Kyoda *et al. Source Code for Biology and Medicine* 2011, **6**:12
http://www.scfbm.org/content/6/1/12

SOURCE CODE FOR
BIOLOGY AND MEDICINE

SOFTWARE REVIEW Open Access

A proof of the DBRF-MEGN method, an algorithm for deducing minimum equivalent gene networks

Koji Kyoda[1,2,3], Kotaro Baba[4,5], Hiroaki Kitano[3,4,6] and Shuichi Onami[1,2,4*]

Abstract

Background: We previously developed the DBRF-MEGN (difference-based regulation finding-minimum equivalent gene network) method, which deduces the most parsimonious signed directed graphs (SDGs) consistent with expression profiles of single-gene deletion mutants. However, until the present study, we have not presented the details of the method's algorithm or a proof of the algorithm.

Results: We describe in detail the algorithm of the DBRF-MEGN method and prove that the algorithm deduces all of the exact solutions of the most parsimonious SDGs consistent with expression profiles of gene deletion mutants.

Conclusions: The DBRF-MEGN method provides all of the exact solutions of the most parsimonious SDGs consistent with expression profiles of gene deletion mutants.

Keywords: DBRF-MEGN method, proof, algorithm, gene network, expression profiles

Background

Identification of gene regulatory networks (hereafter called gene networks) is essential for understanding cellular functions. Large-scale gene deletion projects [1-4] and DNA microarrays [5,6] have enabled the creation of large-scale gene expression profiles of gene deletion mutants [7,8]; these large-scale profiles comprise the expression levels of thousands of genes measured in deletion mutants of those genes. Such profiles are invaluable sources for identifying gene networks. Many procedures have been developed for inferring gene networks from such profiles [9-18].

Kyoda et al. developed the DBRF-MEGN (difference-based regulation finding-minimum equivalent gene network) method, an algorithm for inferring gene networks from large-scale gene expression profiles of gene deletion mutants [14]. In this algorithm, gene networks are modeled as signed directed graphs (SDGs) in which a regulation between two genes is represented as a signed directed edge whose sign - positive or negative - represents whether the effect of the regulation is activation or inhibition and whose direction represents which gene regulates which other gene; the most parsimonious SDGs consistent with the expression profiles are thus deduced. Kyoda et al. showed that the method is applicable to large-scale gene expression profiles of gene deletion mutants and that networks deduced by the method are valid and useful for predicting functions of genes [14]. However, details of the method's algorithm and a proof of the algorithm have not previously been published.

Here we describe in detail the algorithm of the DBRF-MEGN method and prove that the algorithm provides all of the exact solutions of the most parsimonious gene networks consistent with expression profiles of gene deletion mutants.

Implementation

The software of the DBRF-MEGN method was written in C++ under Linux. The complete source code files, a binary Linux executable file, and the software manual are available [see Additional File 1].

Results

Difference-based deduction of initially deduced edges and the minimum equivalent gene networks

The DBRF-MEGN method consists of five processes, namely (1) difference-based deduction of initially deduced edges, (2) removal of non-essential edges from

* Correspondence: sonami@riken.jp
[1]Laboratory for Developmental Dynamics, RIKEN Quantitative Biology Center, and Advanced Computational Sciences Department, RIKEN Advanced Science Institute, 1-7-22 Suehirocho, Tsurumi, Yokohama 230-0045, Japan
Full list of author information is available at the end of the article

the initially deduced edges, (3) selection of the uncovered edges in main components from the non-essential edges, (4) separation of the uncovered edges in main components into independent groups, and (5) restoration of the minimum number of edges from each independent group [14]. First, we define a gene network modeled as an SDG:

Definition 1: A signed directed graph (SDG) is given by a tuple $G = (V, E, f)$ with a set V of nodes (genes), a set $E \subseteq V \times V$ of directed edges, and an edge sign function $f{:}E \rightarrow \{\pm 1\}$, which is an integral part of an SDG.

The first process of the DBRF-MEGN method is "difference-based deduction of initially deduced edges" (Figure 1b), which uses an assumption that is commonly made in genetics and cell biology [14], i.e., there exists a positive (negative) regulation from gene A to gene B when the expression level of gene B in the deletion mutant of gene A is significantly lower (higher) than in the wild-type (Figure 1a). For each possible pair of genes in the profiles, the process determines whether positive (negative) regulations between those genes exist and deduces all edges consistent with both the assumption and the profiles by detecting the difference in expression levels between the wild type and deletion mutants; we call these edges *initially deduced edges*.

Definition 2: Let us assume the intervention experimentshave been performed for the gene set J, $J \subseteq V$. Let $D = (d_{jk}) \in R^{J \times V}$ be a matrix such thatd_{jk} represents the expression of gene $k$after an intervention in gene j (relative to wild-type expression). From this, we deduce the graph initially deduced edges, $G_{ide} = (V, E_{ide}, f)$. We assume a negative regulation of k by j if $d_{jk} > \alpha$ for some suitably chosen constant α. Analogously, a positive regulation of k by j is postulated whenever $d_{jk} < \beta$ for some β (sensibly, we require $\beta < 0 < \alpha$). Formally,

$$E_{ide} = \left\{ (j, k) \in J \times V \mid d_{jk} > \alpha \text{ or } d_{jk} < \beta \right\}$$

and $f{:}E_{ide} \rightarrow \{\pm 1\}$ is given by $f((j,k)) = 1$ if there is a positive regulation of k by j, and otherwise $f((j,k)) = -1$.

The thresholds α and β determine the significance of the difference in expression levels between the wild type and deletion mutants. These thresholds can be specified by various procedures such as by using fold-change or the statistical significance of the expression level [7,8,14,19,20].

The DBRF-MEGN method deduces the most parsimonious SDGs consistent with the SDG that consists of the initially deduced edges. Before defining the most parsimonious SDGs, we need to introduce the function *exp* and the concept *cover* (Figure 2).

Definition 3: If, and only if, $\exists\ (i, j), (j, k), (i, k) \mid f(i, j) \times f(j, k) = f(i, k)$, then $exp(i, j, k) = 1$; otherwise, $exp(i, j, k) = 0$.

Definition 4: Let $E_p \subseteq E_{ide}$ be a set of edges. Define $E_p^{(0)} = E_p$ and by induction $E_p^{(r+1)} = E_p^{(r)} \cup \{ (j, k) \in E_{ide} \mid \exists\, (j, i), (i, k) \in E_p^{(r)}$ such that $exp(j,i,k)=1\}$. Moreover, let $E_p^{\text{cov}} = E_p^{(\infty)}$.

Remark: The family of edge sets on V is partially ordered by set inclusion. If $E_1 \subseteq E_2$, note that by a trivial induction on r, $E_1^{(r)} \subseteq E_2^{(r)}$, and hence $E_1^{\text{cov}} = \cup_{c=0}^{\infty} E_1^{(c)} \subseteq \cup_{c=0}^{\infty} E_2^{(c)} = E_2^{\text{cov}}$. This means that the mapping $.^{\text{cov}} : E \mapsto E^{\text{cov}}$ is monotonic. Let $E \subseteq E_{ide}$. By construction, an edge (j, k) from $\left(E^{\text{cov}}\right)^{\text{cov}}$ is an element of $\left(E^{(r)}\right)^{(s)} = E^{(r+s)}$ for suitable $r,s \in N$. This implies $E^{\text{cov}} \subseteq (E^{\text{cov}})^{\text{cov}} \subseteq \cup_c^{\infty} E^{(c)} = E^{\text{cov}}$. Thus $\left(E^{\text{cov}}\right)^{\text{cov}} = E^{\text{cov}}$, and the mapping $.^{\text{cov}} \mapsto E^{\text{cov}}$ is a so-called closure operation.

Lemma 1: If $E_1 \subseteq E_2$, $E_1^{\text{cov}} \subseteq E_2^{\text{cov}}$.

Proof: The remark proves lemma 1.

Lemma 2: If $E_1 \subseteq E_2^{\text{cov}}$, then $E_1^{\text{cov}} \subseteq E_2^{\text{cov}}$.

Proof: $E_1^{\text{cov}} \subseteq \left(E_2^{\text{cov}}\right)^{\text{cov}} = E_2^{\text{cov}}$ by monotonicity and closure of the mapping $.^{\text{cov}}$.

Lemma 3: If $E_1 \subseteq E_2^{\text{cov}}$ and $E_3 \subseteq E_1^{\text{cov}}$, then $E_3 \subseteq E_2^{\text{cov}}$.

Proof: By $E_3 \subseteq E_1^{\text{cov}} \subseteq \left(E_2^{\text{cov}}\right)^{\text{cov}} = E_2^{\text{cov}}$ by monotonicity and closure of the mapping$.^{\text{cov}}$.

