting families using orthology information. *BMC Bioinformatics*, 9:35, 2008.

[112] S. Jaeger, C. Sers, and U. Leser. Combining modularity, conservation, and interactions of proteins significantly increases precision and coverage of protein function prediction. *BMC Genomics*, 11(1):717, 2010.

[113] D. S. Johnson. The NP-completeness column: An ongoing guide. *J Algorithm*, 6(1): 145–159, Mar. 1985.

[114] D. S. Johnson, M. Minkoff, and S. Phillips. The prize collecting Steiner tree problem: theory and practice. In *SODA '00: Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, Feb. 2000.

[115] W. L. Jorgensen, D. S. Maxwell, and J. Tirado-Rives. Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids. *J Am Chem Soc*, 118(45):11225–11236, 1996.

[116] D. Juan, F. Pazos, and A. Valencia. High-confidence prediction of global interactomes based on genome-wide coevolutionary networks. *P Natl Acad Sci USA*, 105(3):934–939, 2008.

[117] M. Kalaev, M. Smoot, T. Ideker, and R. Sharan. NetworkBLAST: comparative analysis of protein networks. *Bioinformatics*, 24(4th):594–596, 2008.

[118] M. Kanehisa and S. Goto. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Res*, 28(1):27–30, Jan. 2000.

[119] M. Kanehisa, S. Goto, M. Hattori, et al. From genomics to chemical genomics: new developments in KEGG. *Nucleic Acids Res*, 34:D354–D357, 2006.

[120] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[121] R. M. Karp. Mathematical challenges from genomics and molecular biology. *Not Am Math Soc*, 49:544–553, 2002.

[122] R. M. Karp. Heuristic algorithms in computational molecular biology. *J Comput Syst Sci*, 77(1):122–128, 2011. ISSN 0022-0000. Celebrating Karp's Kyoto Prize.

[123] B. P. Kelley, R. Sharan, R. M. Karp, T. Sittler, D. E. Root, B. R. Stockwell, and T. Ideker. Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *P Natl Acad Sci USA*, 100(20):11394–11399, 2003.

[124] B. P. Kelley, B. Yuan, F. Lewitter, R. Sharan, B. R. Stockwell, and T. Ideker. PathBLAST: a tool for alignment of protein interaction networks. *Nucleic Acids Res*, 32:W83–88, 2004.

[125] S. Kerrien, B. Aranda, L. Breuza, A. Bridge, F. Broackes-Carter, C. Chen, M. Duesbury, M. Dumousseau, M. Feuermann, U. Hinz, C. Jandrasits, R. C. Jimenez, J. Khadake, U. Mahadevan, P. Masson, I. Pedruzzi, Pfeiffenberger, E., P. Porras, A. Raghunath, B. Roechert, S. Orchard, and H. Hermjakob. The IntAct molecular interaction database in 2012. *Nucleic Acids Res*, 40:D841–D846, 2012.

[126] G. W. Klau. A new graph-based method for pairwise global network alignment. *BMC Bioinformatics*, 10 Suppl 1:S59, 2009.

[127] D. E. Knuth. Computer Literacy Bookshops Interview, Dec. 7th 1993.

[128] I. Koch. Enumerating all connected maximal common subgraphs in two graphs. *Theor Comput Sci*, 2001.

[129] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[130] M. Koyutürk, Y. Kim, U. Topkara, et al. Pairwise alignment of protein interaction networks. *J Comput Biol*, 13(2):182–199, 2006.

[131] E. Kristiansson, T. Österlund, L. Gunnarsson, G. Arne, D. G. J. Larsson, and O. Nerman. A novel method for cross-species gene expression analysis. *BMC Bioinformatics*, 14:70, 2013.

[132] J. B. Kruskal. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *P Am Math Soc*, 7(1):48, Feb. 1956.

[133] O. Kuchaiev, T. Milenkovic, V. Memisevic, W. Hayes, and N. Przulj. Topological network alignment uncovers biological function and phylogeny. *J R Soc Interface*, 7(50):1341–54, 2010.

[134] H. W. Kuhn. The Hungarian method for the assignment problem. *Nav Res Logist Q*, 2 (1–2):83–97, 1955.

[135] G. Landan and D. Graur. Heads or tails: a simple reliability check for multiple sequence alignments. *Mol Biol Evol*, 24(6):1380–1383, 2007.

[136] E. V. Langemeijer, E. Slinger, S. de Munnik, A. Schreiber, D. Maussang, H. Vischer, F. Verkaar, R. Leurs, M. Siderius, and M. J. Smit. Constitutive β-catenin signaling by the viral chemokine receptor US28. *PLoS ONE*, 7(11):e48935, 11 2012.

[137] E. L. Lawler. The quadratic assignment problem. *Manage Sci*, 9(4):586–599, 1963.

