BMC Bioinformatics

RESEARCH ARTICLE Open Access

Exact score distribution computation for ontological similarity searches

Marcel H Schulz[1,2*], Sebastian Köhler[3,4], Sebastian Bauer[3] and Peter N Robinson[1,3,4*]

Abstract

Background: Semantic similarity searches in ontologies are an important component of many bioinformatic algorithms, e.g., finding functionally related proteins with the Gene Ontology or phenotypically similar diseases with the Human Phenotype Ontology (HPO). We have recently shown that the performance of semantic similarity searches can be improved by ranking results according to the probability of obtaining a given score at random rather than by the scores themselves. However, to date, there are no algorithms for computing the exact distribution of semantic similarity scores, which is necessary for computing the exact *P*-value of a given score.

Results: In this paper we consider the exact computation of score distributions for similarity searches in ontologies, and introduce a simple null hypothesis which can be used to compute a *P*-value for the statistical significance of similarity scores. We concentrate on measures based on Resnik's definition of ontological similarity. A new algorithm is proposed that collapses subgraphs of the ontology graph and thereby allows fast score distribution computation. The new algorithm is several orders of magnitude faster than the naive approach, as we demonstrate by computing score distributions for similarity searches in the HPO. It is shown that exact *P*-value calculation improves clinical diagnosis using the HPO compared to approaches based on sampling.

Conclusions: The new algorithm enables for the first time exact *P*-value calculation via exact score distribution computation for ontology similarity searches. The approach is applicable to any ontology for which the annotation-propagation rule holds and can improve any bioinformatic method that makes only use of the raw similarity scores. The algorithm was implemented in Java, supports any ontology in OBO format, and is available for non-commercial and academic usage under: https://compbio.charite.de/svn/hpo/trunk/src/tools/significance/

Background

Ontologies are knowledge representations using controlled vocabularies that are designed to help knowledge sharing and computer reasoning [1]. Many ontologies can be represented by directed acyclic graphs (DAGs), whereby the nodes of the DAG, which are also called *terms* of the ontology, are assigned to items in the domain and the edges between the nodes represent semantic *relations*. Ontologies are designed such that terms closer to the root are more general than their descendant terms. For the ontologies we consider in this paper, the *annotation-propagation rule* applies, that is, items are annotated to the most specific term possible but are assumed to be implicitly annotated to all ancestors of that term.

Examples for ontologies are the Foundational Model of Anatomy (FMA) ontology [2], the Sequence Ontology [3], the Cell Ontology [4], and the Chemical Entities of Biological Interest (ChEBI) ontology [5], which describe objects from the domains of anatomy, biological sequences, cells, and biologically relevant chemicals. In contrast, other ontologies are used to describe the attributes of the items of a domain. For instance, GO terms are used to annotate genes or proteins by describing the biological functions or characteristics to the proteins. The Mammalian Phenotype Ontology (MPO) [6] and the Human Phenotype Ontology (HPO) [7] describe the attributes of mammalian and human diseases. In this case, the domain object is a disease such as Marfan syndrome, whose attributes are the clinical features of the disease such as arachnodactyly and aortic dilatation. In other words, terms of phenotype ontologies such as the MPO and HPO can be conceived of as

* Correspondence: maschulz@andrew.cmu.edu; peter.robinson@charite.de
[1]Max Planck Institute for Molecular Genetics, Ihnestr. 73, 14195 Berlin, Germany
Full list of author information is available at the end of the article

describing abnormal qualities (e.g. hypoplastic) of anatomical or biochemical entities [8].

Semantic similarity between any two terms within an ontology is based on the annotations to items in the domain and on the structure of the DAG. Different semantic similarity measures have been proposed [9,10] and the measures have been used in many different applications in computational biology. For example, different studies show that semantic similarity between proteins annotated with GO terms correlate with sequence similarity [11-13]. Other studies investigated the correlation of gene coexpression with semantic similarity using GO terms [14,15]. In addition, semantic similarity measures for GO terms have been used to predict protein subnuclear localization [16].

In another application we have implemented a semantic similarity search algorithm in the setting of medical diagnosis. A user enters HPO terms describing the clinical abnormalities observed in a patient and a ranked list of the best matching differential diagnoses is returned [17]. This kind of search can be performed using raw semantic similarity scores calculated using any of the semantic similarity measures [18-21,12,22]. However, among these different measures the node-based pairwise similarity measure defined by Resnik turned out to have the best performance in our previous study [17] and is therefore considered in this work.

The search is based on q attributes (HPO terms) that describe the phenotypic abnormalities seen in a patient for whom a diagnosis is being sought. For each of the entries of a database containing diseases annotated with HPO terms corresponding to their characteristic signs and symptoms, the best match between each of the q terms of the query with one of the terms annotating the disease is found and the average of the semantic similarity scores is determined. The diseases are then ranked according to these scores and returned to the user as suggestions for the *differential diagnosis*.

The distribution of scores that a domain object can achieve varies according to the number and specificity of the ontology terms used to annotate it. In a recent study by Wang et al. [23], it was discovered that many of the commonly used semantic similarity measures, including the ones used in this work, are biased towards domain objects that have more annotations. The effect was termed annotation bias. Applications that use the scores alone therefore tend to preferentially select items with higher numbers of annotations, which may lead to wrong conclusions [23].

Previously, we developed a statistical model to assign P-values to the resulting similarity scores on the basis of the probability of a random query obtaining at least as high a score in order to compensate for the fact that different domain objects may have a different number of annotations. Using extensive simulations, we showed that this approach outperformed searches based on the semantic similarity scores alone [17]. A disadvantage of that procedure was the fact that extensive simulations using randomized queries were necessary in order to estimate the true distribution of the semantic similarity scores, which is needed in order to calculate a P-value for any given similarity score.

In this paper, we describe an algorithm to collapse a DAG representing an ontology into connected components of nodes corresponding to terms that make identical contributions to the semantic similarity score. The new algorithm reduces the amount of computational time needed to calculate the score distribution (and thereby P-values) by many orders of magnitude compared to a naive calculation. A preliminary description of the algorithm was presented in a conference paper [24]. Here, we validate the algorithm by comparing to sampling based approaches and show using simulations that the application of the exact P-value outperforms sampling based approaches in the context of clinical diagnostics with the HPO.