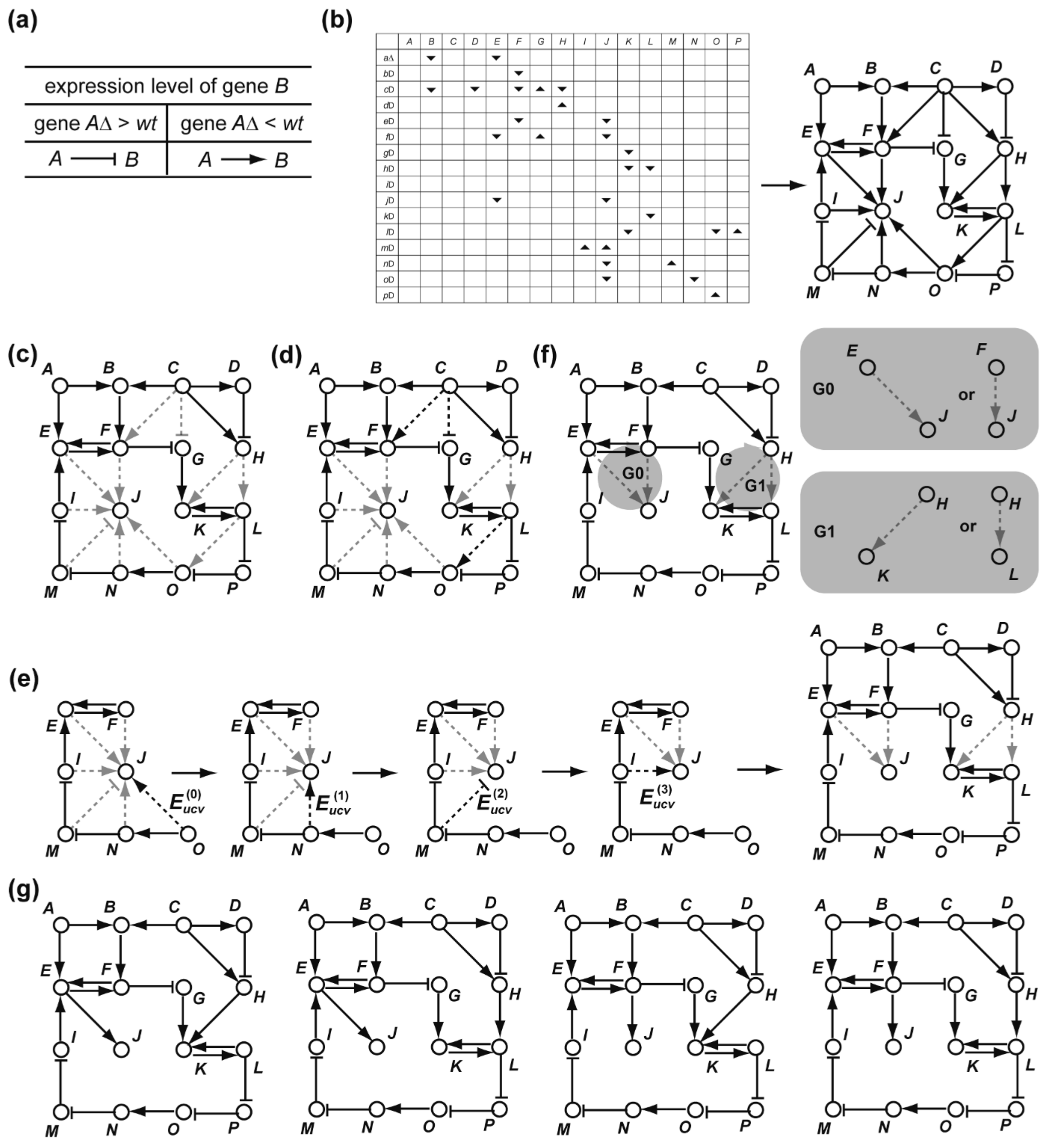
Now, we define the most parsimonious SDGs consistent with the expression profiles of gene deletion mutants. A most parsimonious SDG consists of the minimum number of edges that "cover" all initially deduced edges. By this definition, an edge can be redundant only when it is "explained" by two other initially deduced edges. Importantly, an edge is not redundant when it is "explained" by only three or more initially deduced edges (Figure 3a). We call the most parsimonious SDGs *minimum equivalent gene networks* (*MEGNs*).

Definition 5: $G_0 = \left(V, E_0, f_{E_0}\right)$ (where f_{E_0} is the restriction of f to E_0) is a most parsimonious SDG, named a MEGN, of $G = (V, E_{ide}, f)$ if and only if it satisfies the following conditions: (1) $E_0 \subseteq E_{ide}$, (2) $E_0^{\text{cov}} = E_{ide}$, (3) $\forall\ E_p \subseteq E_{ide}$ such that $E_p^{\text{cov}} = E_{ide}$, $|E_0| \leqslant |E_p|$. Since we keep $G = (V, E_{ide}, f)$ fixed for the rest of the paper, we often call G_0 simply a MEGN, without explicit reference to G.

Removal of non-essential edges from the initially deduced edges

The second process of the DBRF-MEGN method removes all non-essential edges from the initially deduced edges. The process removes all edges that are explained by two other initially deduced edges (Figure 1c). The resulting edges are called *essential edges* and the removed edges are called *non-essential edges*.

Definition 6: If there exist (i, j), (j, k), $(i, k) \in E_{ide}$ such that $exp(i, j, k) = 1$, then (i, k) is called a *non-essential edge*. Let E_{nes} be the set of non-essential edges. The set E_{es} of essential edges is the complement of E_{nes} in E_{ide}, $E_{es} = E_{ide} \backslash E_{nes}$.



expression level of gene *B*	
gene *A*Δ > *wt*	gene *A*Δ < *wt*
$A \dashv B$	$A \rightarrow B$

	A	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*	*K*	*L*	*M*	*N*	*O*	*P*
*a*Δ		▼			▼											
*b*D						▼										
*c*D		▼		▼		▼	▲	▼								
*d*D								▲								
*e*D						▼				▼						
*f*D					▼		▲			▼						
*g*D											▼					
*h*D											▼	▼				
*i*D																
*j*D					▼					▼						
*k*D												▼				
*l*D											▼				▼	▲
*m*D									▲	▲						
*n*D										▼			▲			
*o*D										▼				▼		
*p*D															▲	

Figure 1 An example of the deduction of MEGNs from the expression profiles of gene deletion mutants. **(a)** An assumption used in the DBRF-MEGN method. **(b)** Deduction of the initially deduced edges. The matrix represents a set of expression profiles and the schematic represents a set of initially deduced edges. In the matrix, *A, B*, ... represent expression levels of gene *A*, gene *B*, ..., and *a*Δ, *b*Δ, ... represent deletion mutants of gene *A, B*, ... The up (down) arrows indicate that the gene expression levels are higher (lower) in the deletion mutant than in the wild type. **(c)** Essential edges. Non-essential edges are gray-dotted. **(d)** Uncovered edges. Uncovered edges are gray-dotted and covered edges are black-dotted. **(e)** Exclusion of uncovered edges in peripheral components. $(O, J) \in E_{ucv}^{(0)}$, $(N, J) \in E_{ucv}^{(1)}$, $(M, J) \in E_{ucv}^{(2)}$, and $(I, J) \in E_{ucv}^{(3)}$ are uncovered edges in peripheral components. The resulting four gray-dotted edges are uncovered edges in main components. **(f)** Independent groups of uncovered edges in main components. For each group, the minimum number of edges with which essential edges can explain all edges in the group are shown: (*E, J*) or (*F, J*) for G0, and (*H, K*) or (*H, L*) for G1. **(g)** Four MEGNs of the profiles. Combinations of the minimum numbers of edges of two independent groups (G0 and G1) produce all four MEGNs.

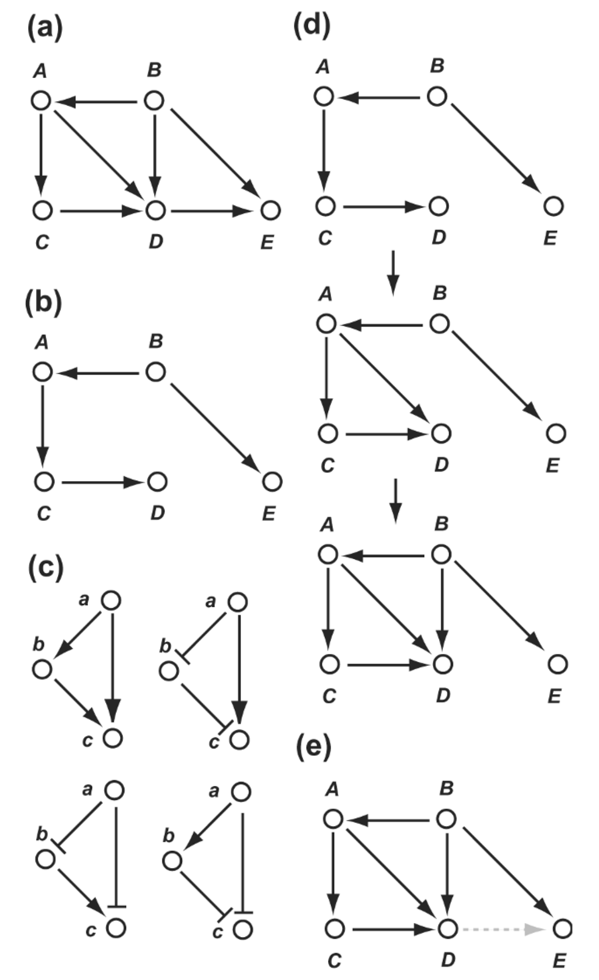


Figure 2 Introduction to the function "exp" and the concept "cover". **(a)** Initially deduced edges (E_{ide}). **(b)** A set of edges (E_p). **(c)** Four cases of *exp*(*a, b, c*) = 1. In each case, (*a, c*) is *explained* by (*a, b*) and (*b, c*). **(d)** The set of edges that are covered by edges in E_p (E_p^{cov}). (*A, C*) and (*C, D*) explain (*A, D*); the (*A, D*) and (*B, A*) explain (*B, D*). Thus, (*A, D*) and (*B, D*) are *covered* by edges in E_p. (*D, E*) is not *covered* by edges in E_p because (*D, E*) cannot be explained by edges in E_p. **(e)** Edges that are covered by edges in E_p (black) and those that are not covered by edges in E_p (gray-dotted).