[138] H. F. Lee and D. R. Dooly. Decomposition algorithms for the maximum-weight connected graph problem. *Nav Res Log*, 1998.

[139] J. A. Lemkul, W. J. Allen, and D. R. Bevan. Practical considerations for building GROMOS-compatible small-molecule topologies. *J Chem Inf Model*, 50(12):2221–2235, 2010.

[140] I. Ljubic, R. Weiskircher, U. Pferschy, G. W. Klau, P. Mutzel, and M. Fischetti. An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem. *Math Program*, 105(2-3):427–449, 2006.

[141] Y. Lu, R. Rosenfeld, G. J. Nau, and Z. Bar-Joseph. Cross species expression analysis of innate immune response. *J Comput Biol*, 17(3):253–68, Mar. 2010.

[142] R. MacCallum, S. Redmond, and G. Christophides. An expression map for anopheles gambiae. *BMC Genomics*, 12(1):620, 2011.

[143] T. L. Magnanti and L. A. Wolsey. *Optimal trees*, volume 7, chapter 9, pages 503–615. Elsevier Science B.V., 1995.

[144] A. K. Malde, L. Zuo, M. Breeze, M. Stroet, D. Poger, P. C. Nair, C. Oostenbrink, and A. E. Mark. An automated force field topology builder (ATB) and repository: version 1.0. *J Chem Theory Comput*, 7(12):4026–4037, 2011.

[145] D. Maussang, D. Verzijl, M. van Walsum, R. Leurs, J. Holl, O. Pleskoff, D. Michel, G. A. M. S. van Dongen, and M. J. Smit. Human cytomegalovirus-encoded chemokine receptor US28 promotes tumorigenesis. *P Natl Acad Sci USA*, 103(35):13068–13073, 2006.

[146] D. Maussang, E. Langemeijer, C. P. Fitzsimons, M. Stigter-van Walsum, R. Dijkman, M. K. Borg, E. Slinger, A. Schreiber, D. Michel, C. P. Tensen, G. A. van Dongen, R. Leurs, and M. J. Smit. The human cytomegalovirus-encoded chemokine receptor US28 promotes angiogenesis and tumor formation via cyclooxygenase-2. *Cancer Res*, 69(7):2861–2869, 2009.

[147] M. J. McGeachy and D. J. Cua. Th17 cell differentiation: the long and winding road. *Immunity*, 28(4):445–453, Apr. 2008.

[148] W. Meulemans, N. H. Riche, B. Speckmann, B. Alper, and T. Dwyer. KelpFusion: A hybrid set visualization technique. *IEEE T Vis Comput Gr*, 19(11):1846–1858, 2013.

[149] R. Minisini, C. Tulone, A. Lüske, D. Michel, T. Mertens, P. Gierschik, and B. Moepps. Constitutive inositol phosphate formation in cytomegalovirus-infected human fibroblasts is due to expression of the chemokine receptor homologue pUS28. *J Virol*, 77(8): 4489–4501, 2003.

[150] K. Mitra, A.-R. Carvunis, S. K. Ramesh, and T. Ideker. Integrative approaches for finding modular structure in biological networks. *Nat Rev Genet*, 14(10):719–732, Oct. 2013.

[151] S. P. Moose and R. H. Mumm. Molecular plant breeding as the foundation for 21st century crop improvement. *Plant Physiol*, 147:969–977, 2008.

[152] J. Munkres. Algorithms for the assignment and transportation problems. *SIAM J Appl Math*, 5:32–38, 1957.

[153] A. O'Garra, B. Stockinger, and M. Veldhoen. Differentiation of human T(H)-17 cells does require TGF-beta! *Nat Immunol*, 9(6):588–590, June 2008.

[154] J. Okyere, E. Oppon, D. Dzidzienyo, L. Sharma, and G. Ball. Cross-Species Gene Expression Analysis of Species Specific Differences in the Preclinical Assessment of Pharmaceutical Compounds. *PLoS ONE*, 9(5):e96853, May 2014.

[155] C. Oostenbrink, A. Villa, A. E. Mark, and W. F. van Gunsteren. A biomolecular force field based on the free enthalpy of hydration and solvation: The GROMOS force-field parameter sets 53A5 and 53A6. *J Comp Chem*, 25(13):1656–1676, 2004.

[156] R. A. Pache, A. Céol, and P. Aloy. Netaligner–a network alignment server to compare complexes, pathways and whole interactomes. *Nucleic Acids Res*, 40:W157–W161, 2012.

[157] H. Park et al. A distinct lineage of CD4 T cells regulates tissue inflammation by producing interleukin 17. *Nat Immunol*, 6(11):1133–1141, Nov. 2005.