Methods

Notation

We consider an ontology O composed of a set of *terms* that are linked via an *is-a* or *part-of* relationship. The ontology O can then be represented by a DAG $\mathcal{G} = (V, E)$, where every term is a node in V and every link is a directed edge in E. A directed edge going from node n_1 to n_2 is denoted $e_{1,2}$ and we refer to n_2 as the *parent* of n_1. An *item* i is defined as an abstract entity to which terms of the ontology are annotated. Let $Anc(n)$ be defined as the ancestors of n, i.e., the nodes that are found on all paths from node n to the root of $\mathcal{G}$, including n. We note that the annotation-propagation rule states that if an item is explicitly annotated to a term n, it is implicitly annotated to $Anc(n)$. In order to describe the implicit annotations we define $\mathcal{T}^{IMPL}$. Let $\mathcal{T}$ be the set of terms that has been explicitly annotated to item i, then $\mathcal{T}^{IMPL} = \cup_{n\in\mathcal{T}} Anc(n)$, namely all terms that are annotated to item i and all their ancestors in $\mathcal{G}$. Let the set of common ancestors of two nodes n_1 and n_2 be defined as $ComAnc(n_1, n_2) = Anc(n_1) \cap Anc(n_2)$. Let $Desc(n)$ be the set of descendant nodes of n, again including n. Note that in this notation descendant nodes are considered only once, even if there are multiple paths leading to them.

Multisets

In what follows we need to compute the similarity also between a *multiset* and a set of terms. The concept of multisets [25] is a generalization of the concept of sets.

In contrast to sets, in which elements can only have a single membership, the elements of multisets may appear more than once.

Formally, a multiset M is a set of pairs, $M = \{(s_1, m_1),..., (s_d, m_d)\}$, in which $s_i \in U = \{s_1,..., s_d\}$ are the elements of the *underlying set* U. Furthermore, m_i defines the *multiplicity* of s_i in the multiset. The sum of the multiplicities of M is called the *multiset cardinality* of M, denoted $|M|$. Only multiplicities in the domain of positive integers are considered, i.e., $m_i \in \mathbb{N}^+$. We define a *multi subset* relation between multiset N and multiset M, denoted *as* $N \subseteq M$, as a generalization of the subset relation between two sets:

$$N \subseteq M \Leftrightarrow \forall (s, n) \in N : \exists m \geq n : (s, m) \in M.$$

The *multiset coefficient* $M(n, q) = \binom{n+q-1}{q}$ denotes the number of distinct multisets of cardinality q, with elements taken from a finite set of cardinality n. It describes how many ways there are to choose q elements from a set of n elements if repetitions are allowed.

Similarity measures

We will concentrate in this work on the class of similarity measures that are based on the information content (IC) of a node:

$$IC(n) = -\log\, p(n), \tag{1}$$

where $p(n)$ denotes the frequency among all items in the domain of annotations to n, which implicitly contains all annotations of descendants of n due to the annotation-propagation rule. The information content is a nondecreasing function on the nodes of $\mathcal{G}$ as we descend in the hierarchy and is therefore *monotonic*. The similarity between two nodes was defined by Resnik as the maximum information content among all common ancestors [19]:

$$sim(n_1, n_2) = max\{IC(a) | a \in ComAnc(n_1, n_2)\}. \tag{2}$$

Equation (2) provides a definition for the similarity between two terms. Other popular pairwise measures that additionally incorporate the IC of the query terms, for example [20,21], are not considered here (see Discussion).

One can extend this concept to define a similarity between two domain objects that are each annotated by multiple ontology terms by taking the average of the best pairwise similarities for all terms [11]:

$$sim^{avg}(\mathcal{T}_1, \mathcal{T}_2) = \frac{1}{|\mathcal{T}_1|} \sum_{n_1 \in \mathcal{T}_1} \max_{n_2 \in \mathcal{T}_2} sim(n_1, n_2). \tag{3}$$

Note that Eq. (3) is not symmetric [12], i.e., it is not necessarily true that $sim^{avg}(\mathcal{T}_1, \mathcal{T}_2) = sim^{avg}(\mathcal{T}_2, \mathcal{T}_1)$. We point out that in other works average often refers to a symmetric definition. Using the nomenclature of Pesquita *et al.* [9], Eq. (3) may be referred to as asymmetric best-match average, here average for short.

Instead of taking the average the maximum similarity between a term annotating one of the domain objects and a term annotating the other domain object can be used to define the following symmetric measure:

$$sim^{max}(\mathcal{T}_1, \mathcal{T}_2) = \max_{n_1 \in \mathcal{T}_1, n_2 \in \mathcal{T}_2} sim(n_1, n_2). \tag{4}$$

Equation (4) can be considered a simplified case of Eq. (3) because instead of averaging over all best-pairwise terms for each $n_1 \in \mathcal{T}_1$ compared to $n_2 \in \mathcal{T}_2$ only the highest similarity of all possible pairs is retained. Therefore, we will show the algorithm applied to Eq. (3) and sketch the changes for Eq. (4) later. One can use equation (3) or (4) to define a similarity between a set of query terms $\mathcal{Q}$, i.e., $\mathcal{T}_1 = \mathcal{Q}$ and an object in a database. Then, $\mathcal{Q}$ can represent any set of terms from the ontology O whereas $\mathcal{T}_2$ refers to database objects (such as diseases annotated to HPO terms). As we are using this setup for the similarity queries we will omit the index and refer to $\mathcal{T}_2$ as the target set $\mathcal{T}$. See Figure 1 for an example computation of sim^{avg}.

