Integrative Biology

Cite this: *Integr. Biol.*, 2012, **4**, 756–764

www.rsc.org/ibiology

PAPER

Efficient key pathway mining: combining networks and OMICS data†

Nicolas Alcaraz, Tobias Friedrich, Timo Kötzing, Anton Krohmer, Joachim Müller, Josch Pauling and Jan Baumbach*

Received 11th October 2011, Accepted 2nd January 2012
DOI: 10.1039/c2ib00133k

Systems biology has emerged over the last decade. Driven by the advances in sophisticated measurement technology the research community generated huge molecular biology data sets. These comprise rather static data on the interplay of biological entities, for instance protein–protein interaction network data, as well as quite dynamic data collected for studying the behavior of individual cells or tissues in accordance with changing environmental conditions, such as DNA microarrays or RNA sequencing. Here we bring the two different data types together in order to gain higher level knowledge. We introduce a significantly improved version of the KeyPathwayMiner software framework. Given a biological network modelled as a graph and a set of expression studies, KeyPathwayMiner efficiently finds and visualizes connected sub-networks where most components are expressed in most cases. It finds all maximal connected sub-networks where all nodes but k exceptions are expressed in all experimental studies but at most l exceptions. We demonstrate the power of the new approach by comparing it to similar approaches with gene expression data previously used to study Huntington's disease.
In addition, we demonstrate KeyPathwayMiner's flexibility and applicability to non-array data by analyzing genome-scale DNA methylation profiles from colorectal tumor cancer patients. KeyPathwayMiner release 2 is available as a Cytoscape plugin and online at http://keypathwayminer.mpi-inf.mpg.de.

1 Introduction

Over the last few decades, extensive usage of high-throughput technologies allowed for producing vast amounts of data in the field of molecular biology. The so-called next generation sequencing technologies have opened the way for cheap and fast DNA sequencing of many organisms. More than 1600 microbial whole-genome DNA sequences are available for download from NCBI. To date, 37 completely sequenced eucaryotic genomes are available; 480 more projects are in the assembly stage, and ∼700 more projects are registered to be in progress.[21] However, knowing an organism's genetic code is only the first step in understanding survival, reproduction and adaptation to changing environmental conditions.[5] Various molecular mechanisms control and fine tune the interplay of biological entities of all types, resulting in biological networks that are modelled as graphs. Nodes correspond to these entities, such as genes or proteins, and edges represent interactions between them. The emerging networks, such as gene regulatory networks or protein–protein interaction networks, are the basis for directed systems biology studies. In parallel, besides these data regarding the general possibility of an interplay between genes, proteins, metabolites, RNAs, *etc.*, we use the so-called OMICS technology

Max Planck Institute for Informatics—Computational Systems Biology, Saarbrucken 66123, Germany. E-mail: jbaumbac@mpi-inf.mpg.de

† Published as part of an iBiology themed issue entitled "Computational Integrative Biology". Guest Editor: Prof. Jan Baumbach.

Insight, innovation, integration

With KeyPathwayMiner we present a computational method that allows for the combined analysis of biological networks together with large-scale OMICS data, such as gene expression data. The software is integrated into Cytoscape as plug-in and requires minimal user input. Given a network, a PPI network for instance, KeyPathwayMiner extracts all maximal connected sub-networks where all genes/proteins but k are expressed in all patients but l. We demonstrate the efficiency and flexibility of our approach by computing disease-specific, dysregulated PPI key pathways for Huntington's disease (gene expression data) and colorectal cancer (methylation data) on a standard desktop PC.

to measure their expression, methylation, ubiquitination, *etc.* Typically, these analyses are performed to identify biological changes in response to external or internal stimuli, perturbations or diseases.

Each kind of data type, the rather static network data as well as the OMICS data, is usually studied in isolation. We may study statistical network properties, such as over-represented network patterns,[12] node degree distributions,[4] or network centrality nodes.[3] More dedicated approaches can be used to identify, for instance, protein complexes in protein–protein interaction (PPI) networks.[26] On the other hand, numerous methods exist for high-throughput OMICS data analysis, such as cancer sub-typing based on the identification of genome-wide gene expression similarities[25] to give only one of many examples. The number of available data sets is growing rapidly. In GEO, the Gene Expression Omnibus, we find transcriptomics data for $>630\,000$ samples.[10] To give an impression about the available network data, the PSICQUIC platform integrates 16 interactome databases and covers more than 16 million protein–protein interactions of various evidence levels.[2]

Here, we aim to combine both kinds of data types in order to find key pathways, *i.e.* well connected sub-networks where most of the nodes are active/expressed/methylated/*etc.* in most cases/measurements/patients/*etc.* In 2008, Ulitsky *et al.* presented an approach that integrates gene expression data with PPI networks. They solve the hard problem of finding *minimal* connected subnetworks with *at least* k nodes differentially expressed in all but l analyzed samples by using the CUSP heuristic (covering using shortest paths).[24] The variables k and l are given by the user. They may be seen as parameters for controlling the amount of accepted noise in the network (k) as well as in the expression data (l). Both k and l impact the size of the reported key pathways: the smaller k and l, the less noise we allow, and the smaller are the reported sub-networks. Since finding *maximal* connected sub-networks that maximize a certain scoring function based on all nodes in the result set is NP-hard,[15] Ulitsky *et al.* search for *minimal* connected subgraphs with at least k nodes. This allows them to pre-process each node individually by excluding all genes that are not expressed in all but l cases. This causes a practical problem for the end-user if no solution for a given k exists, the algorithm needs to be re-started with a smaller value of k. However, suitable values for k are generally unknown *a priori*. Therefore, we recently introduced an adapted interpretation of finding key pathways with given k and l parameters.[1] In our previous work, we aimed to detect all *maximal* connected subnetworks where *all but k* nodes are differentially expressed in all but l analyzed samples. Here, the user does not need to know good values for k in advance since KeyPathwayMiner reports all subgraphs where all nodes but k are "dysregulated" in all but l gene expression samples. We compared KeyPathwayMiner 1.0 with the CUSP method and three other approaches. We used the same data and the same evaluation scheme that Ulitsky *et al.* previously utilized for demonstrating the power of their CUSP method. In terms of accuracy, we were able to show that KeyPathwayMiner keeps up with or outperforms CUSP and the other approaches. However, our worst case run time complexity is $|V|^k$ (note that it is generally independent of l). Hence, in ref. 1 we used a basic Ant Colony Optimization (ACO) based heuristic for tackling this problem. Unfortunately, this sometimes may lead to sub-optimal solutions and a slow running time, especially for larger l-values. KeyPathwayMiner 1.0 was applicable only to smaller values of k and l on a standard desktop PC and to relatively small PPI networks (see ref. 1 for a detailed description of the previous release). The KeyPathwayMiner 1.0 implementation came with a very basic user interface and did not support parallel computing for speeding up the pathway extraction process.