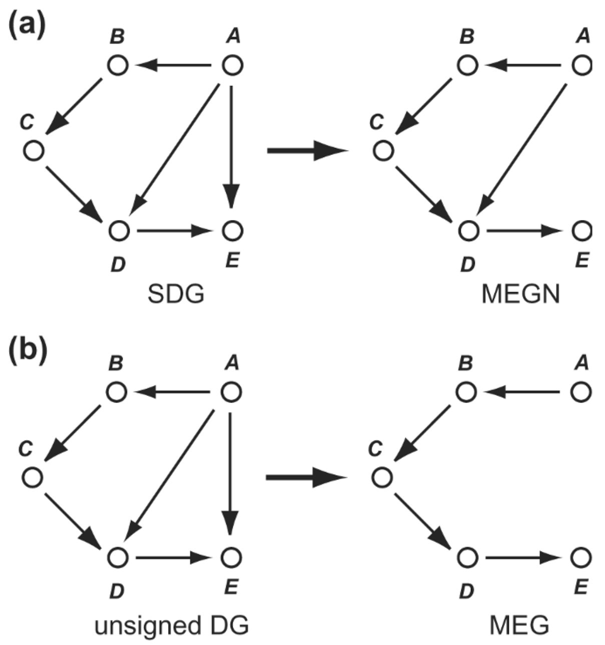


Figure 3 Difference between MEGN and MEG. Deduction of the MEGN **(a)** and the MEG **(b)** from the same graph is shown. The MEGN includes the edge from *A* to *D* because no two edges explain the edge. In contrast, the MEG does not include the edge from *A* to *D* because *A* can reach *D* without using the edge from *A* to *D* (*A*→*B*→*C*→*D*). The MEGN consists only of the essential edges.

Essential edges and non-essential edges have the following properties.

Lemma 4: If $E_p \subseteq E_{ide}$ and $E_p^{\text{cov}} \supseteq E_{es}$, then $E_p \supseteq E_{es}$.

Proof: Assume that there exists $(i, j) \in E_{es}$ such that $(i, j) \in E_p^{\text{cov}}$ and $(i, j) \notin E_p$. Because $(i, j) \in E_p^{\text{cov}}$ and $(i, j) \notin E_p$, there exist $(i, k), (k, j) \in E_p^{\text{cov}}$ such that $exp(i, k, j) = 1$. This contradicts our assumption $(i, j) \notin E_{es}$.

Lemma 5: If $G_0 = \left(V, E_0, f_{E_0}\right)$ is a MEGN, $E_{es} \subseteq E_0$.

Proof: $E_0^{\text{cov}} = E_{ide} \supseteq E_{es}$, hence $E_0 \supseteq E_{es}$ by lemma 4.

When the essential edges cover all initially deduced edges, the SDG consisting of the essential edges is the only MEGN consistent with the profiles.

Theorem 1: If $E_{es}^{\text{cov}} = E_{ide}$, then $G_{es} = \left(V, E_{es}, f_{E_{es}}\right)$ is the unique MEGN of $G = (V, E_{ide}, f)$.

Proof: By hypothesis, conditions (1) $E_{es} \subseteq E_{ide}$, and (2) $E_{es}^{\text{cov}} = E_{ide}$, of a MEGN are met. It remains to show the uniqueness and minimality of E_{es}. (3) Let $G_0 = \left(V, E_0, f_{E_0}\right)$ be an arbitrary MEGN. Then by lemma 5, $E_{es} \subseteq E_0$, and by minimality of E_0, it follows that $E_{es} = E_0$. The theorem is proved.

Selection of the uncovered edges in main components from the non-essential edges

The essential edges sometime fail to cover all initially deduced edges because some edges in the initially deduced edges represent direct gene regulations even when they are explained by two other edges (Figure 1d). In this case, the method restores the minimum number of non-essential edges so that the resulting edges (essential edges and the restored non-essential edges) cover all initially deduced edges. The SDG, consisting of essential edges and of the restored non-essential edges, is a MEGN. Before selecting the sets of non-essential edges to be restored, the method distinguishes non-essential edges that have a chance to be included in the MEGNs from those that do not in order to reduce the number of non-essential edges to be considered for the restoration and thus to reduce the computational cost to find non-essential edges to be restored. This third process of the DBRF-MEGN method consists of two sub-processes, namely (a) selection of uncovered edges and (b) selection of uncovered edges in main components. The resulting non-essential edges are called *uncovered edges in main components*, and from these edges the later processes of the DBRF-MEGN method select edges that are included in the MEGNs.

a) Selection of uncovered edges

The first sub-process distinguishes the non-essential edges that are covered by the essential edges from those that are not (Figure 1d). Those edges are called *covered edges* and *uncovered edges*, respectively.

Definition 7: Let $E_{cv} = (E_{es})^{\mathrm{cov}} \setminus E_{es}$ be the set of *covered* edges. Let $E_{ucv} = E_{ide} \setminus (E_{es} \cup E_{cv})$ be the set of *uncovered* edges. The set of initially deduced edges is thereby partitioned into three disjoint edge sets: $E_{ide} = E_{es} \cup E_{cv} \cup E_{ucv}$.

Here, we prove that the MEGNs do not include covered edges.

Lemma 6: If $G_0 = (V, E_0, f_{E_0})$ is a MEGN, then $E_{es} \subseteq E_0 \subseteq E_{es} \cup E_{ucv}$.

Proof: First, $E_{es} \subseteq E_0$ by lemma 5. By definition 7, $E_{es} \subseteq E_0 \setminus E_{cv}$, hence $(E_0\setminus E_{cv})^{\mathrm{cov}} \supseteq (E_0\setminus E_{cv}) \cup E_{es}^{\mathrm{cov}} \supseteq E_0$ by monotonicity of.$^{\mathrm{cov}}$. It follows that $(E_0\setminus E_{cv})^{\mathrm{cov}} \supseteq E_0^{\mathrm{cov}} = E_{ide}$ by lemma 2. By minimality of E_0, $E_0 = E_0 \setminus E_{cv}$, which is equivalent to $E_0 \cap E_{cv} = \Phi$. By definition 7, this implies $E_0 \subseteq E_{es} \cup E_{ucv}$, completing the proof.

b) Selection of uncovered edges in main components

The second sub-process distinguishes uncovered edges that have a chance to be included in the MEGNs from those that do not (Figure 1e; Figure 4). Those edges are called uncovered edges in main components and *uncovered edges in peripheral components*. The uncovered edges in peripheral components are defined as follows:

Definition 8: Define $E_{ucv}^{(0)}$ be the set of uncovered edges $(i,j) \in E_{ucv}$ which cannot be used to directly explain another uncovered edge in E_{ucv} with the other edges $(k,i) \in E_{ide}$ or $(j,k) \in E_{ide}$.

Lemma 7: $(E_{ide}\setminus E_{ucv}^{(0)})^{\mathrm{cov}} \supseteq E_{ucv}^{(0)}$.

Proof: By definition 8, the edges in $E_{ucv}^{(0)}$ cannot explain another uncovered edges in E_{ucv}. Therefore, the edges in $E_{ucv}^{(0)}$ can be explained by the edges in $E_{ide}\setminus E_{ucv}^{(0)}$. The lemma is proved.

Definition 9: Following the definition 8, define $E_{ucv}^{(r+1)} = E_{ucv}^{(r)} \cup \{(i,j) \in E_{ucv} \setminus E_{ucv}^{(r)}$ which cannot be used to directly explain another uncovered edge in $E_{ucv}\setminus E_{ucv}^{(r)}$ with the other edges $(k,i) \in E_{ide}$ or $(j,k) \in E_{ide}\}$. Let $E_{ucv}^{pc} = \cup_{c=0}^{\infty} E_{ucv}^{(c)}$ be the set of *uncovered edges in peripheral components*. Let $E_{ucv}^{mc} = E_{ucv} \setminus E_{ucv}^{pc}$ be the set of *uncovered edges in main components*. The set of initially deduced edges is thereby partitioned into four disjoint edge sets: $E_{ide} = E_{es} \cup E_{cv} \cup E_{ucv}^{mc} \cup E_{ucv}^{pc}$.