Because we later make use of scores derived at the maximization step in Eq. (3) we define:

$$sim(n_1, \mathcal{T}) = \max_{n_2 \in \mathcal{T}} sim(n_1, n_2), \tag{5}$$

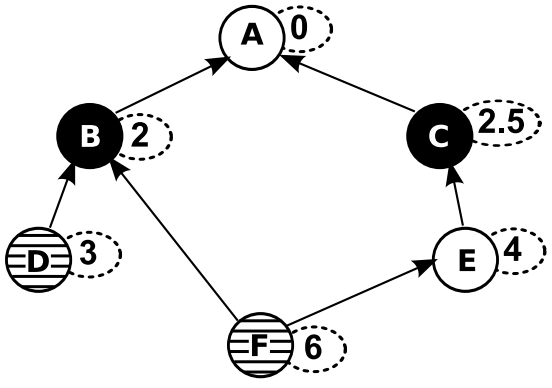

Figure 1 Example Computation of sim^{avg}. Computation of sim^{avg} on a DAG with six nodes. The target set is $\mathcal{T} = \{B, C\}$ (black nodes) and the query set $\mathcal{Q} = \{D, F\}$ (nodes with horizontal lines). The IC value of a node is shown in a small, dashed, attached oval. The most similar terms for D and F are B and C respectively, because $IC(B) > IC(A)$ and $IC(C) > IC(B)$. Therefore, $sim^{avg}(\mathcal{Q}, \mathcal{T}) = \frac{sim(D,B) + sim(F,C)}{2} = \frac{IC(B) + IC(C)}{2} = \frac{2 + 2.5}{2}$. Note that only terms involving nodes in $\mathcal{T}^{IMPL} = \{A, B, C\}$ were considered in the calculation.

to be the *target set similarity score* of n_1 against a target set $\mathcal{T}$. To avoid confusion we will denote scores of the score distribution of sim^{avg} by S and target set similarity scores $sim(n, \mathcal{T})$ by s.

Definition of statistical significance for semantic similarity scores

In this paper we will present methods for analytically calculating the probability distribution of similarity scores for comparisons between a query set $\mathcal{Q}$ with q terms against an item that has been annotated with a *target set* $\mathcal{T}$ of nodes. For example, if a clinician chooses a set $\mathcal{Q}$ of HPO terms describing abnormalities seen in a patient and uses Eq. (3) to calculate an observed score S_{obs} to a disease that has been annotated with terms of the HPO, we would like to know the probability of a randomly chosen set of q nodes achieving a score of S_{obs} or greater. In this case, each disease in the database represents a *target set* (for instance, there are currently over 5000 diseases in the clinical database used by the Phenomizer at the HPO Web site).

In other words, our methods will be used to calculate a P-value for the null hypothesis that a similarity score of S_{obs} or greater for a set of q query terms $\mathcal{Q}$ and a target set $\mathcal{T}$ has been observed by chance. We take all queries to be equally likely and define the P-value to be the proportion of queries having a score of at least S_{obs}:

$$P^{sim}_{q,\mathcal{T}}(S \geq S_{obs}) = \frac{|\{\mathcal{Q} | sim(\mathcal{Q}, \mathcal{T}) \geq S_{obs}, \mathcal{Q} = \{n_1, ..., n_q\} \subseteq V\}|}{\binom{|V|}{q}}. \quad (6)$$

In this definition all nodes of V can be part of a query, even if one node is an ancestor of the other. Note that the number of distinct scores for the complete score distribution of $P^{sim}_{q,\mathcal{T}}$ is dependent on $q, \mathcal{T}$, and the similarity measure.

Simulation of patients for clinical diagnosis

Similar to our previous work [17], we use simulations to compare different approaches. Using 1701 OMIM diseases currently annotated with 2-5 HPO terms in the *Phenotypic abnormality* subontology, we generated artificial queries by *(i)* taking all terms annotated to the disease with no noise or imprecision as the query (NONE), *(ii)* randomly exchanging one term if $q = 3$ or $q = 4$ and two terms if $q = 5$ (NOISE), *(iii)* with probability 0.5 exchange a term with one of its parent terms if possible, (IMPRECISION), or *(iv)* using first IMPRECISION then NOISE.

For each of the 1701 OMIM diseases we generate the query as described above and rank all diseases using one of the measures (Score, P-value sampled 10^3, 10^4, or 10^5 times, and P-value exact). We then calculate the rank of the disease from which the query was generated. In case of ties we take the average rank (e.g. if four diseases rank first with the same value, all four get rank 2.5). Note that for the rankings using P-values (sampled or exact) we ranked first by P-values and then by score.

Results

A naive algorithm: exhaustive computation of score distributions

We represent the score distribution as $\mathcal{SD} = \{(S_1, F_1), \ldots, (S_k, F_k)\}$. Every pair $(S_i, F_i) \in \mathcal{SD}$ contains a unique score S_i and a count F_i that defines its frequency within the distribution.

A naive approach to calculating the complete score distribution is to determine the similarity of each possible term combination $Q \subseteq V$ of size q with the fixed target set $\mathcal{T}$. The complete procedure is outlined in Algorithm 1. It requires two basic operations that are applied to the set $\mathcal{SD}$. The first operation called *getScorePair* returns the pair that represents the given score or *nil* in case the entry does not exist. The second operation denoted *putScorePair* puts the given pair into the set $\mathcal{SD}$, overwriting any previously added pair with the same score. For further analyses we assume that both operations have constant running time.

Input: V, q, $\mathcal{T}$

Output: Score distribution $\mathcal{SD} = \{(S_1, F_1), \ldots, (S_k, F_k)\}$

1 $\mathcal{SD} = \emptyset$
2 **foreach** $Q = \{n_1, n_2, ..., n_q\} \subseteq V$ **do**
3 $S_{new} \leftarrow sim^{avg}(Q, \mathcal{T})$
4 $(S, F) \leftarrow getScorePair(\mathcal{SD}, S_{new})$
5 **if** $(S, F) \neq nil$ **then**
6 $putScorePair(\mathcal{SD}, (S_{new}, F + 1))$
7 **else**
8 $putScorePair(\mathcal{SD}, (S_{new}, 1))$
9 **return** $\mathcal{SD}$

Algorithm 1: Naive score distribution computation for sim^{avg}

As the number of possible term combinations is $\binom{|V|}{q}$ and each similarity computation (line 3) costs $\mathcal{O}(q \cdot |\mathcal{T}|)$ operations for Eq. (3) Algorithm 1 runs in $\mathcal{O}(|V|^q \cdot q \cdot |\mathcal{T}|)$ time. A typical size of $|V| = 10000$ as for the HPO demonstrates that the naive approach is impractical for values $q > 2$. The naive approach neglects the relationships of the nodes in $\mathcal{G}$ and $\mathcal{T}$. We will exploit these relationships in the next section and group nodes in $\mathcal{G}$ according to their contribution to the score distribution computation.