[158] F. Pazos and A. Valencia. Similarity of phylogenetic trees as indicator of protein-protein interaction. *Protein Eng*, 14(9):609–614, 2001.

[159] O. Penn, E. Privman, G. Landan, D. Graur, and T. Pupko. An alignment confidence score capturing robustness to guide tree uncertainty. *Mol Biol Evol*, 27(8):1759–1767, 2010.

[160] P. A. Pevzner. *Computational Molecular Biology: An Algorithmic Approach*. A Bradford Book, 1 edition, Aug. 2000. ISBN 0262161974.

[161] S. Pounds and S. W. Morris. Estimating the occurrence of false positives and false negatives in microarray studies by approximating and partitioning the empirical distribution of p-values. *Bioinformatics*, 19(10):1236–1242, July 2003.

[162] R. Purcell, M. Childs, R. Maibach, C. Miles, C. Turner, A. Zimmermann, and M. Sullivan. HGF/c-Met related activation of beta-catenin in hepatoblastoma. *J Exp Clin Canc Res*, 30(1):96, 2011.

[163] H. Qin et al. TGF-beta promotes Th17 cell development through inhibition of SOCS3. *J Immunol*, 183(1):97–105, July 2009.

[164] J. Randolph-Habecker, B. Rahill, B. Torok-Storb, J. Vieira, P. E. Kolattukudy, B. H. Rovin, and D. D. Sedmak. The expression of the cytomegalovirus chemokine receptor homolog US28 sequesters biologically active CC chemokines and alters IL-8 production. *Cytokine*, 19(1):37–46, 2002.

[165] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proc. 29th ACM Symp. on Theory of Computing*, pages 475–484, 1997.

[166] Resource for Biocomputing, Visualization, and Informatics. RBVI Cytoscape Plugins – Cytoscape Group Support. http://www.rbvi.ucsf.edu/cytoscape/groups, 2012.

[167] M. Richard, J. Louahed, J. B. Demoulin, and J. C. Renauld. Interleukin-9 regulates NF-kappaB activity through BCL3 gene induction. *Blood*, 93(12):4318–4327, June 1999.

[168] N. Riche and T. Dwyer. Untangling Euler diagrams. *IEEE T Vis Comput Gr*, 16(6): 1090–1099, 2010.

[169] N. Robertson and P. D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *J Algorithm*, 7(3):309–322, 1986.

[170] Q. Ruan, S.-J. Zheng, S. Palmer, R. J. Carmody, and Y. H. Chen. Roles of Bcl-3 in the pathogenesis of murine type 1 diabetes. *Diabetes*, 59(10):2549–2557, Oct. 2010.

[171] N. Schmid, A. Eichenberger, A. Choutko, S. Riniker, M. Winger, A. E. Mark, and W. F. van Gunsteren. Definition and testing of the GROMOS force-field versions 54A7 and 54B7. *Eur Biophys J*, 40:843–856, 2011.

[172] B. U. Schraml et al. The AP-1 transcription factor Batf controls T(H)17 differentiation. *Nature*, 460(7253):405–409, July 2009.

[173] A. Schrijver. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003.

[174] A. W. Schüttelkopf and D. M. van Aalten. PRODRG: a tool for high-throughput crystallography of protein-ligand complexes. *Acta Crystallogr*, 60:1355–1363, 2004. ISSN 0907–4449.

[175] W. R. P. Scott, P. H. Hunenberger, I. G. Tironi, A. E. Mark, S. R. Billeter, J. Fennen, A. E. Torda, T. Huber, P. Kruger, and W. F. van Gunsteren. The GROMOS biomolecular simulation program package. *J Phys Chem A*, 103(19):3596–3607, 1999.

[176] B. Servin, O. C. Martin, M. Mézard, and F. Hospital. Toward a theory of marker-assisted gene pyramiding. *Genetics*, 168(1):513–523, 2004.

[177] R. Sharan and T. Ideker. Modeling cellular machinery through biological network comparison. *Nat Biotechnol*, 24(4):427–433, 2006.

[178] R. Sharan, T. Ideker, B. Kelley, R. Shamir, and R. M. Karp. Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data. *J Comput Biol*, 12(6):835–846, 2005.

[179] M. Sharma, S. Khanna, G. Bulusu, and A. Mitra. Comparative modeling of thioredoxin glutathione reductase from Schistosoma mansoni: a multifunctional target for antischistosomal therapy. *J Mol Graphics Model*, 27(6):665–675, 2009.

[180] C. Shifriss, M. Pilowsky, and J. M. Zacks. Resistance to Leveillula Taurica mildew (=Oidiopsis taurica) in Capsicum annuum. *Phytoparasitica*, 20(4):279–283, 1992.

[181] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE T Vis Comput Gr*, 12(5):733–740, 2006.