Here, we present KeyPathwayMiner 2.0. We describe three algorithmic approaches that (1) solve the computational problem exactly or (2) significantly improve approximation accuracy with only slightly increased running time. The new KeyPathwayMiner version (3) can run in parallel on multi-processor machines and (4) ensures flexibility, *i.e.* applicability to many kinds of networks as well as "expression" data types. Afterwards, we introduce two data sets that we will use to demonstrate the power of KeyPathwayMiner. First, we apply our tool to the same evaluation data described in ref. 1 and 24 (gene expression data of Huntington's disease patients) in order to compare KeyPathwayMiner 2.0 against the previous version 1.0 and against other approaches, *e.g.* CUSP[24] and jActiveModules.[15] Finally, we demonstrate KeyPathwayMiner's flexibility and applicability to non-gene expression data by analyzing genome-scale DNA methylation profiles from colorectal cancer patients.

2 Methods and data

2.1 Algorithms

In this section we give three algorithms (one greedy, one optimal, one based on ant colony optimization). All three algorithms solve the same formal graph problem, which is an abstraction of the problem of finding key pathways (as described above) to the level of labeled graphs. Note that we describe these algorithms as returning only the best solution found, while our implementation of these algorithms will output several of the top solutions found.

Definition 2.1. Let $G = (V, E, d)$ be a labeled graph on a set of vertices V, edges (sets of exactly two vertices) E and a labeling function d: $V \to \mathbb{N}$. Let $k, \ell \in \mathbb{N}$. Then the (k, ℓ)-*KeyPathway problem* asks for a set $U \subseteq V$ of maximal cardinality such that

- U is connected;
- the number of elements $u \in U$ with $d(u) \leq \ell$ is at most k.

We call any set U such that the above two bullets are fulfilled a (k, ℓ)-*component*. Any vertex $v \in V$ with $d(v) \leq \ell$ is called an *exception vertex*.

The interpretation of the graph, the labels, and the (k, ℓ)-component is as follows. Vertices of the graph represent biological entities (here, genes or proteins); edges correspond to an interplay between two entities, *e.g.* a protein–protein interaction; and the labels on a vertex v correspond to the number of cases where v is active/expressed/methylated/*etc.*

2.1.1 Preprocessing.

In order to derive efficient algorithms, we apply a preprocessing stage (the same for all algorithms). Essentially, we construct an auxiliary labeled graph to decrease the problem size and to help direct the algorithms to better regions of the search space. The main idea for this kind of preprocessing was derived from ref. 1.

Definition 2.2. Given a labeled graph $G = (V, E, d)$ and $\ell \in \mathbb{N}$, we let $C(G, \ell)$ be the ℓ-component graph derived from G as follows. The vertex set of $C(G, \ell)$ are all the exception vertices of G. Two exception vertices are connected by an edge in $C(G, \ell)$ if there is a path between them in G which does not use exception vertices as inner vertices.

For any set $U \subseteq V$ of exception vertices we define $S(U)$ as the set of all vertices $v \in V$ reachable in G from an element of U without passing by an exception vertex not in U. Intuitively, we just need to choose a connected set of exception vertices U in $C(G, \ell)$ with k vertices to get a (k, ℓ)-component of G, namely $S(U)$.

2.1.2 Greedy algorithm.

We define the following greedy algorithm on $C(G, \ell)$. For every vertex u, we iteratively construct a set W_u starting with $W_u = \{u\}$. In every iteration we add to W_u a vertex v from $C(G, \ell)$ which is adjacent (in $C(G, \ell)$) to W_u and which maximizes $|S(W_u \cup \{v\})|$. We stop the iterations when $|W_u| = k$. Return $S(W_u)$ of maximal size found for some u.

2.1.3 Exact branch and bound algorithm.

We define the following algorithm for the computation of the optimal solution by employing a branch and bound method. As a lower bound, the best current solution is used. To compute an upper bound, the algorithm proceeds as follows. If, for some x, the algorithm has a partial solution which already has $k - x$ exception vertices, then the algorithm determines all possible new exception vertices that can be reached from the current subgraph in x steps, takes the x nodes with the highest weights and adds those together. This is an upper bound for the fitness this partial solution can achieve.

Using these two bounds, the algorithm finds the optimal solution by an (otherwise) exhaustive search. This branch and bound method guarantees optimality of the solution.

2.1.4 Ant colony algorithm.

In ref. 23 a search heuristic was proposed for solving the traveling salesperson problem, based on ant colony optimization (ACO), called MMAS (Max–Min Ant System). This algorithm (or variations thereof) has undergone thorough theoretical investigation.[18,27]

We employ a variant of MMAS which works as follows. For every vertex u of $C(G, \ell)$ we construct a solution containing u; finally, we pick the best solution found. For each such u we iteratively try to find better solutions until some termination criterion is met; in our implementation we use a fixed number of iterations as the termination criterion. In each iteration we create a fixed number a of new solutions, see Chart 1.