A faster algorithm: exploiting redundant computations

Recall that all terms from the target set $\mathcal{T}$ are contained in $\mathcal{T}^{IMPL}$. We will prove now that only the *IC* values of nodes in $\mathcal{T}^{IMPL}$ are relevant for the score distribution computation.

Lemma 1. *Given a DAG* $\mathcal{G} = (V, E)$ *and a target set* $\mathcal{T} = \{n_1, \ldots, n_k\} \subseteq V$, *all scores in the score distribution of the similarity measure of Eq. (3) are derived from* IC *values of the nodes in* $\mathcal{T}^{IMPL}$.

Proof. Computing the complete score distribution involves repeatedly evaluating $sim^{avg}(\mathcal{Q}, \mathcal{T})$ in Alg. 1 using equation (3). The first step for the computation of Eq. (3) is to maximize $sim(n_1,n_2)$ for each node $n_1 \in \mathcal{Q}$ compared to nodes $n_2 \in \mathcal{T}$. The maximum *IC* value for $sim(n_1, n_2)$ must be taken from a node in $\mathcal{T}^{IMPL}$, because by definition $Anc(n_2) \subseteq \mathcal{T}^{IMPL}$.

Lemma 1 implies that the computations in the naive algorithm, which enumerates all nodes in V, are highly redundant as the size of $\mathcal{T}^{IMPL}$ is an upper bound on the number of different target set similarities encountered during score distribution computation. Figure 2 shows the contribution of all possible queries of size q = 2 for an example ontology. For instance, whenever node C or D are part of a query the target set similarity score obtained from Eq. (5) is $IC(C)$ = 4, highlighted in red in Figure 2, and used for computing $sim^{avg}(\mathcal{Q}, \mathcal{T})$.

Therefore, instead of enumerating over the nodes in V, we will first group nodes that have the same target set similarity score s in the maximization step in Eq. (3). Denote all nodes $n \in$ V that have the same target set similarity score s for a given target set $\mathcal{T}$ as N_s:

$$N_s = \{n | n \in V, sim(n, \mathcal{T}) = s\}. \quad (7)$$

Example 1. It can be seen in Figure 2 that $N_0 = \{A\}$, $N_2 = \{B\}$, and $N_4 = \{C, D\}$ for $\mathcal{G}$ with $\mathcal{T} = \{A, B, C\}$.

Observe that two nodes $n_i, n_j \in \mathcal{T}^{IMPL}, n_i \neq n_j$, belong to the same set N_s, if $IC(n_i) = IC(n_j)$. This observation will be essential when we devise an algorithm for computing N_s.

The intuition behind the fast computation is that instead of selecting combinations of all nodes of V and constructing the score distribution one by one, we focus on the combinations of different target set similarity scores s and use their frequency $|N_s|$ to avoid redundant enumeration. For any $\mathcal{T}$ the set $\mathcal{U}$ of distinct target set similarity scores is defined as:

$$\mathcal{U} = \{IC(n) | n \in \mathcal{T}^{IMPL}\}. \quad (8)$$

Instead of considering sets of nodes in V we will now consider multisets M^q of target set similarity scores in $\mathcal{U}$, where $|M^q| = q$. In order to do that we define as $\mathcal{M}$ the multiset induced by all target similarity scores s and their corresponding multiplicities m, that is,

$$\mathcal{M} = \{(s_1, m_1), \ldots, (s_d, m_d) | s_i \in \mathcal{U}, m_i = |N_{si}|\}. \quad (9)$$

Then M^q_{all} represents the set of all multi subsets of $\mathcal{M}$ that have multiset cardinality q, i.e.,

$$M^q_{all} = \{M^q | M^q \subseteq \mathcal{M}, |M^q| = q\}. \quad (10)$$

The value of sim^{avg} computed for a particular M^q is the same for all query sets of nodes that correspond to M^q (see Figure 2, Example 2). Therefore, if we can calculate the number of such sets as well as the score corresponding to each multiset M^q of target set similarity scores in $\mathcal{U}$, we can determine the distribution of similarity scores sim^{avg} for all possible queries of any given size q.

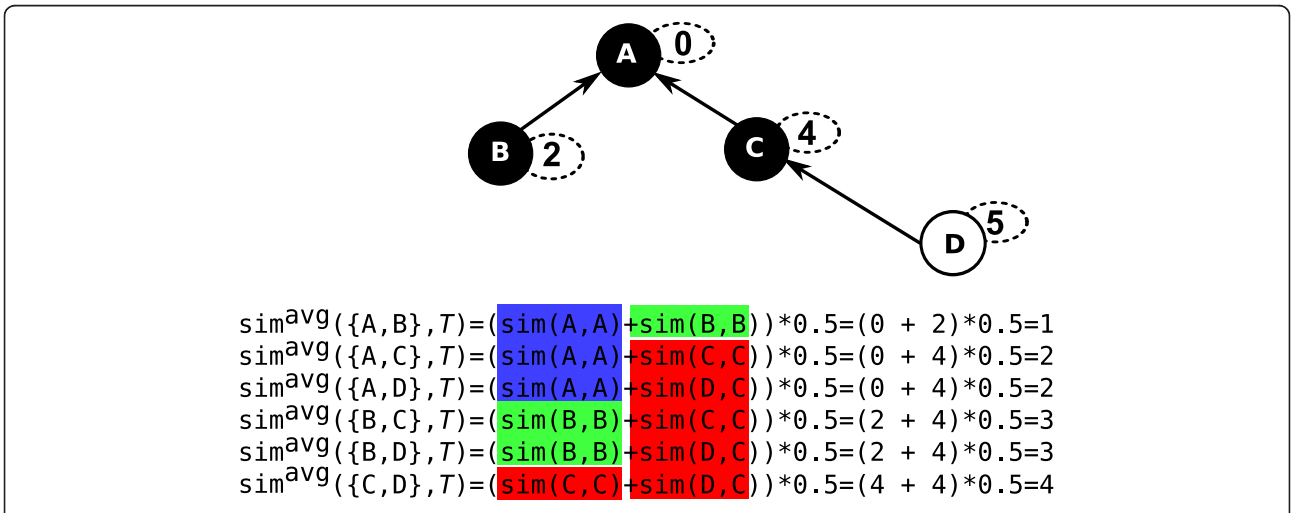

Figure 2 Redundancy in Naive Score Distribution Computation with sim^{avg} for Queries of Size Two. Computation of the score distribution for sim^{avg} on a DAG $\mathcal{G}$ with four nodes for all possible queries of size two. The target set $\mathcal{T} = \{A, B, C\}$ is shown as black nodes. Note that $\mathcal{T} = \mathcal{T}^{IMPL}$ here. The *IC* value for nodes is shown in a small dashed oval. All computations of Eq. (5) that result in the same target similarity score are colored in blue, green, and red for the target set similarity scores 0, 2, and 4, respectively.