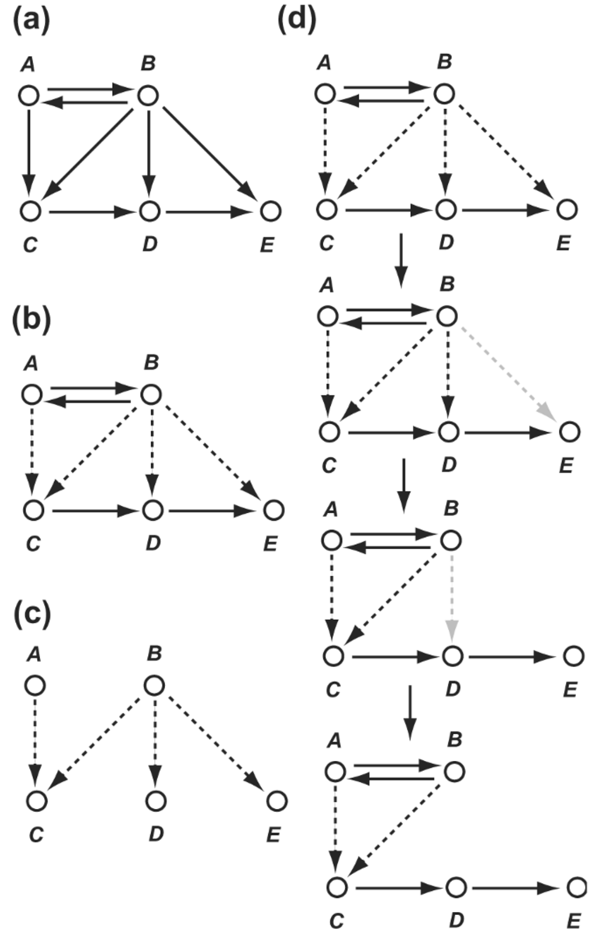


Figure 4 Example of uncovered edges in peripheral and main components. **(a)** Initially deduced edges (E_{ide}). **(b)** Essential edges (E_{es}). Non-essential edges are dotted. **(c)** Uncovered edges (E_{ucv}). Because all non-essential edges cannot be covered by essential edges, the non-essential edges are called *uncovered edges* (E_{ucv}). **(d)** Uncovered edges in peripheral components (gray-dotted). (*B, D*) and (*B, E*) are uncovered edges in peripheral components because (*B, E*) does not explain any other edges in E_{ucv} with an edge in E_{ide}, and (*B, D*) does not explain any other edges in E_{ucv} except (*B, E*) with an edge in E_{ide}. (*A, C*) and (*B, C*) are uncovered edges in main components. If edges in a MEGN cover (*A, C*) and (*B, C*), the edges also cover (*B, D*) and (*B, E*). (*B, D*) and (*B, E*) cover no edges except themselves. Thus, (*B, D*) and (*B, E*) are not included in the MEGNs.

In the following, we prove that the MEGNs do not include uncovered edges in peripheral components. First, we prove that uncovered edges in peripheral components have the following properties.

Lemma 8: $(E_{ide}\setminus \cup_{c=0}^{r} E_{ucv}^{(c)})^{\mathrm{cov}} \supseteq \cup_{c=0}^{r} E_{ucv}^{(c)}$.

Proof: We prove lemma 8 by mathematical induction. (1) By lemma 7, lemma 8 is true when $r = 0$. By definitions 8 and 9, $\left(E_{ide}\backslash\left(E_{ucv}^{(0)}\cup E_{ucv}^{(1)}\right)\right)^{\text{cov}}\supseteq E_{ucv}^{(1)}$, hence $\left(E_{ide}\backslash\left(E_{ucv}^{(0)}\cup E_{ucv}^{(1)}\right)\right)^{\text{cov}}\supseteq\left(E_{ide}\backslash\left(E_{ucv}^{(0)}\cup E_{ucv}^{(1)}\right)\right)\cup E_{ucv}^{(1)}=\left(E_{ide}\backslash E_{ucv}^{(0)}\right)$. By lemmas 2 and 7, $\left(E_{ide}\backslash\left(E_{ucv}^{(0)}\cup E_{ucv}^{(1)}\right)\right)^{\text{cov}}\supseteq E_{ucv}^{(0)}$. Thus, lemma 8 is true when $r = 1$. (2) Assume that lemma 8 is true when $r = m$. This means that we assume that $\left(E_{ide}\backslash\cup_{c=0}^{m}E_{ucv}^{(c)}\right)^{\text{cov}}\supseteq\cup_{c=0}^{m}E_{ucv}^{(c)}$ (2a). By definition 9, $\left(E_{ide}\backslash\cup_{c=0}^{m+1}E_{ucv}^{(c)}\right)^{\text{cov}}\supseteq E_{ucv}^{(m+1)}$ (2b). Because $\left(E_{ide}\backslash\cup_{c=0}^{m+1}E_{ucv}^{(c)}\right)^{\text{cov}}\supseteq E_{ide}\backslash\cup_{c=0}^{m+1}E_{ucv}^{(c)}$ and (2b), $\left(E_{ide}\backslash\cup_{c=0}^{m+1}E_{ucv}^{(c)}\right)^{\text{cov}}\supseteq E_{ide}\backslash\cup_{c=0}^{m}E_{ucv}^{(c)}$ (2c). Because (2a), (2c) and lemma 3, $\left(E_{ide}\backslash\cup_{c=0}^{m+1}E_{ucv}^{(c)}\right)^{\text{cov}}\supseteq\cup_{c=0}^{m}E_{ucv}^{(c)}$ (2d). Because (2b) and (2d), $\left(E_{ide}\backslash\cup_{c=0}^{m+1}E_{ucv}^{(c)}\right)^{\text{cov}}\supseteq\cup_{c=0}^{m+1}E_{ucv}^{(c)}$. Thus, lemma 8 is true when $r = m + 1$, if it is true when $r = m$. By (1) and (2), lemma 8 is true.

Lemma 9: $\left(E_{ide}\backslash E_{ucv}^{pc}\right)^{\text{cov}}\supseteq E_{ucv}^{pc}$.

Proof: By lemma 8, $\left(E_{ide}\backslash\cup_{c=0}^{\infty}E_{ucv}^{(c)}\right)^{\text{cov}}\supseteq E_{ide}\backslash\cup_{c=0}^{\infty}E_{ucv}^{(c)}$. Because $\cup_{c=0}^{\infty}E_{ucv_u}^{(c)}=E_{ucv}^{pc}$, lemma 9 is true.

Now we prove that the MEGNs do not include uncovered edges in peripheral components.

Lemma 10: If $G_0=\left(V,E_0,f_{E_0}\right)$ is a MEGN, $E_0\subseteq E_{es}\cup E_{ucv}^{mc}$.

Proof: Assume that there exists $(i,j)\in E_{ucv}^{pc}\cap E_0$. Because of lemma 5 and definition 7, $E_0^{\text{cov}}\supset\left(E_0\backslash\{(i,j)\}\right)^{\text{cov}}\supseteq E_{es}\cup E_{cv}$, hence $E_0^{\text{cov}}\backslash\left(E_0\backslash\{(i,j)\}\right)^{\text{cov}}\subseteq E_{ucv}$ by lemma 6. By the assumption $(i,j)\in E_{ucv}^{pc}\cap E_0$ and definition 8, $E_0^{\text{cov}}\backslash\left(E_0\backslash\{(i,j)\}\right)^{\text{cov}}\subseteq E_{ucv}^{pc}$, hence $\left(E_0\backslash\{(i,j)\}\right)^{\text{cov}}\supseteq E_{ide}\backslash E_{ucv}^{pc}$. By lemmas 2 and 9, $\left(E_0\backslash\{(i,j)\}\right)^{\text{cov}}\supseteq\left(E_{ide}\backslash E_{ucv}^{pc}\right)^{\text{cov}}\supseteq\left(E_{ide}\backslash E_{ucv}^{pc}\right)\cup E_{ucv}^{pc}=E_{ide}$. This contradicts our assumption that $G_0=\left(V,E_0,f_{E_0}\right)$ is a MEGN. Therefore, $E_0\cap E_{ucv}^{pc}=\phi$. By definition 9 and lemma 6, this implies $E_0\subseteq E_{es}\cup E_{ucv}^{mc}$, completing the proof.

Separation of the uncovered edges in main components into independent groups and restoration of the minimum number of edges from each independent group

The fourth process of the DBRF-MEGN method separates uncovered edges in main components into "independent groups" so that edges to be restored can be deduced independently for each group (Figure 1f; Figure 5). For each group, the fifth process of the DBRF-MEGN method deduces the minimum number of edges with which essential edges can cover all edges in the group. All sets of such edges are deduced for each group. The essential edges and any possible combination of these sets from each group generate a MEGN of the profiles (Figure 1g).

The independent groups are generated so that the edges in one group do not cover those in other groups.