The construction of a new candidate solution for a target graph G works as follows. We imagine an artificial ant performing a random walk on $C(G, \ell)$, at each step choosing vertices of a new candidate solution. The construction terminates when k vertices have been chosen.

In each step of its random walk on $C(G, \ell)$, we want the ant to choose a vertex v in $C(G, \ell)$ adjacent to one of the previously chosen vertices with a probability based on the *pheromone value* $\tau(v)$ and on the *heuristic value* $\eta(v)$ of that vertex. Pheromone values represent the memory of the ACO algorithm about the quality of previously sampled tours and direct the search towards promising areas; we will say more about pheromones later. The heuristic information on a vertex v is the number of new vertices in G that can be reached when including v, *i.e.* $\eta(v) = |S(U \cup \{v\})| - |S(U)|$; we use the heuristic value to bias the algorithm to favor promising vertices which make many new vertices reachable. Our ACO algorithm takes two parameters α and β; when choosing a new edge, we sample proportionally to the pheromone value to the power of α times the heuristic value to the power of β.

```
function $MMAS$ on $G = (V,E,d)$, $k$ and $\ell$ is
    foreach $u \in C(G,\ell)$ do
        $\tau(v) \leftarrow |V|/2$, for all $v \in V$;
        while termination criterion not met do
            for $i = 1$ to $a$ do
                $W_i \leftarrow$ construct$(u)$;
            $\tau \leftarrow$ update$(\tau, (W_i)_i)$;
        Let $W_u$ be the best solution found for this $u$;
    return $W_u$ for some $u$ with $|S(W_u)|$ maximal;
```

Chart 1 The algorithm MMAs.

```
function construct$(u)$ based on $\tau, \eta, \alpha, \beta$ is
    $W \leftarrow \{u\}$;
    for $i = 1$ to $k$ do
        $R \leftarrow \sum_{v \in N(W)} \tau(v)^\alpha \cdot \eta(v)^\beta$;
        Choose one neighbor $v \in N(W)$ where the probability of selecting
        any fixed $v \in N(W)$ is $\frac{\tau(z)^\alpha \cdot \eta(z)^\beta}{R}$;
        $W \leftarrow W \cup \{v\}$;
    return $W$;
```

Chart 2 The algorithm construct.

We use a procedure construct based on the pheromones τ as given in Chart 2.

Pheromones are the values $\tau(v)$ associated with the vertices $v \in C(G, \ell)$ and range between 1 and $|V|$; they are initialized as $|V|/2$. After each iteration, the pheromones will be updated; we now describe the update procedure. This procedure depends on the *evaporation factor* ρ ($0 \le \rho \le 1$, a parameter of the ACO algorithm); small values of ρ (close to 0) indicate low evaporation and small changes to pheromones per iterations; large values (close to 1) indicate fast changes. The evaporation factor ρ is a tunable parameter of the algorithm.

At the beginning, all pheromones are decreased by multiplication with $(1 - \rho)$; this corresponds to the evaporation of old pheromone, deposited in previous iterations.

Next, for each solution W_i found, we add a value of $|S(W_i)| \cdot \rho$ to the pheromone value of each vertex in W_i; this way the algorithm is biased to choose those vertices which are included in solutions leading to a large (k, ℓ)-component.

The Max–Min Ant System derives its name from maximal and minimal pheromone values that can be attained. At the end of the update procedure, all pheromone values below 1 are increased to be 1, and all values above $|V|$ are decreased to be $|V|$.

2.2 Data sources

2.2.1 Evaluation network data—Ulitsky *et al.*

In order to evaluate the newly developed algorithms and compare them to previous methods, we tested them using the same PPI network from Ulitsky *et al.*,[24] which consists of 7384 nodes corresponding to Entrez Gene identifiers and 23 462 interactions based mostly on small scale experiments and obtained from several interaction databases. The network as well as more detailed information about this data source can be obtained from the CUSP web site http://acgt.cs.tau.ac.il/clean. We will use this network for evaluation with the below-described Huntington's disease gene expression data set, as previously done in ref. 1 and 24.

2.2.2 New network data—HPRD. To complement the network compiled by Ulitsky and colleagues, we merged it with the latest protein–protein interactions contained in the Human Protein Reference Database.[20] HPRD is a protein interaction repository consisting of high quality human curated interaction data sets. Their latest version (release 9) consists of 9672 proteins and 39 194 interactions. Computing the union of both results in a network consisting of 9867 proteins and 41 609 interactions. We will use this extended network for analyzing the colon cancer methylation data, as described below.

2.2.3 Gene expression data set—Huntington's disease. Huntington's disease (HD) is a degenerative neurological disorder caused by a genetic defect on chromosome number four which encodes a mutated version of the huntingtin (*HTT*) protein. Studies have shown that mutant huntingtin interferes with the function of widely expressed transcription factors, suggesting that gene expression may be altered in a variety of tissues in HD and may be a good target for "systems" methods that take into account its interacting partners.

We tested the pathway extraction methods on the same Huntington's disease data sets used in ref. 1 and 24. The expression data sets, with GEO accession number GSE3790, were obtained using oligonucleotide arrays[14] consisting of 32 unaffected control samples and 38 affected samples taken from the caudate nucleus region of the brain. To determine differential expression and compute the indicator matrix, the same thresholds and *p*-values were used as in ref. 24.

2.2.4 Differential methylation data set—colon cancer. Colorectal cancer (CRC) is a complex form of cancer that arises from the accumulation of several genetic and epigenetic changes. Recent efforts have been focusing on trying to integrate these different sources of biological information in order to extract meaningful groups of regulatory interactions. Complex diseases such as CRC are good targets of study from a "systems" point of view, where both types of biological data can be integrated with network data to extract affected pathways.

KeyPathwayMiner was tested on a comprehensive genome-scale DNA methylation profiling of 125 colorectal tumors and 29 adjacent normal tissues.[13] Datasets are deposited in GEO and can be found by their accession number GSE25062.