Denote the similarity for a multiset M^q as:

$$sim^{avg}(M^q) = \frac{1}{q} \sum_{(s,m)\in M^q} m \cdot s. \quad (11)$$

The number of ways of drawing m nodes from a component of size $|N_s|$ can be calculated using the binomial coefficient. The total number of combinations is then the product of all binomial coefficients, denoted as the *multiset frequency* for a multiset M^q:

$$freq(M^q) = \prod_{(s,m)\in M^q} \binom{|N_s|}{m}. \quad (12)$$

Example 2. In total there are 2 query sets with $sim^{avg}(\mathcal{Q}, \mathcal{T}) = 2$ for the DAG in Figure 2, namely {*A*, C}, {*A, D*}. After preprocessing, we obtain N_0 = {A}, N_2 = {*B*}, and N_4 = {*C, D*} (Example 1). Alg. 2 enumerates all valid multisets of cardinality 2 for the sets N_s considering their size $|N_s|$. The only way of attaining an average score of 2 is to select one node out of N_0 and N_4, represented by the multiset M^2 = {(0,1), (4,1)} for which $sim^{avg}(M^2) = 2$. The multiset frequency of M^2 gives the same result as shown in Figure 2, $freq(M^2) = \binom{|N_0|}{1} \cdot \binom{|N_4|}{1} = 1 \cdot 2 = 2$. Instead of iterating over two sets we consider one multiset.

Theorem 1. *Let* $\mathcal{SD} = \{(S_1, F_1), \ldots, (S_k, F_k)\}$*be the score distribution computed with* sim^{avg} *for an ontology DAG* $\mathcal{G} = (V, E)$*, target set* $\mathcal{T} \subseteq V$*and query size q. The frequency F with which any given score S occurs amongst all possible queries of size q is then*:

$$F = \sum_{M^q \in M^q_{all},\ sim^{avg}(M^q)=S} freq(M^q). \quad (13)$$

A proof of Theorem 1 is provided in Appendix A and a faster algorithm based on Theorem 1 is shown in Alg. 2. We enumerate all distinct multisets of M^q_{all} and add their frequency to the score distribution $\mathcal{SD}$, instead of iterating over all sets of size q in Alg. 1, thereby reducing the number of operations. In order to apply the algorithm to score distribution computation for sim^{max}, line 3 of Alg. 2 needs to be replaced. Instead of computing the average of all scores in the multiset, the maximum among them is assigned to S_{new}.

Preprocessing of the DAG for faster computation

So far we have neglected how we can compute the values $|N_s|, s \in \mathcal{U}$ but we will introduce an efficient algorithm in this section. We denote the algorithm as preprocessing because computation of $|N_s|$ is independent of q. The preprocessing will divide the original graph into a set of connected components from which the $|N_s|$ values can be deduced.

Input: M^q_{all}

Output: Score distribution $\mathcal{SD} = \{(S_1, F_1), \ldots, (S_k, F_k)\}$

1 $\mathcal{SD} = \emptyset$
2 **foreach** *multiset* $M^q \in M^q_{all}$ **do**
3 $S_{new} \leftarrow sim^{avg}(M^q)$
4 $(S, F) \leftarrow getScorePair(\mathcal{SD}, S_{new})$
5 **if** $(S,F) \neq nil$ **then**
6 $putScorePair(\mathcal{SD}, (S_{new}, F + freq(M^q)))$
7 **else**
8 $putScorePair(\mathcal{SD}, (S_{new}, freq(M^q)))$
9 **return** $\mathcal{SD}$

Algorithm 2: Faster score distribution computation for sim^{avg}

First, we invert the direction of all edges in E such that the edges are directed from the root towards the leaves of the DAG, and introduce edge weights $w_{i,j}$ to the edges of $\mathcal{G}$. Let

$$w_{i,j} = \begin{cases} IC(n_i), & \text{if } n_i \in \mathcal{T}^{IMPL} \\ \max\{w_{h,i} | e_{h,i} \in E\} & \text{otherwise} \end{cases}. \quad (14)$$

The edge weights are defined in a recursive manner. First, all weights of edges emerging from nodes in $\mathcal{T}^{IMPL}$ are set. Then the maximum edge weight of all incoming edges for each node not in $\mathcal{T}^{IMPL}$ are propagated to all outgoing edges of the node, and as such propagated throughout the graph. Computing the edge weights is efficiently done after the nodes of $\mathcal{G}$ have been sorted in topological order, see Alg. 3. We now iterate across all nodes $n_i \in V$. For each node $n_i \in V, n_i \notin \mathcal{T}^{IMPL}$, there is at least one path that leads to the node $n_j = \arg\max_{n_k \in \mathcal{T}} sim(n_i, n_k)$. If a node has multiple parents, then by construction of the edge weights, an edge with a maximum weight will be a member of a path to n_j. We therefore remove all other incoming edges. If there are multiple incoming edges with an identical, maximum edge weight, one of them can be chosen arbitrarily and the others are removed (Alg. 3, lines 7-9). We now iterate over all remaining edges $e_{i,j}$ and remove all edges for which $n_i, n_j \in \mathcal{T}^{IMPL}$ holds (Alg. 3, lines 10-12). Note that exactly $|\mathcal{T}^{IMPL}|$ many connected components C_i one for each $n_i \in \mathcal{T}^{IMPL}$ remain.

For all pairs of connected components such that $IC(n_i) = IC(n_j)$ for $n_i, n_j \in \mathcal{T}^{IMPL}, n_i \neq n_j$, the connected components C_i and C_j are merged to arrive at the desired sets $N_s, s \in \mathcal{U}$ (Alg. 3, lines 13-16).