Definition 10: Define $E_{ucv}^{mc(0)}$ be a set of an edge $(i,j)\in E_{ucv}^{mc}$, and by induction

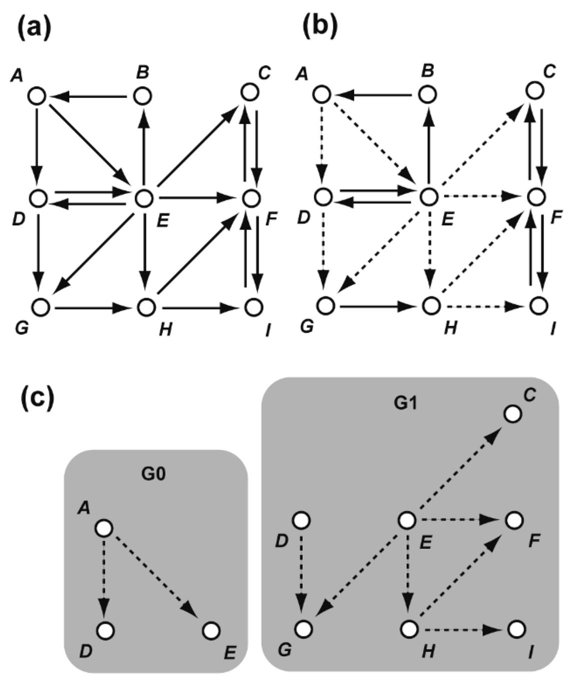


Figure 5 An example of independent groups. **(a)** Initially deduced edges (E_{ide}). **(b)** Essential edges (E_{es}). Uncovered edges in main components (E_{ucv}^{mc}) are dotted. **(c)** Independent groups of uncovered edges in main components. Uncovered edges in main components are separated into two independent groups G0 ($E_{ucv}^{ig_0}$) and G1 ($E_{ucv}^{ig_i}$). Edges in one group do not explain those in other group. $E_{ucv}^{ig_0}$ consists of $E_{ucv}^{mc(0)}=\{(A,\ D)\}$ and $E_{ucv}^{ig_i}$. $E_{ucv}^{ig_i}$ consists of $E_{ucv}^{mc(0)}=\{(D,\ G)\}$, $E_{ucv}^{mc(1)}=\{(E,\ G)\}$, $E_{ucv}^{mc(3)}=\{(E,\ F),\ (H,\ F)\}$, $E_{ucv}^{mc(3)}=\{(E,\ F),\ (H,\ F)\}$, and $E_{ucv}^{mc(4)}=\{(E,\ C),(H,\ I)\}$.

$E_{ucv}^{mc(r+1)}=E_{ucv}^{mc(r)}\cup\{(i,j)\in E_{ucv}^{mc}\backslash E_{ucv}^{mc(r)}|\exists(j,k)\in E_{ide},(i,k)\in E_{ucv}^{mc(r)}$ such that $exp(i,\ j,\ k)=1$ or $\exists(k,i)\in E_{ide},(k,j)\in E_{ucv}^{mc(r)}$ such that $exp(k,\ i,\ j)=1$ or $\exists(i,k)\in E_{ucv}^{mc(r)},(k,j)\in E_{ide}$ such that $exp(i,\ k,\ j)=1$ or $\exists(i,k)\in E_{ide},(k,j)\in E_{ucv}^{mc(r)}$ such that $exp(i,\ k,\ j)=1\}$. Let $E_{ucv}^{ig_0}=E_{ucv}^{mc(\infty)}$ be the set of edges in an *independent group*. Let $E_{ucv}^{ig_{s+1}}=E_{ucv}^{mc(\infty)}$, where $E_{ucv}^{mc(0)}$ is a set of an edge $(i,j)\in E_{ucv}^{mc}\backslash\cup_{c=0}^{s}E_{ucv}^{ig_c}$ and by induction $E_{ucv}^{mc(r+1)}=E_{ucv}^{mc(r)}\cup\left\{(i,j)\in\left(E_{ucv}^{mc}\backslash\cup_{c=0}^{s}E_{ucv}^{ig_c}\right)\backslash E_{ucv}^{mc(r)}|\exists(j,k)\in E_{ide},(i,k)\in E_{ucv}^{mc(r)}\right.$ such that $exp(i,\ j,\ k)=1$ or $\exists(k,i)\in E_{ide},(k,j)\in E_{ucv}^{mc(r)}$ such that $exp(k,\ i,\ j)=1$ or $\exists(i,k)\in E_{ucv}^{mc(r)},(k,j)\in E_{ide}$ such that $exp(i,\ k,\ j)=1$ or $\exists(i,k)\in E_{ide},(k,j)\in E_{ucv}^{mc(r)}$ such that $exp(i,\ k,\ j)=1\}$. Then, $\cup_{c=0}^{\infty}E_{ucv}^{ig_c}=E_{ucv}^{mc}$.

The essential edges and a combination of sets of the minimum number of edges for each independent group generate a MEGN of the profiles.

Definition 11: Let $E_{ucv\min}^{ig_i}$ be the set of edges in ith independent group that satisfies (1) $E_{ucv\min}^{ig_i}\subseteq E_{ucv}^{ig_i}$, (2)

$E_{ucv}^{mc(4)} = \{(E,\ C), (H,\ I)\}$, and (3) $\forall E_p \subseteq E_{ucv}^{ig_i}$ such that $\left|E_{ucv\min}^{ig_i}\right| \leqslant |E_p|$, $\left|E_{ucv\min}^{ig_i}\right| \leqslant |E_p|$.

We prove that the essential edges and a combination of sets of the minimum number of edges for each independent group generate a MEGN of the profiles as follows:

Lemma 11: If there exist $(i,\ j) \in E_{ucv}^{ig_i}$, $(i,\ k)$, $(k,\ j) \in E_{ide}$ such that $exp(i,\ k,\ j) = 1$, then $\{(i,\ k), (k,\ j)\} \cap E_{ucv} \subseteq E_{ucv}^{ig_i}$.

Proof: By definition 10, lemma 11 is true.

Lemma 12: $(\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es})^{\text{cov}} = E_{ide}$.

Proof: By definitions 7 and 11, $(\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es})^{\text{cov}} \supseteq \cup_{c=0}^{\infty} E_{ucv}^{ig_c} \cup E_{es} \cup E_{cv}$. Because $(\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es})^{\text{cov}} \supseteq E_{ucv}^{mc} \cup E_{es} \cup E_{cv} = E_{ide} \backslash E_{ucv}^{pc}$, $(\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es})^{\text{cov}} \supseteq E_{ucv}^{mc} \cup E_{es} \cup E_{cv} = E_{ide} \backslash E_{ucv}^{pc}$. By lemmas 2 and 8, $(\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es})^{\text{cov}} \supseteq E_{ucv}^{pc}$. Therefore, $(\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es})^{\text{cov}} = E_{ide}$.

Theorem 2: $G_{\min} = (V, E_{\min} = \cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es}, f_{E_{\min}})$ is a MEGN.

Proof: (1) $(\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es}) \subseteq E_{ide}$ by the condition of theorem 2.(2) By lemma 12, $(\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es})^{\text{cov}} = E_{ide}$. (3) By lemmas 4, 11 and definition 11, $\forall\ E_p \subseteq E_{ide}$ such that $E_p^{\text{cov}} \supseteq \cup_{c=0}^{\infty} E_{ucv}^{ig_c} \cup E_{es}$, $\left|\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es}\right| \leqslant |E_p|$. Because $(\cup_{c=0}^{\infty} E_{ucv}^{ig_c} \cup E_{es})^{\text{cov}} \supseteq (\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es})^{\text{cov}} = E_{ide}$ and lemma 2, $\forall\ E_p \subseteq E_{ide}$ such that $E_p^{\text{cov}} = E_{ide}$, $\left|\cup_{c=0}^{\infty} E_{ucv\min}^{ig_c} \cup E_{es}\right| \leqslant |E_p|$. The theorem is proved.

Remark: When there exist more than one solution of the minimum number of edges for independent groups, the SDGs each of which consists of the essential edges and a possible combination of sets of the solutions for each independent group are MEGNs because these SDGs must satisfy the conditions in definition 5.

Algorithms of the DBRF-MEGN method

We are concerned with algorithms that are computationally efficient for deducing MEGNs from expression profiles of single-gene deletion mutants. We list these in a form easily translatable into a computer program.

(A1) Algorithm for deducing initially deduced edges

```
double $d[n][n]$: gene expression profiles
  int $t[n][n]$
  void $dbrf()$
    int $i, j$;
    for $i = 1$ to $n$ do
      for $j = 1$ to $n$ do
        if $d[i][j] < \beta$ &$i \neq j$ then
          $t[i][j]: = +1$;
        else if $d[i][j] > \alpha$ &$i \neq j$ then
          $t[i][j]: = -1$;
        else
          $t[i][j]: = 0$;
```

The matrix $d[n][n]$ represents the gene expression profiles. Each entry $d[i][j]$ represents the log-ratio of the expression of gene j in gene i deletion mutants to that in the wild-type. The non-zero entries of the resulting matrix $t[n][n]$ represent the initially deduced edges. If an entry $t[i][j]$ is +1 or-1, it represents a positive or negative edge from gene i to gene j, respectively. The number of complete iterations is bounded by n^2.

(A2) Algorithm for distinguishing the essential edges from the non-essential edges

```
int $t[n][n]$: initially deduced edges
  void $ess\_noness()$
    int $i, j, k$;
    for $j = 1$ to $n$ do
      for $i = 1$ to $n$ do
      if $t[i][j] \neq 0$ then
        for $k = 1$ to $n$ do
          if $t[j][k] \neq 0$ &$t[i][k] \neq 0$ &$t[i][k] = t[i][j] \times t[j][k]$ then
            $check(t[i][k])$;
```

The checked entries of the matrix $t[n][n]$ represent non-essential edges. The unchecked non-zero entries of the resulted matrix $t[n][n]$ represent essential edges. We created this algorithm by modifying Warshall's algorithm [21]. The number of complete iterations is bounded by n^3.