3 Results and discussion

3.1 Evaluation

Here we used two data sets to compare the running times and performance of all algorithms. The first set consists of the same data and the same evaluation scheme as in ref. 1 and 24. The second sets consist of the CRC methylation data sets consisting of 125 affected samples, which are mapped to the extended graph, as described above. All computations have been performed with our KeyPathwayMiner implementation on a standard desktop PC (Intel 2.8 GHz Quad Core). We show that at least for those cases where we can compute the exact solution with the above introduced branch and bound algorithm, both heuristics, ACO as well as the greedy algorithm, find the correct solutions in all cases resulting in an increased accuracy compared to the previous release. Afterwards, we demonstrate this empirically with the HD data set. Finally, we analyze the run time performance on the CRC methylation data sets and the extended network.

3.1.1 Algorithms. First we compared the running times and sizes of the largest subnetwork extracted with all algorithms. For $k > 4$ (HD data) and $k > 3$ (CRC data) the exact branch and bound (BB) algorithm was not able to recover a solution in a reasonable time. Both the new ACO algorithm and the greedy strategy were able to find the largest pathway computed by the BB algorithm up to those k values. In most of the cases ACO terminated faster than greedy; however, for larger values of k the greedy slightly outperforms the new ACO algorithm by recovering larger pathways (see Fig. 1). The ACO algorithm from KeyPathwayMiner 1.0 (which we will refer to as ACO 1.0) recovers pathways faster than all algorithms for smaller values of l, but it gets increasingly slow for larger numbers of accepted case exceptions. However, ACO 1.0 converges to suboptimal solutions and recovers smaller key pathways than the new algorithms for all parameter values, k as well as l (see Fig. 2).

Additionally, both the greedy and the new ACO algorithm were parallelized in order to take advantage of multi-processor architectures. Fig. 3 and 4 illustrate the substantial decrease in running times on two Intel Quad Core CPUs for both algorithms when compared to the old KeyPathwayMiner 1.0 single threaded versions.

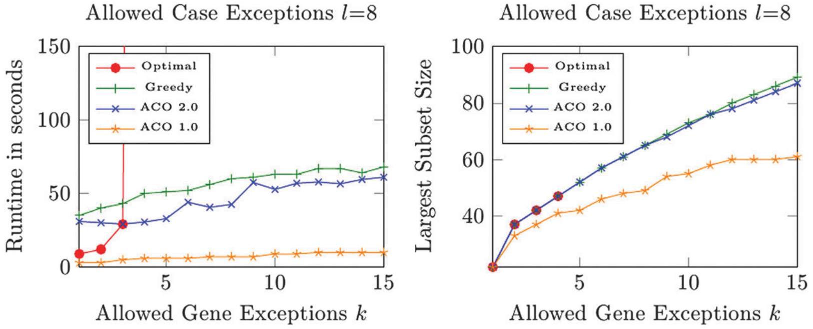

Fig. 1 Performance results of the algorithms on the HD data sets for $l = 8$ case exceptions and varying values of k (gene/protein exceptions). The previous ACO algorithm (ACO 1.0) converges faster than the new one (ACO 2.0) but to suboptimal solutions.

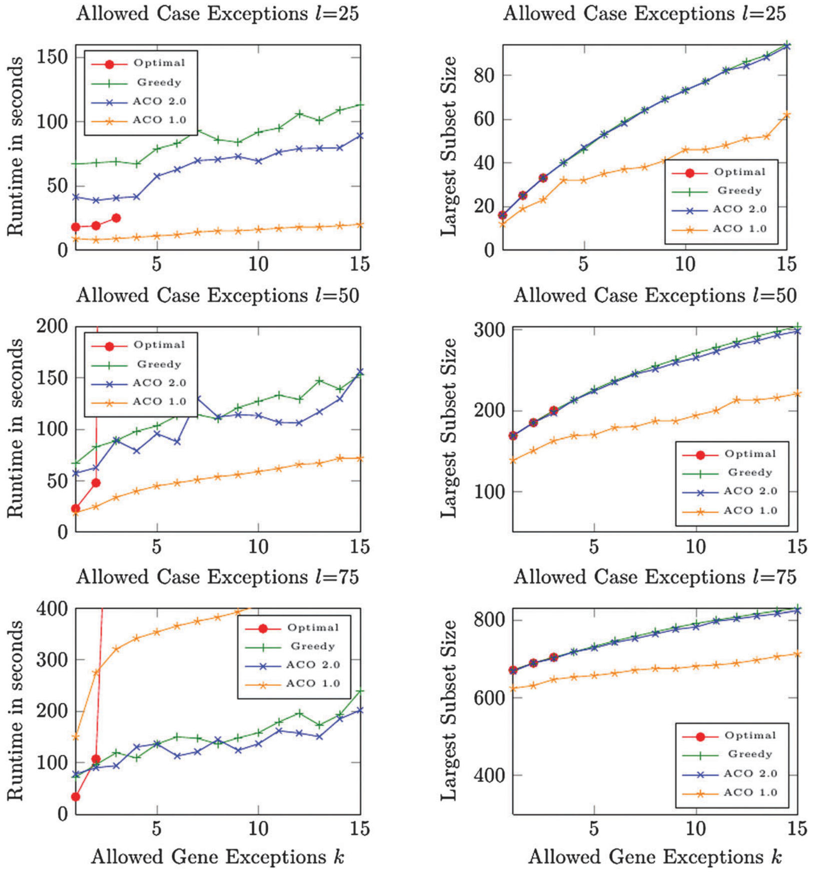

Fig. 2 Performance results of the algorithms for the colon cancer methylation data sets for varying numbers of gene exceptions *k* and for case exception values of *l* between 20% (top row) and 60% (bottom row).