[182] P. Simonetto, D. Auber, and D. Archambault. Fully automatic visualisation of overlapping sets. *Comput Graph Forum*, 28(3):967–974, 2009.

[183] R. Singh, J. Xu, and B. Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *P Natl Acad Sci USA*, 105(35): 12763–12768, 2008.

[184] E. Slinger, D. Maussang, A. Schreiber, M. Siderius, A. Rahbar, A. Fraile-Ramos, S. A. Lira, C. Soderberg-Naucler, and M. J. Smit. HCMV-encoded chemokine receptor US28 mediates proliferative signaling through the IL-6-STAT3 axis. *Sci Signal*, 3(133):ra58, 2010.

[185] A. M. Smith, W. Xu, Y. Sun, J. R. Faeder, and G. E. Marai. Rulebender: integrated modeling, simulation and visualization for rule-based intracellular biochemistry. *BMC Bioinformatics*, 13(Suppl 8):S3, 2012.

[186] M. E. Smoot, K. Ono, J. Ruscheinski, P.-L. Wang, and T. Ideker. Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics*, 27(3):431–432, 2011.

[187] G. K. Smyth. Limma: linear models for microarray data. In R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, and W. Huber, editors, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, pages 397–420. Springer, New York, 2005.

[188] C. Söderberg-Nauclér. Does cytomegalovirus play a causative role in the development of various inflammatory diseases and cancer? *J Inter Med*, 259(3):219–246, 2006.

[189] R. E. Steuer. *Multiple Criteria Optimization: Theory, Computation and Application*. Krieger Pub Co, 1986.

[190] D. N. Streblow, C. Soderberg-Naucler, J. Vieira, P. Smith, E. Wakabayashi, F. Ruchti, K. Mattison, Y. Altschuler, and J. A. Nelson. The human cytomegalovirus chemokine receptor US28 mediates vascular smooth muscle cell migration. *Cell*, 99(5):511–520, 1999.

[191] D. N. Streblow, J. Vomaske, P. Smith, R. Melnychuk, L. Hall, D. Pancheva, M. Smit, P. Casarosa, D. D. Schlaepfer, and J. A. Nelson. Human cytomegalovirus chemokine receptor US28-induced smooth muscle cell migration is mediated by focal adhesion kinase and Src. *J Biol Chem*, 278(50):50456–50465, 2003.

[192] K. Sugiyama and K. Misue. Visualization of structural information: automatic drawing of compound digraphs. *IEEE T Syst Man Cyb*, 21(4):876–892, 1991.

[193] D. Szklarczyk, A. Franceschini, M. Kuhn, et al. The STRING database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic Acids Res*, 39:D561–D568, 2011.

[194] A. L. Tarca, S. Draghici, P. Khatri, S. S. Hassan, P. Mittal, J.-s. Kim, C. J. Kim, J. P. Kusanovic, and R. Romero. A novel signaling pathway impact analysis. *Bioinformatics*, 25(1):75–82, Jan. 2009.

[195] R. Tatusov, E. Koonin, and D. Lipman. A genomic perspective on protein families. *Science*, 278(5338):631–637, 1997.

[196] The UniProt Consortium. Reorganizing the protein space at the Universal Protein Resource (UniProt). *Nucleic Acids Res*, 40:D71–D75, 2012.

[197] E. R. M. Tillier and R. L. Charlebois. The human protein coevolution network. *Genome Res*, 19(10):1861–1871, 2009.

[198] S. Tuomela et al. Identification of early gene expression changes during human Th17 cell differentiation. *Blood*, 119(23):e151–60, June 2012.

[199] R. Uehara. The number of connected components in graphs and its applications. Technical Report IEICE Technical Report COMP99-10, Natural Science Faculty, Komazawa University, Japan, 1999.

[200] P. Uetz and R. L. Finley, Jr. From protein networks to biological systems. *FEBS Lett*, 579(8):1821–1827, 2005.

[201] M. J. van de Vijver, Y. D. He, L. J. van't Veer, H. Dai, A. A. M. Hart, D. W. Voskuil, G. J. Schreiber, J. L. Peterse, C. Roberts, M. J. Marton, M. Parrish, D. Atsma, A. Witteveen, A. Glas, L. Delahaye, T. van der Velde, H. Bartelink, S. Rodenhuis, E. T. Rutgers, S. H. Friend, and R. Bernards. A gene-expression signature as a predictor of survival in breast cancer. *New Engl J Med*, 347(25):1999–2009, Dec. 2002.

[202] W. F. van Gunsteren, D. Bakowies, R. Baron, I. Chandrasekhar, M. Christen, X. Daura, P. Gee, D. P. Geerke, A. GlÃďttli, P. H. Hünenberger, M. A. Kastenholz, C. Oostenbrink, M. Schenk, D. Trzesniak, N. F. A. van der Vegt, and H. B. Yu. Biomolecular modeling: Goals, problems, perspectives. *Angew Chem Int Ed*, 45(25):4064–4092, 2006.