All these steps are summarized in Alg. 3 and Figure 3.

Theorem 2. *Given a DAG* $\mathcal{G} = (V, E)$*and a target set* $\mathcal{T} = \{n_1, \ldots, n_k\} \subseteq V$*the score distribution of Eq. (3) is computed by Alg.* 2 *and Alg.* 3 *in* $\mathcal{O}(|E| + |V| + M(|\mathcal{T}^{IMPL}|, q))$*time and space.*

Proof. The preprocessing of the *DAG* in Alg. 3 involves inverting edges, topological ordering of V,

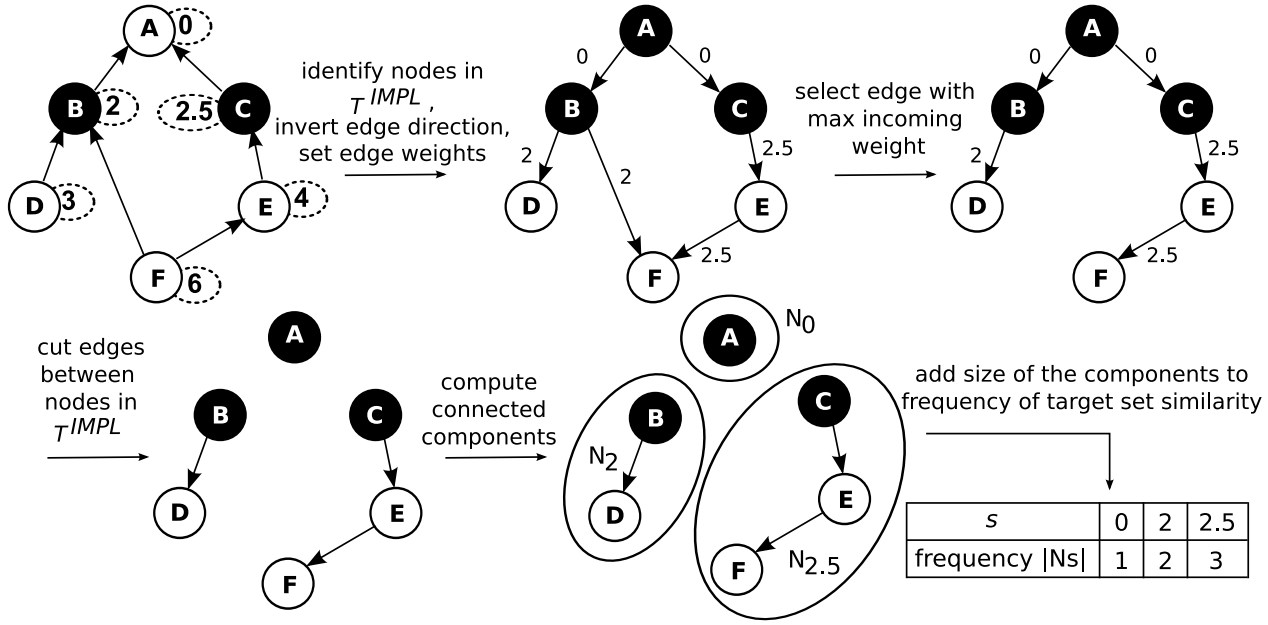

Figure 3 Overview of the Algorithm for Preprocessing. The general steps of Alg. 3 are shown on the DAG and $\mathcal{T}$ of Figure 1. Nodes in $\mathcal{T}^{IMPL}$ are colored in black. The *IC* value of a node is depicted in a dashed oval.

Input: V, $\mathcal{T}^{IMPL}$

Output: node sets with identical target similarity score, i.e., N_s

1 **for** $n_i \in V$ *in topological order* **do**
2 **for** j *in* $e_{i,j} \in E$ **do** /* Set weights */
3 **if** $n_i \in \mathcal{T}^{IMPL}$ **then**
4 $w_{i,j} \leftarrow IC(n_i)$
5 **else**
6 $w_{i,j} \leftarrow \max\{w_{h,I} \mid e_{h,i} \in E\}$
7 **for** $n_i \in V \setminus root$ **do**
8 choose $e_{h,i} \in E$ s.t. $|w_{h,i} \geq w_{h',i}$ for all edges $e_{h',i} \in E$
9 remove all incoming edges of n_i except $e_{h,i}$
10 **for** $e_{i,j} \in E$ **do** /* Connected components C_i */
11 **if** $n_i, n_j \in \mathcal{T}^{IMPL}$ **then**
12 remove $e_{i,j}$ from E
13 **for** $s \in \{IC(n_i) | n_i \in \mathcal{T}^{IMPL}\}$ **do** /* Merging */
14 $N_s = \emptyset$
15 **foreach** $n_i \in \mathcal{T}^{IMPL}$ **do**
16 $N_s \leftarrow N_s \cup C_i$
17 **return** N_s

Algorithm 3: Graph preprocessing for faster computation

introducing edge weights to E, removing edges in E, and computing the connected components of $\mathcal{G}$. This can be done with depth-first search (DFS) traversals of $\mathcal{O}(|E| + |V|)$ with to a worst-case performance of $\mathcal{O}(|E| + |V|)$ time and space.

Algorithm 2 runs in $\mathcal{O}(M(|\mathcal{T}^{IMPL}|, q))$ time and space. The outer *foreach* loop runs over all distinct multisets with cardinality q. The multiset coefficient $M(|\mathcal{T}^{IMPL}|, q)$ provides an upper bound for the number of these multisets. In each iteration the computation of the similarity score (line 3) and the multiset frequency, $freq(M^q)$, have constant cost assuming a preprocessed lookup table for binomial coefficients and if common partial sim^{avg} values are stored between the iterations, avoiding recomputation for similar multisets. In total, Alg. 2 and Alg. 3 run in $\mathcal{O}(|E| + |V| + M(|\mathcal{T}^{IMPL}|, q))$ time and space.

The theorem concludes the improvement to the naive algorithm, for example on average $|\mathcal{T}^{IMPL}| \sim 38$ for all diseases currently annotated with terms of the HPO, which currently has approximately 10000 terms and 13000 relations. For instance, for a query with 5 terms, the naive algorithm would thus run in time proportional to $10000^5 \cdot 5 \cdot 38 = 1.9 \times 10^{22}$, and the new algorithm in time proportional to $9000 + 11000 + 5 \cdot M(38, 5) = 4.3 \times 10^6$.