(A3.1) Algorithm for distinguishing uncovered edges from covered edges

```
int $t[n][n]$: initially deduced edges
  int $e[n][n]$: essential edges
    void $covered\_edge()$
      int $i, j, k$;
      bool $finished$;
      $finished : = false$;
      while $finished = false$ do
        $finished : = true$;
        for $i = 1$ to $n$ do
          for $j = 1$ to $n$ do
            if $e[i][j] \neq 0$ then
              for $k = 1$ to $n$ do
                if $e[j][k] \neq 0$ &$t[i][k] \neq 0$ &$t[i][k] = e[i][j] \times e[j][k]$ then
                  $e[i][k]: = t[i][k]$;
                  $check(e[i][k])$;
                  $finished : = false$;
```

The checked entries of the matrix $e[n][n]$ represent covered edges. The non-zero entries of the matrix $t[n]$

[n] that differ from the non-zero entries of the resulted matrix $e[n][n]$ represent uncovered edges. This algorithm iterates over the while loop to find edges in E_{nes} that can be covered by the essential edges. Thus, the number of iterations is bounded by $|E_{nes}| \cdot n^3$.

(A3.2) Algorithm for finding uncovered edges in peripheral components

```
int $t[n][n]$: initially deduced edges
  int $u[n][n]$: uncovered edges
  void peripheral_uncovered()
    int i, j, k;
    bool flag, sflag;
    flag : = false;
    while flag = false do
      flag : = true;
      for $j = 1$ to $n$ do
      for $i = 1$ to $n$ do
        if $u[i][j] \neq 0$ then
        sflag : = false;
        for $k = 1$ to $n$ do
          if $t[j][k] \neq 0$ &$u[i][k] \neq 0$ &$t[i][k] = t[i][j] \times t[j][k]$ then
          sflag : = true;
          if $t[k][i] \neq 0$ &$u[k][j] \neq 0$ &$t[k][j] = t[k][i] \times t[i][j]$ then
          sflag : = true;
          if sflag = false then
            check(u[i][j]);
            flag : = false;
            rm_checked_edge(); // set all checked entries to 0
```

The entries of the resulted matrix $u[n][n]$ that have been changed from +1 or -1 to 0 represent uncovered edges in peripheral components. The non-zero entries of the resulted matrix $u[n][n]$ represent uncovered edges in main components. This algorithm iterates over the while loop to find edges in E_{ucv}that are to be included in E_{ucv}^{pc}. Thus, the number of complete iterations is bounded by $n^3 \cdot |E_{ucv}|$.

(A4.1) Algorithm for dividing uncovered edges in main components (E_{ucv}^{mc}) into independent groups

```
int $t[n][n]$: initially deduced edges
  int $e[n][n]$: uncovered edges in main components
  ig indgrp : independent group
  list <edge >el : edge list
  list <ig >igl : independent group list
  void independent_group()
    int i, j;
    for $i = 1$ to $n$ do
      for $j = 1$ to $n$ do
        if $e[i][j] \neq 0$ then
          el.clear();
          el.append($e_{ij}$);
          append_group(i, j);
          indgrp.init();
          indgrp.set_el(el);     // store edge list el in indgrp
          igl.append(indgrp);     // indgrp : an independent group
  void append_group(int i, int j)
    int x;
    for $x = 1$ to $n$ do
      if $t[i][x] \neq 0$ &$t[x][j] \neq 0$ then
      if $t[i][j] = t[i][x] \times t[x][j]$ then
        if $e[i][x] \neq 0$ then
        el.append($e_{ix}$);
        $e[i][x]: = 0$;
        append_group(i, x);
      if $e[x][j] \neq 0$ then
        el.append($e_{xj}$);
        $e[x][j]: = 0$;
        append_group(x, j);
      if $t[x][i] \neq 0$ &$t[x][j] \neq 0$ then
      if $t[x][j] = t[x][i] \times t[i][j]$ then
        if $e[x][j] \neq 0$ then
        el.append($e_{xj}$);
        $e[x][j]: = 0$;
        append_group(x, j);
      if $t[j][x] \neq 0$ &$t[i][x] \neq 0$ then
      if $t[i][x] = t[i][j] \times t[j][x]$ then
        if $e[i][x] \neq 0$ then
        el.append($e_{ix}$);
        $e[i][x]: = 0$;
        append_group(i, x);
```

The number of complete iterations of *independent_group*() is bounded by n^2. The number of complete iterations of *append_group*(*int, int*) is bounded $(|E_{ucv}^{mc}| - 1) \cdot n$. Thus, the number of complete iterations is bounded by $(|E_{ucv}^{mc}| - 1) \cdot n^3$.

(A4.2) Algorithm for finding all sets of minimum number of edges to be restored in each independent group

```
int $e[n][n]$: essential edges and uncovered edges in peripheral components
  ig indgrp : independent group
  list <ig >igl : independent group list
  list <edge >el, tmp_el : edge list
  list <edge list >combi_el : combination of edge list
    void find_min_ig()
      int i, num_edge;
      for $i = 1$ to igl.size() do
        combi_el.clear(); el.clear();
        indgrp ← igl.get_ig(i);     // copy the ith independent group from igl
        el ← indgrp.get_el();
        for num_edge = 1 to el.size() do
        add_edge(num_edge, 1);
        if (combi_el.size() > 0) then
          break;
          set_min_combi_el(i, combi_el);    // store combi_el in the ith independent group
    void add_edge(int num_edge, int start)
      int j;
      if start + num_edge - 1 >el.size() then
        return;
      for j = start to el.size() do
        set_edge(j);    // set the entry of e[n][n] corresponding to the jth edge
                        // in el to +1 or -1 according to the sign of the edge
        check(el.get_edge(j));    // check the jth edge in el
        if num_edge > 1 then
        add_edge(num_edge - 1, j + 1);
  else
      if (confirm() = true) then
        tmp_el.clear();
        set_tmp_el();    // append all checked edges to tmp_el
        combi_el.append(tmp_el);    // tmp_el : a set of the minimum number of
                                    // edges to be restored
        reset_edge(j);    // set the entry of e[n][n] corresponding to the jth edge
                          // in el to 0
      uncheck(el.get_edge(j));    // uncheck the jth edge in el
  bool confirm(): when resulting edges e[n][n] can covered all edges in the group, return true.
```

The number of complete iterations is bounded by $\sum_{j=1}^{G}\sum_{i=1}^{m_j} {}_{R_j}C_i \cdot (R_j - i) \cdot n_j^3$, where G is the number of independent groups, R_j is the number of edges in the jth independent group, n_j is the number of genes in the jth independent group, and m_j is the number of edges to be restored in the jth independent group.

(A5) Algorithm for deducing all MEGNs by making all possible combinations of sets of the minimum number of edges for each independent group

```
int e[n][n]: essential edges
  int megn : the number of MEGNs
  list <ig >igl : independent group list
  list <edge list >tmp_combi_el : combination of edge list
  list <edge >tmp_el : edge list
  void megn()
    int i;
    i : = 1; megn : = 0;
    sub_megn(i);
    if megn = 0 then
       e[n][n]: MEGN    // e[n][n] represents the MEGN when $E_{es}^{cov} = E_{ide}$
  void sub_megn(int i)
    int x, y, count;
    if i >igl.size() then
      return;
  tmp_combi_el ← get_min_combi_el(i);    // copy combi_el of the ith independent
                                         // group
  for y = 1 to tmp_combi_el.size() do
      tmp_el ← tmp_combi_el.get_el(y);    // copy the yth edge list of tmp_combi_el
      set_edges(tmp_el);    // set the entries of e[n][n] corresponding to the edges in
                            // tmp_el to +1 or -1 according to the signs of the edges
      if i = igl.size() then
        megn++;
        e[n][n]: MEGN    // e[n][n] represents a MEGN when $E_{es}^{cov} \neq E_{ide}$
      else
        sub_megn(i + 1)
      reset_edges(tmp_el);    // set the entries of e[n][n] corresponding to the edges in
                              // tmp_el to 0
```

The number of complete iterations is bounded by $\prod_{j=1}^{G} S_j$, where S_j is the number of sets of minimum number of edges to be restored for the jth independent group.

Discussion

We have described in detail the algorithm of the DBRF-MEGN method and have proved that the algorithm provides all of the exact solutions of the most parsimonious gene networks consistent with expression profiles of gene deletion mutants. The resulting gene networks, called MEGNs, are the most parsimonious SDGs consistent with an SDG that consists of the initially deduced edges. In graph theory, many algorithms have been developed for deducing the most parsimonious unsigned directed graphs consistent with a given unsigned directed graph; these graphs are called minimum equivalent graphs (MEGs) [22-25]. MEGN is not just an "SDG version" of MEG, as is explained below. Although both MEGN and MEG are the most parsimonious graphs of a given graph, the parsimoniousness of the graph is defined differently between these graphs. MEGN consists of the minimum number of edges that *cover* all edges of a given graph (initially deduced edges), whereas MEG consists of the minimum number of edges that *retain the reachability* of a given graph [22]. MEGNs use the cover instead of the reachability because a MEGN is a prediction of a gene network consisting only of direct gene regulations [14]. When positive regulations from gene *A* to gene *B*, from gene *B* to gene *C*, from gene *C* to gene *D*, and from gene *A* to gene *D* are detected and regulation from gene *A* to gene *C* is not detected, the regulation from gene *A* to gene *D* is likely to be a direct regulation instead of an indirect regulation as a result of the other three regulations (Figure 3a). The use of cover makes MEGNs include edges representing such likely direct regulations (Figure 3a). In contrast, the MEGs, using reachability, do not include those edges (Figure 3b). Therefore, the DBRF-MEGN method, which deduces MEGNs, is fundamentally different from algorithms that deduce MEGs or algorithms for transitive reduction of SDG [16-18].