[203] J. P. van Hamburg, M. J. W. de Bruijn, C. Ribeiro de Almeida, M. van Zwam, M. van Meurs, E. de Haas, L. Boon, J. N. Samsom, and R. W. Hendriks. Enforced expression of GATA3 allows differentiation of IL-17-producing cells, but constrains Th17-mediated pathology. *Eur J Immunol*, 38(9):2573–2586, Sept. 2008.

[204] M. P. van Iersel, T. Kelder, A. R. Pico, K. Hanspers, S. Coort, B. R. Conklin, and C. Evelo. Presenting and exploring biological pathways with PathVisio. *BMC Bioinformatics*, 9 (1):399, 2008.

[205] R. E. van Kesteren, C. P. Tensen, A. B. Smit, J. van Minnen, L. F. Kolakowski, W. Meyerhof, D. Richter, H. van Heerikhuizen, E. Vreugdenhil, and W. P. Geraerts. Co-evolution of ligand-receptor pairs in the vasopressin/oxytocin superfamily of bioactive peptides. *J Biol Chem*, 271(7):3619–3626, 1996.

[206] V. van Noort, B. Snel, and M. A. Huynen. Predicting gene function by conserved co-expression. *Trends Genet*, 19(5):238–242, May 2003.

[207] C. J. Vaske, S. C. Benz, J. Z. Sanborn, D. Earl, C. Szeto, J. Zhu, D. Haussler, and J. M. Stuart. Inference of patient-specific pathway activities from multi-dimensional cancer genomics data using PARADIGM. *Bioinformatics*, 26(12):i237–i245, June 2010.

[208] J. Vesanto. SOM-based data visualization methods. *Intell Data Anal*, 3(2):111–126, 1999.

[209] A. Villa and A. E. Mark. Calculation of the free energy of solvation for neutral analogs of amino acid side chains. *J Comp Chem*, 23(5):548–553, 2002.

[210] Vivid Solutions. Java Topology Suite. http://www.vividsolutions.com/jts, 2003.

[211] T. von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. van Wijk, J.-D. Fekete, and D. Fellner. Visual analysis of large graphs: State-of-the-art and future research challenges. *Comput Graph Forum*, 30(6):1719–1749, 2011.

[212] P. Waltman, T. Kacmarczyk, A. R. Bate, D. B. Kearns, D. J. Reiss, P. Eichenberger, and R. Bonneau. Multi-species integrative biclustering. *Genome Biol*, 11(9):R96, 2010.

[213] L. Wei, A. Laurence, K. M. Elias, and J. J. O'Shea. IL-21 is produced by Th17 cells and drives IL-17 production in a STAT3-dependent manner. *J Biol Chem*, 282(48):34605–34610, 2007.

[214] D. L. Wheeler, T. Barrett, D. A. Benson, S. H. Bryant, K. Canese, V. Chetvernin, D. M. Church, M. DiCuccio, R. Edgar, S. Federhen, L. Y. Geer, Y. Kapustin, O. Khovayko, D. Landsman, D. J. Lipman, T. L. Madden, D. R. Maglott, J. Ostell, V. Miller, K. D.

Pruitt, G. D. Schuler, E. Sequeira, S. T. Sherry, K. Sirotkin, A. Souvorov, G. Starchenko, R. L. Tatusov, T. A. Tatusova, L. Wagner, and E. Yaschenko. Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res*, 35(D1):D5–D12, Jan. 2007.

[215] C. M. Wilke, K. Bishop, D. Fox, and W. Zou. Deciphering the role of Th17 cells in human disease. *Trends Immunol*, 32(12):603–611, Dec. 2011.

[216] I. Wohlers, R. Andonov, and G. W. Klau. Algorithm engineering for optimal alignment of protein structure distance matrices. *Optim Lett*, 2011. ISSN 1862–4472.

[217] L. Wolsey. *Integer programming*. Wiley-Interscience series in discrete mathematics and optimization. Wiley, 1998.

[218] A. Yamaguchi, K. F. Aoki, and H. Mamitsuka. Graph complexity of chemical compounds in biological pathways. *Genome Inform*, 14:376–377, 2003.

[219] T. Yamamoto, H. Bannai, M. Nagasaki, and S. Miyano. Better Decomposition Heuristics for the Maximum-Weight Connected Graph Problem Using Betweenness Centrality. In *Discovery Science*, pages 465–472. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[220] C. Yang, X. Zhu, J. Li, and R. Shi. Exploration of the mechanism for LPFFD inhibiting the formation of beta-sheet conformation of A beta(1-42) in water. *J Mol Model*, 16(4): 813–21, 2010.