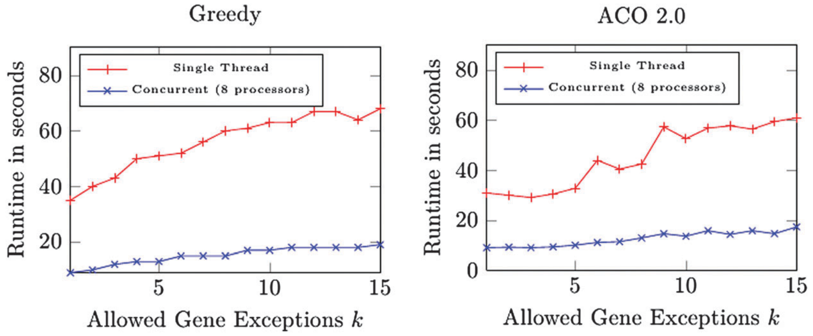

Fig. 3 Runtime comparison of the greedy and ACO algorithms, single-threaded *vs.* multi-processor, on the HD data sets for $l = 8$ and varying values of *k*.

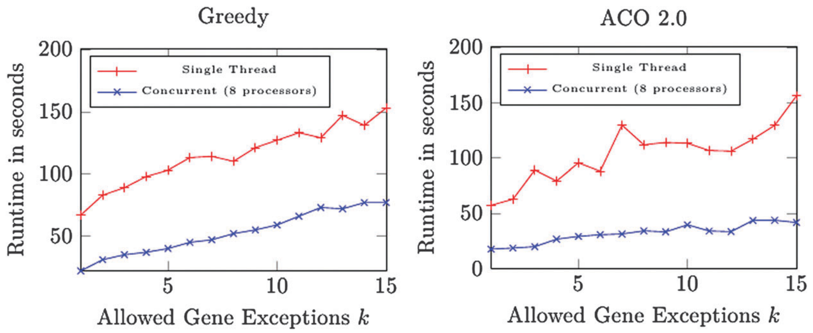

Fig. 4 Runtime comparison of the greedy and ACO algorithms, single-threaded *vs.* multi-processor, on the CRC data sets for $l = 50$ and varying values of *k*.

Table 1 Comparison of the largest affected pathways found for $k = 2$ and $l = 8$ by KeyPathwayMiner 1.0 (KPM 1.0), KeyPathwayMiner 2.0 (KPM 2.0) and other pathway extraction methods such as CUSP,[24] GiGA,[8] jActiveModules[15] and the top 34 active genes with the most significant *t*-scores

	KPM 1.0 (ACO)	KPM 2.0 (ALL)	CUSP	GiGA	jActive-Modules	*t*-test top
Number of genes	33	37	34	34	282	34
Contains *HTT*?	Yes	Yes	Yes	No	No	No
HD modifiers	7	8	7	3	12	2
KEGG HD pathway	8	8	4	0	4	0
Calcium pathway	5	5	6	5	10	3

Table 2 Comparison of the largest affected pathway with the most number of overlap with relevant genes found for $k = 8$ and $l = 8$ by both KeyPathwayMiner versions (greedy and ACO)

	KPM 1.0 (ACO)	KPM 2.0 (ACO)	KPM 2.0 (GREEDY)
Number of genes	49	65	65
Contains *HTT*?	Yes	Yes	Yes
HD modifiers	7	8	8
KEGG HD pathway	10	11	11
Calcium pathway	7	7	7

3.1.2 Huntington's disease data. For $k = 2$, $l = 8$ exceptions we find that the optimal solution consists of 37 genes (see Table 1), four more genes than in the solution found in the previous KeyPathwayMiner version where one of these genes *CTNNB1* is an *HTT* modifier (Fig. 5).

When raising the number of gene exceptions to $k = 8$, the largest pathways recovered with both the new ACO and greedy algorithms have 65 genes (Table 2). One of these pathways (see Fig. 6) contains the *PLCB1* and *GNAQ* genes which are both part of the HD KEGG and calcium signaling pathways. In addition, *TP53* is present in this pathway, also part of the HD KEGG pathway and which has been found to be hyperactive during HD.[11]

The genes *HTT*, *CTNNB1*, *GNAQ* and *TP53* are all "exception" genes, having more than eight cases where they are not differentially expressed. This supports the idea that exception genes may be important genes that are generally overlooked by most other methods, although they often connect highly differentially expressed regions within the biological network.

3.2 Analysis of colon cancer DNA methylation profiles

Colorectal tumors with a CpG island methylator phenotype (CIMP) exhibit a high frequency of cancer-specific DNA hypermethylation. DNA hypermethylation of promoter CpG islands has been associated with transcriptional gene silencing and can cooperate with other genetic mechanisms to alter key signaling pathways. In ref. 13 genes were clustered into different subgroups using DNA methylation profiles of 125 affected patients with CRC. It is known that CRC can originate from key mutations in CIMP-related tumor genes such as *BRAF*, *KRAS*, *TP53*, *APC* and *PIK3CA*, where changes in each of these genes can provoke alterations on different pathways. Also, hypermethylation of *MLH1* has been associated with development of sporadic CRC with microsatellite instability (MSI) (Fig. 7).

The samples collected in ref. 13 are quite heterogeneous where the mentioned genes do not show significant differential methylation in a large fraction of the cases, and setting $l = 25$ (exactly a maximum of one fifth of cases) makes them all exception genes. Although they appear in some of the largest pathways found (see Table 3), even when setting the maximum allowed gene exceptions to $k = 8$, no pathways were recovered containing at least two of these genes, confirming that CRC originating from different mutations can affect different pathways, which highlights the importance of not reporting exclusively the largest extracted key pathway.

3.3 Implementation. KeyPathwayMiner was implemented as a Cytoscape[6,9,22] (http://www.cytoscape.org) plugin which is now freely available for download through the Cytoscape's plugin manager or through the project's web site.