The selection of uncovered edges in main components (the third process) and the generation of independent groups (the fourth process) make the DBRF-MEGN method applicable to large-scale gene expression profiles. Without these processes, the computational cost for finding all sets of non-essential edges to be included in the MEGNs is $n^3\sum_{i=1}^{m} {}_{|E_{nes}|}C_i \cdot (|E_{nes}| - i)$ where n is the number of genes and m is the number of non-essential edges to be included in a MEGN. This computation is impractical for large-scale gene expression profiles because ${}_{|E_{nes}|}C_m$ increases rapidly as $|E_{nes}|$ or m increase. The selection of uncovered edges in main components reduces the computational cost to $n^3\sum_{i=1}^{m} {}_{|E_{ucv}^{mc}|}C_i \cdot \left(\left|E_{ucv}^{mc}\right| - i\right)$ and the generation of independent groups further reduces it to $\sum_{j=1}^{t}\sum_{i=1}^{m_j} {}_{\left|E_{ucv}^{ig_j}\right|}C_i \cdot \left(\left|E_{ucv}^{ig_j}\right| - i\right) \cdot n_j^3$, where t is the number of independent groups, n_j is the number of genes in the jth independent group, and m_j is the number of edges in the jth independent group to be included in a MEGN. $\left|E_{ucv}^{ig_j}\right|$ and m_j are usually far smaller than $|E_{nes}|$ and m. Because of these reductions of the computational cost, the DBRF-MEGN method successfully deduced MEGNs from sets of large-scale gene expression profiles [14] [see Additional file 2, Table S1; Additional file 3]. Although there is no guarantee that the method will deduce MEGNs from any given expression profiles in an acceptable time, the method would most probably deduce MEGNs from most sets of expression profiles in an acceptable time.

Because MEGNs are deduced from initially deduced edges, the accuracy of MEGNs depends on that of initially deduced edges. The primary source for the inaccuracy in initially deduced edges is the noise of the expression profiles. Importantly, the number of false-positive edges in MEGN depends more on that of falsely-detected edges than that of falsely-missed edges in initially deduced edges; the number of false-negative edges in MEGN depends more on that of falsely-missed edges than that of falsely-detected edges in initially deduced edges [see Additional file 2, Table S2; Additional file 2, Figure S1]. These dependencies suggest the following guideline for the thresholds α and β (Definition 2): when the number of false-positive edges is more important than that of false-negative edges in MEGN, α (β) should be a little higher (lower) than the optimal value; in contrast, when the number of false-negative edges is more important than that of false-positive edges in MEGN, α (β) should be a little lower (higher) than the optimal value.

The DBRF-MEGN method is applicable not only to gene expression profiles of deletion mutants but also to those of gene overexpressions and conditional knock-downs/knock-outs [26-28]. We cannot obtain gene expression profiles of deletion mutants for essential genes. Thus, the method cannot deduce gene networks including essential genes when we use gene expression profiles of deletion mutants. A possible solution for this problem is to use the expression profiles of gene overexpressions or conditional knock-downs/knock-outs. Applications of the DBRF-MEGN method to those profiles will deduce gene regulations that cannot be deduced from gene expression profiles of gene deletion mutants.

A limitation of the DBRF-MEGN method is its inability to deduce (1) self-regulation of genes, and (2) combinatorial gene regulations such as regulation in which the expression of gene *A* is down-regulated only when both gene *B* and gene *C* are inactive. Self-regulation could be deduced by using chromatin immunoprecipitation [29]. Combinatorial gene regulations could be deduced by using the expression profiles of multiple gene deletion mutants [30]. Synthetic genetic arrays can systematically construct a collection of double-gene deletion mutants [31]. A combination of the DBRF-MEGN method and the above techniques would provide more accurate information about gene networks.

When the DBRF-MEGN method is applied to gene expression profiles measured by using DNA microarray, each of the deduced edges represents regulation of one gene's mRNA level by another gene's activity. Therefore, the deduced MEGNs do not include edges that represent post-transcriptional gene regulations although they play major roles in the cell. However, because the algorithm of the DBRF-MEGN method is based on logic that is most commonly used in genetics and cell biology to infer gene networks from small-scale experiments, we can predict post-transcriptional modulators of transcriptional activity from those MEGNs. We predicted total 72 transcriptional regulators and 232 post-transcriptional modulators of 18 transcriptional regulators from the MEGNs deduced from a set of gene expression profiles for 265 *Saccharomyces cerevisiae* genes [14]. The DBRF-MEGN method is applicable not only to gene expression profiles measured by using DNA microarray but also to those measured by using other technologies such as 2D-PAGE-MS [32] and protein chips [33]. MEGNs deduced from those non-DNA microarray expression profiles will include edges that represent post-transcriptional gene regulations in the cell.

Conclusions

We described in detail the processes of the DBRF-MEGN method and proved that these processes provide all of the exact solutions of the most parsimonious gene networks consistent with the expression profiles of gene deletion mutants, which are called MEGNs. The DBRF-MEGN method provides invaluable information for understanding cellular functions.

Availability and requirements

Project name: DBRF-MEGN
Project home page: http://so.gsc.riken.jp/dbrf-megn
Operating system: Linux
Programming language: C++
Other requirements: None
Licence: GNU LGPL
Any restrictions to use by non-academics: Licence required

Additional material

Additional file 1: The complete source code files, a binary Linux executable file, and the software manual.

Additional file 2: Supporting text for the applicability of the DBRF-MEGN method to the large-scale expression profiles and the sensitivity of the DBRF-MEGN method to the noise of the expression profiles.

Additional file 3: The MEGNs deduced from large-scale gene expression profiles.

List of abbreviations

DBRF: difference-based regulation finding; MEGN: minimum equivalent gene network; SDG: signed directed graph; MEG: minimum equivalent graph.

Acknowledgements

We thank the anonymous reviewer for his/her thorough review and constructive suggestions, which significantly contributed to improving the readability of definitions and lemmas. We thank K. Oka for his support and useful discussions. We also thank S. Hamahashi and A. Kimura for critical comments on this manuscript. This work was supported in part by a grant from the Ministry of Agriculture, Forestry and Fisheries of Japan (Rice Genome Project SY1106) to HK and SO; by Special Coordination Funds for Promoting Science and Technology (to HK and SO), and by a Grant-in-Aid for Scientific Research on Priority Areas "Systems Genomics" (to KK and SO) from the Ministry of Education, Culture, Sports, Science and Technology of Japan; and by a grant from the Institute for Bioinformatics Research and Development (BIRD), Japan Science and Technology Agency to SO.

Author details

[1]Laboratory for Developmental Dynamics, RIKEN Quantitative Biology Center, and Advanced Computational Sciences Department, RIKEN Advanced Science Institute, 1-7-22 Suehirocho, Tsurumi, Yokohama 230-0045, Japan. [2]Graduate School of Science and Technology, Keio University, 3-14-1 Hiyoshi, Kohoku, Yokohama 223-8522, Japan. [3]Kitano Symbiotic Systems Project, ERATO, Japan Science and Technology Corporation, M31 6A, 6-31-15 Jingumae, Shibuya, Tokyo 150-0001, Japan. [4]The Systems Biology Institute, 5-6-9 Shirokanedai, Minato, Tokyo 108-0071, Japan. [5]National Institute of Agrobiological Sciences, 2-1-2 Kannondai, Tsukuba, Ibaraki 305-8602, Japan. [6]Sony Computer Science Laboratories, Inc, 3-14-13 Higashi-gotanda, Shinagawa, Tokyo 141-0022, Japan.