[221] X. O. Yang et al. T helper 17 lineage differentiation is programmed by orphan nuclear receptors ROR alpha and ROR gamma. *Immunity*, 28(1):29–39, Jan. 2008.

[222] G. Ye and K. F. Smith. Marker-assisted gene pyramiding for inbred line development: Basic principles and practical guidelines. *International Journal of Plant Breeding*, 2(1): 1–10, 2008.

[223] C.-H. Yeang and D. Haussler. Detecting coevolution in and among protein domains. *PLoS Comput Biol*, 3(11):e211, 2007.

[224] N. Yosef et al. Dynamic regulatory network controlling TH17 cell differentiation. *Nature*, 496(7446):461–468, Apr. 2013.

[225] H. Yu, P. Braun, M. A. Yildirim, et al. High-quality binary protein interaction map of the yeast interactome network. *Science*, 322(5898):104–110, Oct. 2008.

[226] W. Zheng and R. A. Flavell. The transcription factor GATA-3 is necessary and sufficient for Th2 cytokine gene expression in CD4 T cells. *Cell*, 89(4):587–596, May 1997.

[227] B.-M. Zhu, Y. Ishida, G. W. Robinson, M. Pacher-Zavisin, A. Yoshimura, P. M. Murphy, and L. Hennighausen. SOCS3 negatively regulates the gp130-STAT3 pathway in mouse skin wound healing. *J Invest Dermatol*, 128(7):1821–1829, July 2008.

[228] J. Zhurinsky, M. Shtutman, and A. Ben-Ze'ev. Differential mechanisms of LEF/TCF family-dependent transcriptional activation by β-catenin and plakoglobin. *Mol Cell Biol*, 20(12):4238–4252, 2000.

[229] G. E. Zinman, S. Naiman, D. M. O'Dee, N. Kumar, G. J. Nau, H. Y. Cohen, and Z. Bar-Joseph. ModuleBlast: identifying activated sub-networks within and across species. *Nucleic Acids Res*, 43(3):e20–e20, Feb. 2015.

Summary

We consider several problems in computational biology. To resolve these, we apply combinatorial optimization techniques using the following scheme. The starting point is to formulate the biological problem as a combinatorial optimization problem by defining the space of feasible solutions together with an objective function that operates on this space—assigning an objective value to each feasible solution. The aim is, thus, to find a solution whose objective value is optimal. To do so, we analyze the complexity and combinatorial structure of the formulation. These analyses give us insights that we use in designing a practically efficient algorithm. Finally, we assess the biological quality of the identified solutions of practical problem instances.

This thesis is split up in three parts: Networks, modules and breeding schedules. The first part starts with a biological problem rooted in comparative network analysis. Here, the goal is to identify commonalities between biological networks from different strains or species, or derived from different conditions. We solve this problem using Lagrangian relaxation. Next, we focus on the prediction of protein-protein interactions using the notion of coevolution: Evidence of coevolution of the protein families of two proteins may indicate an evolutionary preserved interaction between the two proteins. For solving this problem, we reuse the combinatorial problem formulation for network alignment.

The problems we consider in the second part concern the extraction of smaller connected subnetworks from a larger network. We start by considering the maximum-weight connected subgraph problem, which is a combinatorial formulation of the active module problem: Given differential expression data and a protein-protein interaction network, find a connected subnetwork that is significantly differentially expressed. For solving this problem, we use integer linear programming by applying a branch-and-cut scheme. To interpret identified active modules, we introduce a set-based visualization technique. In follow-up work, we generalize the active module problem across species. Another problem we consider is the partitioning of a molecule into charge groups. This problem occurs in the automated parameterization of molecular compounds for use in molecular dynamics simulations. We exploit properties of practical input data, including bounded treewidth, and develop a dynamic programming based method.

In the final part, we introduce the crossing schedule optimization problem, which, given a set of parental genotypes, asks for an efficient way of crossing these and their offspring with the goal of arriving at a specified desired genotype. We introduce a mixed integer linear programming formulation for solving it.

Samenvatting

Dit proefschrift behandelt een aantal problemen uit de computationale biologie. Voor het oplossen van deze problemen passen we technieken toe afkomstig uit de combinatoriële optimalisering. We gebruiken het volgende schema hiervoor. Het startpunt is een combinatoriële formulering van het biologisch probleem. Deze verkrijgen we door het definiëren van de verzameling van toegestane oplossingen alsook een optimaliseringscriterium. Dit criterium is een functie die een waarde toekent aan elk element in de verzameling van toegestane oplossingen. Het doel is het vinden van een toegestane oplossing met een optimale waarde. Hiertoe analyseren we de complexiteit en de combinatoriële structuur van het probleem. Deze analyses leiden tot inzichten die we gebruiken in het ontwerpen van een algoritme dat praktisch efficiënt is. Tot slot onderzoeken we de biologische kwaliteit van de gevonden oplossingen voor praktische probleeminstanties.