Experiments

We now show the results of the new algorithm applied to the HPO [7]. In our previous work we implemented the Phenomizer as a system for experts in the differential diagnosis in medical genetics; the Phenomizer can be queried with a set of HPO terms to get a ranked list of

candidate diseases most similar to the query based on *P*-values derived from Resnik similarity scores, Eq. (3) [17]. However, for the Phenomizer we used Monte Carlo sampling to approximate the score distribution and we will investigate now the difference in using the exact *P*-value compared to sampling.

As we are interested in ranking diseases for differential diagnosis we will take a similar simulation approach as in [17] and generate sets of artificial patients for which we know the OMIM disease, see Methods. In Figure 4 the results are shown for the investigated scenarios NONE, NOISE, IMPRECISION, and NOISE + IMPRECISION. We compared the ranking of patients with the similarity score alone, sampling based *P*-values (10^3 - 10^5 repetitions, the latter used in the Phenomizer), and exact computation using the algorithm in this work. In

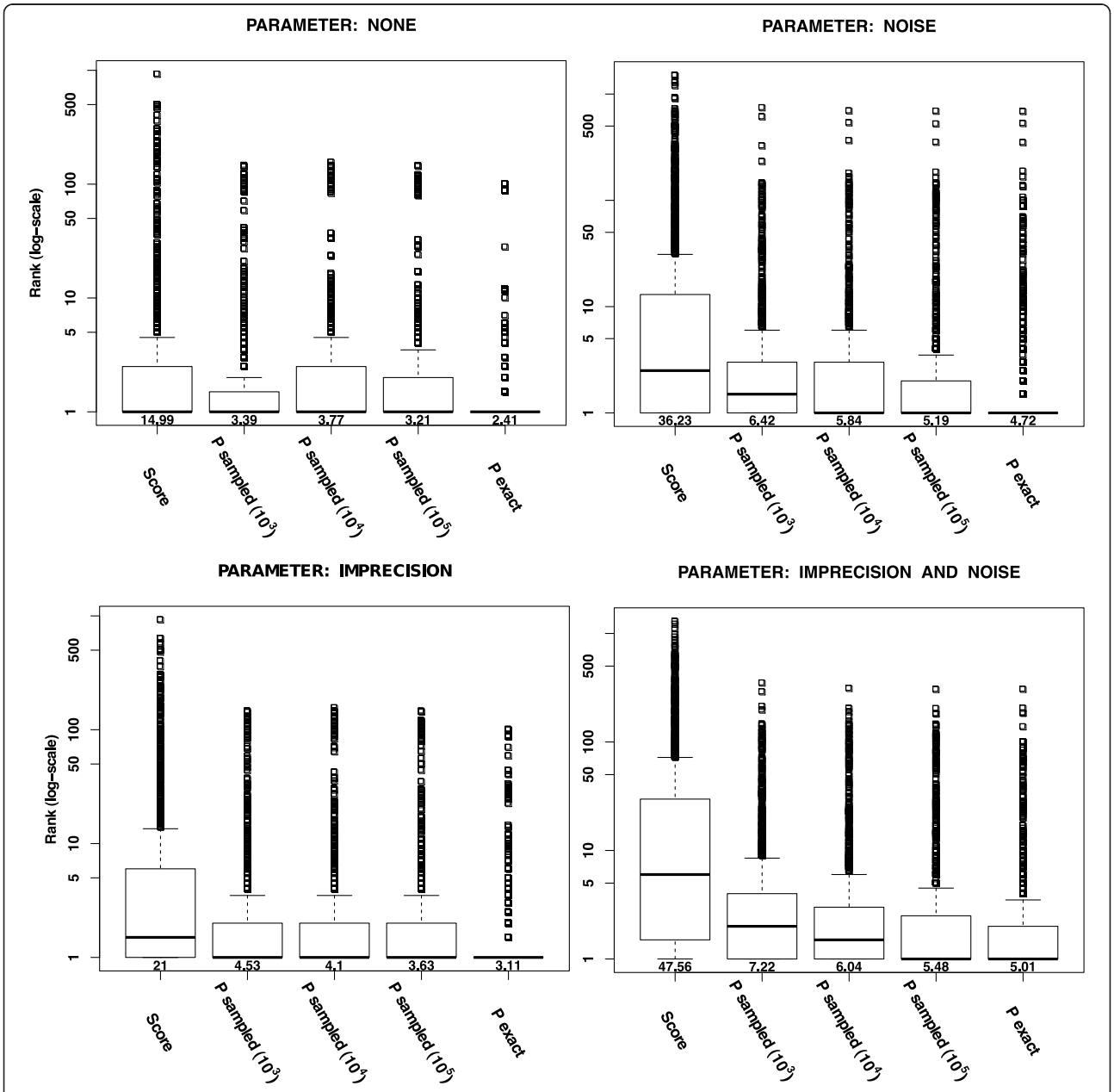

Figure 4 Impact of Exact *P*-value Computation for Clinical Diagnostics with the HPO. Simulations for Clinical Diagnostics using the HPO. Patient (phenotype) data was simulated and queried against the complete database of all 4992 annotated diseases. The best result is obtained if the original disease is assigned the rank one (y-axis) by the search algorithm. Different approaches are compared (x-axis). Data were generated without error NONE and with NOISE (top row, left and right) and with IMPRECISION and both IMPRECISION and NOISE (bottom row, left and right) as explained in the Methods section. The mean rank is shown below each boxplot.

all cases, using the exact *P*-value computation significantly outperforms the four alternative ranking methods (Mann-Whitney *P*-value < 0.001) and ranks the true disease on rank one most of the time. The improvement for the exact score distribution computation is due to the fine-grained resolution especially for small *P*-values, where sampling is often underrepresented, but which are important for selecting the best rank (see Additional File 1).