Once users install the plugin, they are able to upload their computed indicator matrix from their own differential expression

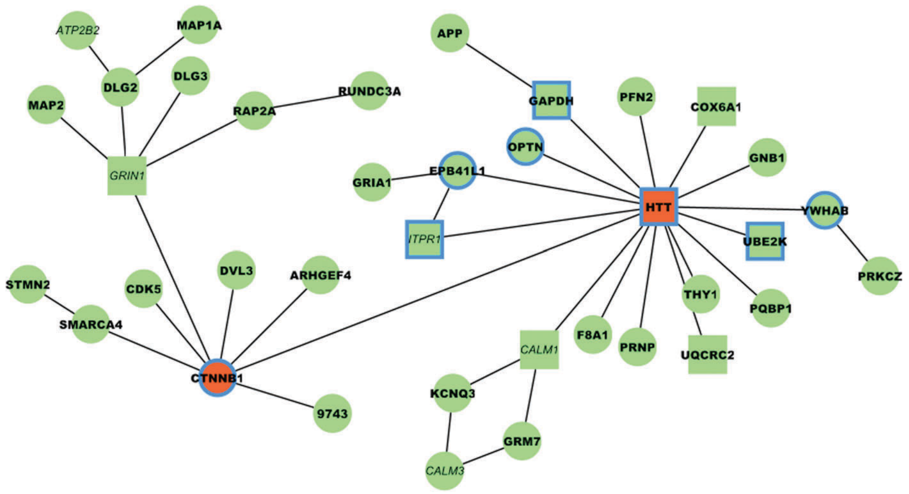

Fig. 5 Largest subnetwork found for $k = 2$. Red nodes represent exception nodes, squared nodes are genes also reported as part of the Huntington's disease KEGG pathway, nodes with blue borders are *HTT* modifiers, and nodes with italic font are part of the calcium signaling pathway.

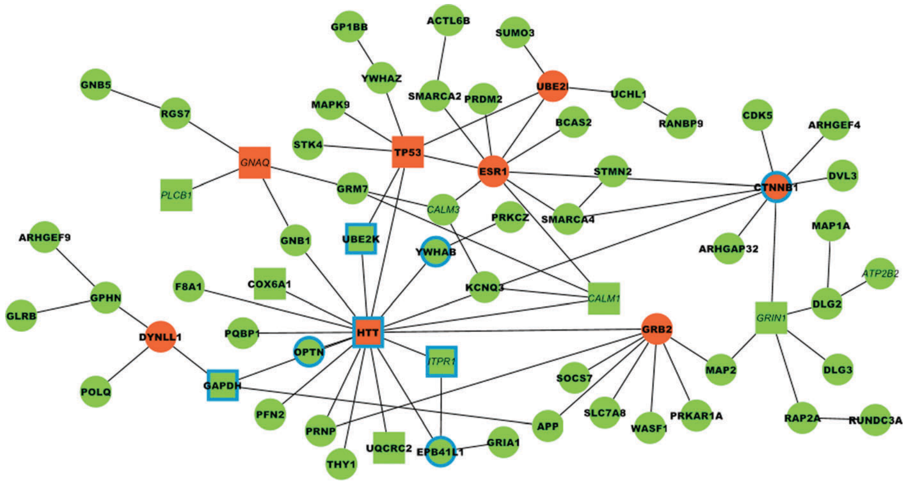

Fig. 6 Largest subnetwork found for $k = 8$ and $l = 8$ by using the new ACO algorithm. Red nodes represent exception nodes, squared nodes are genes also reported as part of the Huntington's disease KEGG pathway, nodes with blue borders are *HTT* modifiers, and nodes with italic font are part of the calcium signaling pathway.

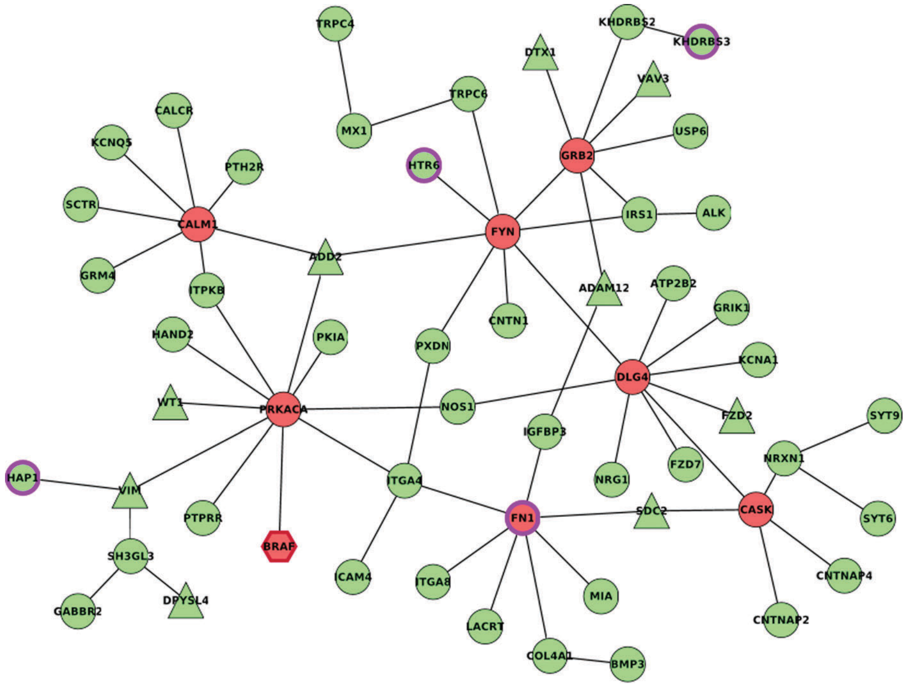

Fig. 7 Largest subnetwork found containing the *BRAF* gene for $k = 8$ and $l = 25$ using the greedy algorithm. Red nodes represent exception nodes, triangle nodes are hypermethylated genes that also show significant decrease in expression levels and nodes with purple border are genes with promoters classified as CIMP.

analysis and map it to an interaction network. Since the indicator matrix is highly application specific, we provide details for gene expression data in ref. 1.