De onderwerpen die we behandelen vallen onder de volgende categorieën: netwerken, modules en kruisingsschema's. We beginnen het eerste deel met een biologisch probleem afkomstig uit de vergelijkende netwerkanalyse. In dit vakgebied poogt men overeenkomsten en verschillen tussen biologische netwerken in kaart te brengen. De knopen in deze netwerken zijn biologische entiteiten zoals eiwitten, genen of transcriptiefactoren. De kanten beschrijven interacties tussen paren van knopen. Vaak zijn biologische netwerken verkregen onder verschillende condities, of zelfs afkomstig van verschillende biologische soorten. We lossen dit probleem op door het toepassen van Lagrange relaxatie. Het volgende onderwerp dat we behandelen is het voorspellen van eiwit-eiwit interacties gebruikmakend van het concept van co-evolutie. De onderliggende aanname is dat bewijs voor de gezamenlijk evolutie van de eiwitfamilies van twee eiwitten tevens een indicatie is voor een interactie tussen de twee eiwitten. Voor het oplossen van dit probleem hergebruiken we de wiskundige formulering voor netwerkalignering.

Het tweede deel van dit proefschrift gaat over de extractie van kleinere samenhangende deelnetwerken uit een groter invoernetwerk. We beginnen met het behandelen van het actieve module probleem: gegeven genexpressiedata en een eiwit-eiwit interactienetwerk, vind een samenhangend deelnetwerk wier knopen significant differentieel tot expressie komen. Voor het oplossen van dit probleem gebruiken we geheeltallige programmering door middel van een branch-and-cut schema. We interpreteren gevonden actieve modules door een zelf-ontwikkelde visualitietechniek waarmee we zowel annotatieverzamelingen van knopen als de topologie van de module tonen. Het volgende probleem dat we behandelen is een generalisatie van het

actieve module probleem naar verschillende biologische soorten. Vervolgens behandelen we een probleem over het partitioneren van een molecuul in ladingsgroepen. Dit probleem komt voor in het parametriseren van moleculaire structuren voor gebruik in moleculaire dynamica simulaties. We maken gebruik van eigenschappen van invoerdata, zoals constante graad en ontwikkelen een methode gebaseerd op dynamische programmering.

Het laatste deel behandelt het kruisingsschemaprobleem. Dit probleem heeft een toepassing in de zaadveredeling, waar men gegeven een verzameling oudergenotypes op zoek is naar een schema, bestaande uit kruisingen tussen de oudergenotypes en hun nageslacht, dat resulteert in een opgegeven gewenst genotype. We lossen dit probleem op door gebruik te maken van gemengd-geheeltallige programmering.

Acknowledgments

Over the past couple of years I've come to realize that science is a lot of fun! Especially because I've had the pleasure of working with like-minded people on very interesting and relevant problems. I'm indebted to many friends, colleagues and relatives for helping me grow personally and for the wonderful time that I've experienced. The words below, in Dutch and English, will no doubt fail to express the extent of my gratitude—but I will try nevertheless.

I start by thanking my advisors Jaap Heringa and Gunnar Klau for their excellent guidance and supervision. Jaap, I'm very grateful for the freedom that you've giving me in pursuing my research interests. I'm especially grateful for telling me to consider doing a PhD when I was in the Bioinformatics master's program—this was something that was not on my radar at all, and I'm very glad that I took your advice to heart! Gunnar, I'll always remember how much fun we had while drafting introductions to papers just a few days prior to the submission deadline. I've learned many things from you, from supervising students to writing papers and giving good talks. Your following sentence is something that I won't forget: 'There is always a deadline.' It reminds me of your working attitude in which you have successfully managed to find a great work-life balance—something that I hope to achieve as well!

Science is not a one man job. I've been lucky to have had the pleasure of working with many people.

- Stefan Canzar
- Thomas Hume
- Kasper Dinkla
- Inken Wohlers, Christine Staiger
- Leen Stougie
- Khaled Elbassioni
- Daan Geerke
- Rene Pool
- Alan Mark
- Martin Stroet

- Denise Kirschner
- Simeone Marino
- Daniela, Wurzburg people
- ...

Fun during PhD: CWI running team! Timo, I'm getting close to achieving my goal of running a sub 20 min 5k!

Students!

Tot slot ben ik mijn ouders heel dankbaar voor hun continue motivatie en belangstelling, van jongs af aan. Zonder hen zou ik nooit in staat zijn geweest om deze woorden te mogen schrijven. Shokran bezaf!