Authors' contributions

KK participated in this study's conception, designed the algorithm, proved the algorithm, and drafted the manuscript. KB implemented the algorithm. HK participated in algorithm design. SO conceived this study, designed the algorithm, proved the algorithm, and drafted the manuscript. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Received: 13 September 2010 Accepted: 24 June 2011
Published: 24 June 2011

References

1. Liu LX, Spoerke JM, Mulligan EL, Chen J, Reardon B, Westlund B, Sun L, Abel K, Armstrong B, Hardiman G, King J, McCague L, Basson M, Clover R, Johnson CD: **High-throughput isolation of *Caenorhabditis elegans* deletion mutants.** *Genome Res* 1999, **9**:859-867.
2. Winzeler EA, Shoemaker DD, Astromoff A, Liang H, Anderson K, Andre B, Bangham R, Benito R, Boeke JD, Bussey H, Chu AM, Connelly C, Davis K, Dietrich F, Dow SW, Bakkoury ME, Foury F, Friend SH, Gentalen E, Giaever G, Hegemann JH, Jones T, Laub M, Liao H, Liebundguth N, Lockhart DJ, Lucau-Danila A, Lussier M, M'Rabet N, Menard P, Mittmann M, Pai C, Rebischung C, Revuelta JL, Riles L, Roberts CJ, Ross-MacDonald P, Scherens B, Snyder M, Sookhai-Mahadeo S, Storms RK, Véronneau S, Voet M, Volckaert G, Ward TR, Wysocki R, Yen GS, Yu K, Zimmermann K, Philippsen P, Johnston M, Davis RW: **Functional characterization of the *S. cerevisiae* genome by gene deletion and parallel analysis.** *Science* 1999, **285**:901-906.
3. Hamer L, Adachi K, Montenegro-Chamorro MV, Tanzer MM, Mahanty SK, Lo C, Tarpey RW, Skalchunes AR, Heiniger RW, Frank SA, Darveaux BA, Lampe DJ, Slater TM, Ramamurthy L, DeZwaan TM, Nelson GH, Shuster JR, Woessner J, Hamer JE: **Gene discovery and gene function assignment in filamentous fungi.** *Proc Natl Acad Sci* 2001, **98**:5110-5115.
4. Giaever G, Chu AM, Connelly C, Riles L, Véronneau S, Dow S, Lucau-Danila A, Anderson K, André B, Arkin AP, Astromoff A, Bakkoury ME, Bangham R, Benito R, Brachat S, Campanaro S, Curtiss M, Davis K, Deutschbauer A, Entian KD, Flaherty P, Foury F, Garfinkel DJ, Gerstein M, Gotte D, Güldener U, Hegemann JH, Hempel S, Herman Z, Jaramillo DF, Kelly DE, Kelly SL, Kötter P, LaBonte D, Lamb DC, Lan N, Liang H, Liao H, Liu L, Luo C, Lussier M, Mao R, Menard P, Ooi SL, Revuelta JL, Roberts CJ, Rose M, Ross-Macdonald P, Scherens B, Schimmack G, Shafer B, Shoemaker DD, Sookhai-Mahadeo S, Storms RK, Strathern JN, Valle G, Voet M, Volckaert G, Wang CY, Ward TR, Wilhelmy J, Winzeler EA, Yang Y, Yen G, Youngman B, Yu K, Bussey H, Boeke JD, Snyder M, Philippsen P, Davis RW, Johnston M: **Functional profiling of the *Saccharomyces cerevisiae* genome.** *Nature* 2002, **418**:387-391.
5. Schena M, Shalon D, Davis RW, Brown PO: **Quantitative monitoring of gene expression patterns with a complementary DNA microarray.** *Science* 1995, **270**:467-470.
6. Lockhart DJ, Dong H, Byrne MC, Follettie MT, Gallo MV, Chee MS, Mittmann M, Wang C, Kobayashi M, Horton H, Brown EL: **Expression monitoring by hybridization to high-density oligonucleotide arrays.** *Nat Biotechnol* 1996, **14**:1675-1680.
7. Hughes TR, Marton MJ, Jones AR, Roberts CJ, Stoughton R, Armour CD, Bennett HA, Coffey E, Dai H, He YD, Kidd MJ, King AM, Meyer MR, Slade D, Lum PY, Stepaniants SB, Shoemaker DD, Gachotte D, Chakraburtty K, Simon J, Bard M, Friend SH: **Functional discovery via a compendium of expression profiles.** *Cell* 2000, **102**:109-126.
8. Hu Z, Killion PJ, Iyer VR: **Genetic reconstruction of a functional transcriptional regulatory network.** *Nat Genet* 2007, **39**:683-687.
9. Ideker TE, Thorsson V, Karp RM: **Discovery of regulatory interactions through perturbation: inference and experimental design.** *Pac Symp Biocomput* 2000, **5**:305-316.
10. Kyoda KM, Morohashi M, Onami S, Kitano H: **A gene network inference method from continuous-value gene expression data of wild-type and mutants.** *Genome Inform Ser Workshop Genome Inform* 2000, **11**:196-204.

11. Pe'er D, Regev A, Elidan G, Friedman N: **Inferring subnetworks from perturbed expression profiles.** *Bioinformatics* 2001, **Suppl 17**:S215-S224.
12. Wagner A: **How to reconstruct a large genetic network from *n* gene perturbations in fewer than n^2 easy steps.** *Bioinformatics* 2001, **17**:1183-1197.
13. Rung J, Schlitt T, Brazma A, Freivalds K, Vilo J: **Building and analysing genome-wide gene disruption networks.** *Bioinformatics* 2002, **Suppl 18**: S202-S210.
14. Kyoda K, Baba K, Onami S, Kitano H: **DBRF-MEGN method: an algorithm for deducing minimum equivalent gene networks from large-scale gene expression profiles of gene deletion mutants.** *Bioinformatics* 2004, **20**:2665-2675.
15. Bonneau R, Reiss D, Shannon P, Facciotti M, Hood L, Baliga NS, Thorsson V: **The Inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets *de novo*.** *Genome Biol* 2006, **7**: R36.
16. Albert R, DasGupta B, Dondi R, Kachalo S, Sontag E, Zelikovsky A, Westbrooks K: **A novel method for signal transduction network inference from indirect experimental evidence.** *J Comput Biol* 2007, **14**:927-949.
17. Tresch A, Beissbarth T, Sultmann H, Kuner R, Poustka A, Buness A: **Discrimination of direct and indirect interactions in a network of regulatory effects.** *J Comput Biol* 2007, **14**:1217-1228.
18. Klamt S, Flassig RJ, Sundmacher K: **TRANSWESD: inferring cellular networks with transitive reduction.** *Bioinformatics* 2010, **17**:2160-2168.
19. Schena M, Shalon D, Heller R, Chai A, Brown PO, Davis RW: **Parallel human genome analysis: microarray-based expression monitoring of 1000 genes.** *Proc Natl Acad Sci* 1996, **93**:10614-10619.
20. DeRisi JL, Iyer VR, Brown PO: **Exploring the metabolic and genetic control of gene expression on a genomic scale.** *Science* 1997, **278**:680-686.
21. Warshall S: **A theorem on Boolean matrices.** *J Assoc Comput Mach* 1962, **9**:11-12.
22. Moyles DM, Thompson GL: **An algorithm for finding a minimum equivalent graph of a digraph.** *J Assoc Comput Mach* 1969, **16**:455-460.
23. Hsu H: **An algorithm for finding a minimal equivalent graph of a digraph.** *J Assoc Comput Mach* 1975, **22**:11-16.
24. Martello S: **An algorithm for finding a minimal equivalent graph of a strongly connected digraph.** *Computing* 1979, **21**:183-194.
25. Martello S, Toth P: **Finding a minimum equivalent graph of a digraph.** *Networks* 1982, **12**:89-100.
26. Mnaimneh S, Davierwala AP, Haynes J, Moffat J, Peng WT, Zhang W, Yang X, Pootoolal J, Chua G, Lopez A, Trochesset M, Morse D, Krogan NJ, Hiley SL, Li Z, Morris Q, Grigull J, Mitsakakis N, Roberts CJ, Greenblatt JF, Boone C, Kaiser CA, Andrews BJ, Hughes TR: **Exploration of essential gene functions via tetratable promoter alleles.** *Cell* 2004, **118**:31-44.
27. Sopko R, Huang D, Preston N, Chua G, Papp B, Kafadar K, Snyder M, Oliver SG, Cyert M, Hughes TR, Boone C, Andrews B: **Mapping pathways and phenotypes by systematic gene overexpression.** *Mol Cell* 2006, **21**:319-330.
28. Chua G, Morris QD, Sopko R, Robinson MD, Ryan O, Chan ET, Frey BJ, Andrews BJ, Boone C, Hughes TR: **Identifying transcription factor functions and targets by phenotypic activation.** *Proc Natl Acad Sci* 2006, **103**:12045-12050.
29. Lee TI, Rinaldi NJ, Robert F, Odom DT, Bar-Joseph Z, Gerber GK, Hannett NM, Harbison CT, Thompson CM, Simon I, Zeitlinger J, Jennings EG, Murray HL, Gordon DB, Ren B, Wyrick JJ, Tagne JB, Volkert TL, Fraenkel E, Gifford DK, Young RA: **Transcriptional regulatory networks in *Saccharomyces cerevisiae*.** *Science* 2002, **298**:799-804.
30. Tringe SG, Wagner A, Ruby SW: **Enriching for direct regulatory targets in perturbed gene-expression profiles.** *Genome Biol* 2004, **5**:R29.
31. Tong AH, Evangelista M, Parsons AB, Xu H, Bader GD, Pagé N, Robinson M, Raghibizadeh S, Hogue CW, Bussey H, Andrews B, Tyers M, Boone C: **Systematic genetic analysis with ordered arrays of yeast deletion mutants.** *Science* 2001, **294**:2364-2368.
32. Gygi SP, Rochon Y, Franza BR, Aebersold R: **Correlation between protein and mRNA abundance in yeast.** *Mol Cell Biol* 1999, **19**:1720-1730.
33. Zhu H, Klemic JF, Chang S, Bertone P, Casamayor A, Klemic KG, Smith D, Gerstein M, Reed MA, Snyder M: **Analysis of yeast protein kinases using protein chips.** *Nat Genet* 2000, **26**:283-289.

doi:10.1186/1751-0473-6-12
Cite this article as: Kyoda *et al.*: A proof of the DBRF-MEGN method, an algorithm for deducing minimum equivalent gene networks. *Source Code for Biology and Medicine* 2011 **6**:12.