For parameter adjustment, we suggest that l be set in proportion to the total number of differentially expressed cases. The higher the difference between control and affected samples the more restrictive (lower) values should be chosen in order to filter out genes and recover smaller but more focused networks. In contrast, if the studies contain very few differentially expressed cases, then l should be set to larger values to avoid recovering very small subnetworks with little information. As k determines the network size it is much more difficult to predict. We suggest setting it after l has been chosen. Once the rate of "exception" to "non-exception" genes is known, then k should be proportional to this number. With increasing percentage of exception genes, then k can be slowly increased in order to find pathways with genes connecting highly affected regions. If the number of "exception" genes is too low, then k should be decreased to prevent recovering very large subnetworks difficult to analyze.

The new KeyPathwayMiner consists of a renovated user interface where users can choose from the three implemented algorithms (greedy, ACO or BB algorithm) presented in this work. Both greedy and ACO are also implemented to take

Table 3 Key pathways recovered for $k = 8$, $l = 25$ containing relevant genes to CRC tumorigenesis, with size, number of genes with promoters classified as CIMP, and number of genes showing hypermethylation in promoters while also showing a significant decrease in expression levels according to ref. 13

Contains (exclusively)	Size	Contained promoters classified as CIMP	Contained hypermethylated genes with significant decrease in expression levels
TP53	62	5	9
KRAS	58	3	10
BRAF	56	4	10
APC	59	3	9
PIK3CA	56	3	7
MLH1	56	5	8

advantage of multi-core architectures and run concurrently to produce faster results.

Once the pathway extraction process is completed, the resulting key pathways are shown in the form of a table where users can sort them by different criteria, such as average number of differentially expressed cases. A table lists all the nodes in the graph, showing the number of pathways that contain the selected node. Users are also able to create pathway views from the table to perform further analysis with other network analysis tools included in Cytoscape.

4 Conclusion

In this paper we presented KeyPathwayMiner 2.0, an assembly of algorithmic approaches for extracting sub-networks from biological networks where all nodes but k are active in all studied cases but l. The major advantage over other methods is the interpretability of the extracted pathways and the two intuitive parameters. We described three new algorithmic approaches, an improved user interface and software implementation, and a new application of KeyPathwayMiner to methylation data. To sum up, our main contributions with the new KeyPathwayMiner version are

- We defined a precise algorithm solving the computationally intense problem exactly.
- With an improved ACO algorithm and a new greedy heuristic we significantly increased accuracy.
- Our new implementation comes with a parallel compute architecture that allows for using multi-processor computers.
- And finally, we demonstrated the applicability of KeyPathwayMiner 2.0 to other kinds of "expression" data types, namely such as genome-wide colorectal cancer methylation.

In the future we plan to extend our approach to directed networks. Furthermore, we aim to integrate the method into CoryneRegNet,[7,19] MycoRegNet[16] and RhizoRegNet.[17] We will also provide computation on a stand-alone web server and continue improving the Cytoscape front-end (current plug-in implementation: KeyPathwayMiner 2.1, Dec 21st, 2011).

Acknowledgements

JB, JM, JP, NA and TF are grateful for financial support from the Cluster of Excellence for Multimodal Computing and Interaction (MMCI), Germany. TF and TK received support from the German Research Foundation (DFG, grant NE 1182/5-1).

References

1 N. Alcaraz, H. Kucuk, J. Weile, A. Wipat and J. Baumbach, KeyPathwayMiner—detecting case-specific biological pathways by using expression data, *Internet Math.*, 2011, **7**(4), 299–313.

2 B. Aranda, H. Blankenburg, S. Kerrien, F. S. L. Brinkman, A. Ceol, E. Chautard, J. M. Dana, J. De Las Rivas, M. Dumousseau, E. Galeota, A. Gaulton, J. Goll, R. E. W. Hancock, R. Isserlin, R. C. Jimenez, J. Kerssemakers, J. Khadake, D. J. Lynn, M. Michaut, G. O'Kelly, K. Ono, S. Orchard, C. Prieto, S. Razick, O. Rigina, L. Salwinski, M. Simonovic, S. Velankar, A. Winter, G. Wu, G. D. Bader, G. Cesareni, I. M. Donaldson, D. Eisenberg, G. J. Kleywegt, J. Overington, S. Ricard-Blum, M. Tyers, M. Albrecht and H. Hermjakob, PSICQUIC and PSISCORE: accessing and scoring molecular interactions, *Nat. Methods*, 2011, **8**(7), 528–529.

3 Y. Assenov, F. Ramírez, S.-E. Schelhorn, T. Lengauer and M. Albrecht, Computing topological parameters of biological networks, *Bioinformatics*, 2008, **24**(2), 282–284.

4 S. Balaji, L. M. Iyer, L. Aravind and M. Madan Babu, Uncovering a hidden distributed architecture behind scale-free transcriptional regulatory networks, *J. Mol. Biol.*, 2006, **360**(1), 204–212.

5 J. Baumbach, On the power and limits of evolutionary conservation—unraveling bacterial gene regulatory networks, *Nucleic Acids Res.*, 2010, **38**(22), 7877–7884.

6 J. Baumbach and L. Apeltsin, Linking cytoscape and the corynebacterial reference database coryneregnet, *BMC Genomics*, 2008, **9**, 184.

7 J. Baumbach, A. Tauch and S. Rahmann, Towards the integrated analysis, visualization and reconstruction of microbial gene regulatory networks, *Briefings Bioinf.*, 2009, **10**(1), 75–83.

8 R. Breitling, A. Amtmann and P. Herzyk, Graph-based iterative group analysis enhances microarray interpretation, *BMC Bioinf.*, 2004, **5**(1), 100.

9 M. S. Cline, M. Smoot, E. Cerami, A. Kuchinsky, N. Landys, C. Workman, R. Christmas, I. Avila-Campilo, M. Creech, B. Gross, K. Hanspers, R. Isserlin, R. Kelley, S. Killcoyne, S. Lotia, S. Maere, J. Morris, K. Ono, V. Pavlovic, A. R. Pico, A. Vailaya, P.-L. Wang, A. Adler, B. R. Conklin, L. Hood, M. Kuiper, C. Sander, I. Schmulevich, B. Schwikowski, G. J. Warner, T. Ideker and G. D. Bader, Integration of biological networks and gene expression data using Cytoscape, *Nat. Protoc.*, 2007, **2**(10), 2366–2382.