- Collaborators
- Supervisors
- Supervised students
- People at IBIVU
- People at LS
- Friends
- My parents
- Australia guys
- Michigan, for research stay in 2011.
- Martijn and Marceline
- Squash
- Wurzburg people + Daniella
- Everyone else
- Committee
- Jury BioSB YIA
- Marlies, accolade openen
- Stefan Canzar
- Bernd Brandt

Publications

M. El-Kebir†, L. Oesper†, H. Acheson-Field, B. J. Raphael. Reconstruction of clonal trees and tumor composition from multi-sample sequencing data. *Bioinformatics*, 31 (12):i62–i70, ISMB/ECCB 2015.

M. El-Kebir†, H. Soueidan†, T. Hume†, D. Beisser, M. Dittrich, T. Müller, G. Blin, J. Heringa, M. Nikolski, L. F. A. Wessels, G. W. Klau. xHeinz: An algorithm for mining cross-species network modules under a flexible conservation model. *Bioinformatics*, btv316, 2015.

M. El-Kebir and G. W. Klau. Solving the Maximum-Weight Connected Subgraph Problem to Optimality. Presented at the 11th DIMACS Challenge workshop, 4/5 Dec 2014, Providence (RI), U.S.A. *Submitted.*

K. Dinkla†, M. El-Kebir†, C.-I. Bucur, M. Siderius, M. J. Smit, M. A. Westenberg, and G. W. Klau. eXamine: Exploring annotated modules in networks. *BMC Bioinformatics*, 15(1):201, 2014.

M. El-Kebir†, B. W. Brandt†, J. Heringa, and G. W. Klau. NatalieQ: A web server for protein-protein interaction network querying. *BMC Systems Biology*, 8(1):40, 2014.

M. El-Kebir†, T. Marschall†, I. Wohlers†, M. Patterson, J. Heringa, A. Schönhuth, and G. W. Klau. Mapping proteins in the presence of paralogs using units of coevolution. *BMC Bioinformatics*, 14(Suppl 15):S18, 2013.

M. El-Kebir†, M. van der Kuip†, A. M .van Furth, and D. E. Kirschner. Computational modeling of tuberculous meningitis reveals an important role for tumor necrosis factor-α. *Journal of Theoretical Biology*, 382(C):43–53, 2013.

S. Canzar†, M. El-Kebir†, R. Pool, K. Elbassioni, A. K. Malde, A. E. Mark, D. P. Geerke, L. Stougie, and G. W. Klau. Charge Group Partitioning in Biomolecular Simulation. *Journal of Computational Biology*, 20(3):188–198, Mar. 2013.

S. Canzar†, M. El-Kebir†, R. Pool, K. M. Elbassioni, A. K. Malde, A. E. Mark, D. P. Geerke, L. Stougie, and G. W. Klau. Charge group partitioning in biomolecular simulation. In *Research in Computational Molecular Biology, RECOMB 2012, Barcelona, Spain, April 21–24, 2012*, pages 29–43, 2012.

S. Canzar[†] and M. El-Kebir[†]. A mathematical programming approach to marker-assisted gene pyramiding. In T. M. Przytycka and M.-F. Sagot, editors, *WABI 2011*, volume 6833 of *Lecture Notes in Computer Science*, pages 26–38. Springer, 2011.

M. El-Kebir, J. Heringa, and G. W. Klau. Lagrangian relaxation applied to sparse global network alignment. In *Pattern Recognition in Bioinformatics, PRIB 2011, Delft, The Netherlands, November 2–4, 2011*, pages 225–236, 2011.

S. Marino, M. El-Kebir, and D. E. Kirschner. A hybrid multi-compartment model of granuloma formation and T cell priming in Tuberculosis. *Journal of Theoretical Biology*, 280(1):50–62, Jul. 2011.

M. Fallahi-Sichani, M. El-Kebir, S. Marino, D. E. Kirschner, and J. J. Linderman. Multiscale computational modeling reveals a critical role for TNF-Receptor 1 dynamics in Tuberculosis granuloma formation. *The Journal of Immunology*, 186(6):3472–3483, Mar. 2011.

[†]joint first authorship

Curriculum vitae

Mohammed El-Kebir was born on September 18th, 1985 in Amsterdam. From 1997 to 2003 he attended the Dr. Mollercollege in Waalwijk. In 2003, he started studying Computer Science and Engineering at the Eindhoven University of Technology. In 2009, Mohammed obtained a Master's degree in Computer Science and Engineering at the Eindhoven University of Technology. A year later he obtained a second Master's degree in Bioinformatics at the VU University Amsterdam. Both degrees were obtained cum laude. In 2010, Mohammed joined the Life Sciences group at Centrum Wiskunde & Informatica and the Bioinformatics group at the VU University Amsterdam as a joint PhD student. Currently, he works as a postdoctoral researcher at Brown University.