10 R. Edgar, M. Domrachev and A. E. Lash, Gene Expression Omnibus: NCBI gene expression and hybridization array data repository, *Nucleic Acids Res.*, 2002, **30**(1), 207–210.

11 P. Giuliano, T. De Cristofaro, A. Affaitati, G. M. Pizzulo, A. Feliciello, C. Criscuolo, G. De Michele, A. Filla, E. V. Avvedimento and S. Varrone, DNA damage induced by polyglutamine-expanded proteins, *Hum. Mol. Genet.*, 2003, **12**(18), 2301–2309.

12 M. L. Hartsperger, R. Strache and V. Stümpflen, HiNO: an approach for inferring hierarchical organization from regulatory networks, *PLoS One*, 2010, **5**(11), e13698.

13 T. Hinoue, D. J. Weisenberger, C. P. E. Lange, H. Shen, H.-M. Byun, D. Van Den Berg, S. Malik, F. Pan, H. Noushmehr, C. M. van Dijk, R. A. E. M. Tollenaar and P. W. Laird, Genome-scale analysis of aberrant DNA methylation in colorectal cancer, *Genome Res.*, 2011, **22**, 271–282.

14 A. Hodges, A. D. Strand, A. K. Aragaki, A. Kuhn, T. Sengstag, G. Hughes, L. A. Elliston, C. Hartog, D. R. Goldstein, D. Thu, Z. R. Hollingsworth, F. Collin, B. Synek, P. A. Holmans, A. B. Young, N. S. Wexler, M. Delorenzi, C. Kooperberg, S. J. Augood, R. L. Faull, J. M. Olson, L. Jones and R. Luthi-Carter, Regional and cellular gene expression changes in human Huntington's disease brain, *Hum. Mol. Genet.*, 2006, **15**(6), 965–977.

15 T. Ideker, O. Ozier, B. Schwikowski and A. F. Siegel, Discovering regulatory and signalling circuits in molecular interaction networks, *Bioinformatics*, 2002, **18**(Suppl 1), S233–S240.

16 J. Krawczyk, T. A. Kohl, A. Goesmann, J. Kalinowski and J. Baumbach, From *Corynebacterium glutamicum* to *Mycobacterium tuberculosis*—towards transfers of gene regulatory networks and integrated data analyses with MycoRegNet, *Nucleic Acids Res.*, 2009, **37**(14), e97.

17 E. Krol, J. Blom, J. Winnebald, A. Berhörster, M. J. Barnett, A. Goesmann, J. Baumbach and A. Becker, RhizoRegNet—a

database of rhizobial transcription factors and regulatory networks, *J. Biotechnol.*, 2011, **155**(1), 127–134.

18 F. Neumann, D. Sudholt and C. Witt, Analysis of different MMAS ACO algorithms on unimodal functions and plateaus, *Swarm Intell.*, 2009, **3**(1), 35–68.

19 J. Pauling, R. Röttger, A. Tauch, V. Azevedo and J. Baumbach, CoryneRegNet 6.0—updated database content, new analysis methods and novel features focusing on community demands, *Nucleic Acids Res.*, 2012, **40**(D1), D610–D614.

20 T. S. Keshava Prasad, K. Kandasamy and A. Pandey, Human protein reference database and human proteinpedia as discovery tools for systems biology, *Methods Mol. Biol.*, 2009, **577**, 67–79.

21 E. W. Sayers, T. Barrett, D. A. Benson, E. Bolton, S. H. Bryant, K. Canese, V. Chetvernin, D. M. Church, M. DiCuccio, S. Federhen, M. Feolo, I. M. Fingerman, L. Y. Geer, W. Helmberg, Y. Kapustin, D. Landsman, D. J. Lipman, Z. Lu, T. L. Madden, T. Madej, D. R. Maglott, A. Marchler-Bauer, V. Miller, I. Mizrachi, J. Ostell, A. Panchenko, L. Phan, K. D. Pruitt, G. D. Schuler, E. Sequeira, S. T. Sherry, M. Shumway, K. Sirotkin, D. Slotta, A. Souvorov, G. Starchenko, T. A. Tatusova, L. Wagner, Y. Wang, W. John Wilbur, E. Yaschenko and J. Ye, Database resources of the national center for biotechnology information, *Nucleic Acids Res.*, 2011, **39**(Database issue), D38–D51.

22 P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski and T. Ideker, Cytoscape: a software environment for integrated models of biomolecular interaction networks, *Genome Res.*, 2003, **13**(11), 2498–2504.

23 T. Stützle and H. H. Hoos, MAX-MIN ant system, *Future Gener. Comput. Syst.*, 2000, **16**(8), 889–914.

24 I. Ulitsky, R. Karp and R. Shamir, Detecting disease-specific dysregulated pathways *via* analysis of clinical expression profiles, *Proc. RECOMB, Res. Comput. Mol. Biol.*, 2008, **4955**, 347–359.

25 T. Wittkop, D. Emig, S. Lange, S. Rahmann, M. Albrecht, J. H. Morris, S. Böcker, J. Stoye and J. Baumbach, Partitioning biological data with transitivity clustering, *Nat. Methods*, 2010, **7**(6), 419–420.

26 T. Wittkop, S. Rahmann, R. Röttger, S. Böcker and J. Baumbach, Extension and robustness of transitivity clustering for protein–protein interaction network analysis, *Internet Math.*, 2011, **7**, 255–273.

27 Y. Zhou, Runtime analysis of an ant colony optimization algorithm for TSP instances, *IEEE Trans. Evol. Comput.*, 2009, **13**(5), 1083–1092.