We then investigated the runtime for different q values as compared to using the naive algorithm and Monte Carlo sampling (Table 1). For that purpose we selected four diseases with a different number of annotated HPO terms, and therefore different size of $\mathcal{T}^{IMPL}$, and show the runtime of the three approaches in milliseconds. The naive algorithm cannot be utilized for $q > 2$. The exact *P*-value computation is faster than random sampling with 10^5 repetitions for $q = 2,3$ and for the disease with only 17 terms in $\mathcal{T}^{IMPL}$ independent of the analyzed q. Starting from $q = 4$ the sampling based approach is faster for large $|\mathcal{T}^{IMPL}|$ because of the huge size of the score distribution to be computed, but even for $q = 5$ the complete score distribution can be computed in under 4 seconds for diseases with many annotations. Note again that the average size of $\mathcal{T}^{IMPL}$ is 38 in the HPO.

Table 1 Runtime in milliseconds averaged over 20 runs comparing the naive, exact, and sampled distribution computation for q = 2,3,4, and 5

Runtime Analysis with the HPO						
				runtime in milliseconds		
OMIM ID	$\lvert\mathcal{T}\rvert$	$\lvert\mathcal{T}^{IMPL}\rvert$	$\lvert\mathcal{U}\rvert$	**naive**	**exact**	**sampled***
$q = 2$						
264300	5	17	16	3779	4	50
613124	7	36	36	3794	6	53
113450	12	80	72	3789	6	65
129500	20	66	61	3702	15	89
$q = 3$						
264300	5	17	16	$\sim 1.2 \cdot 10^7$	4	49
613124	7	36	36	$\sim 1.2 \cdot 10^7$	6	53
113450	12	80	72	$\sim 1.2 \cdot 10^7$	19	66
129500	20	66	61	$\sim 1.2 \cdot 10^7$	15	79
$q = 4$						
264300	5	17	16	-	5	46
613124	7	36	36	-	20	55
113450	12	80	72	-	250	65
129500	20	66	61	-	135	77
$q = 5$						
264300	5	17	16	-	7	48
613124	7	36	36	-	141	54
113450	12	80	72	-	3896	63
129500	20	66	61	-	1776	79

Four OMIM diseases with a varying number of annotated HPO terms ($|\mathcal{T}|$) were used; 264300: 17-β Hydroxysteroid Dehydrogenase III deficiency, 613124: Hydrops fetalis, nonimmune, with gracile bones and dysmorphic features, 113450: Brachydactyly-distal symphalangism syndrome, 129500: Ectodermal dysplasia 2, hidrotic. Entries denoted "-" were terminated after four hours. *Sampling with 10^5 repetitions.

Discussion

In this work we have tackled the unstudied problem of computing the score distribution for similarity searches with ontologies. We have devised an efficient preprocessing of the underlying DAG of the ontology that reduces the complexity for similarity measures based on Resnik's popular definition of similarity [19]. We have introduced a new algorithm based on multiset enumeration, which can be applied to score distribution computation for Eq. (3) as well as variants based on maximum similarity Eq. (4). In experiments with the HPO, as well as in theory, we show that the new algorithm is much faster than exhaustive enumeration of the score distribution or resampling approaches and that it is applicable to current ontologies.

The algorithm we describe here can be used as a component of a procedure to find the best hit in a database, i.e., we need to calculate the score for each entry in the database and rank the results according to *P*-value. This allows users to enter a list of characteristics or features in order to identify objects whose characteristics best match the query using semantic similarity. We have implemented our algorithm in the setting of medical diagnostics, where the features are the signs and symptoms of diseases and the domain objects are diseases. We have previously shown that this kind of search is useful for medical differential diagnosis [17].

Summarizing all nodes that have the same target set similarity score makes use of the fact that the pairwise similarity defined by Resnik only considers the common ancestors of the relevant terms (Lemma 1). Extending the proposed algorithm for other popular semantic similarity measures based on the information content of a node, like Jiang and Conrath or Lin [20,21], or the symmetric definition of Eq. (3) [12], has not been considered here as definition of pairwise similarity additionally incorporates the information content of the nodes in the query. Therefore, additional steps are necessary which render the computations more complicated. Although this can be considered a limitation of the current approach, we believe the methodology introduced here will prove useful for other measures as well. For example the term overlap similarity measure [22], comparably, only considers common ancestors of query and target set terms, thus an algorithm with similar complexity appears possible from the results presented in this paper. One of the reasons why the *P*-value based rankings outperform the rankings based on scores is that the former account for

the annotation bias as observed by Wang et al. [23]. The best-match average semantic similarity measures based on Resnik, like Eq. (3), were shown to have a strong bias. The annotation bias is a further argument to use *P*-values instead of the similarity scores alone.

In the mentioned study by Wang et al. [23], the authors consider the comparison of two proteins via their annotated GO terms, instead of considering any possible subset of the ontology terms as query as in our search setup. Their approach is to compensate for the annotation bias by simulating the distribution of pairwise similarity scores for all annotated ontology term sets and normalizing using a power transformation. Similarly to our experiments, their method might improve when the exact score distribution is computed using our algorithm.

In a practical implementation of our algorithm, the *P*-values could be precomputed for each entry in the database (such as all the diseases in OMIM or each protein in the human proteome). For small q, the *P*-values could be calculated dynamically. This might be useful if users are allowed to filter out portions of the database from the search based on some predefined groups (for instance, in genetics, the differential diagnosis might be restricted to diseases showing a certain mode of inheritance).

Due to its simple structure the new algorithm could be parallelized to run with several threads with close to linear speedup, by keeping the scores in different hash structures for each thread and merging all hashes at the end to get the complete distribution. Also, as often only the *P*-value is of interest, a branch and bound formulation of the new algorithm might lead to a significant speedup in practice.

Conclusions

The algorithmic improvement reported here might prove useful for *P*-value computation of other semantic similarity measures that are based on the information content of a node as introduced by Resnik [12]. However, when the similarity score includes more dependencies the size of the complete score distribution may increase significantly. Further algorithmic development will be necessary to increase the class of similarity measures for which *P*-values can be computed efficiently.

We believe that our methods would be applicable to other applications in which users search for domain objects that best exemplify a set of desired attributes and that they can be used to improve bioinformatic methods that use the semantic similarity scores alone. For that purpose we implemented a software in Java that computes exact score distributions for both similarity measures discussed here. The software works with any ontology available in OBO format and is available for non-commercial and academic usage under: https://compbio.charite.de/svn/hpo/trunk/src/tools/significance/

Appendix A

In this Appendix, we will prove Theorem 1 for arbitrary q. In the following text, we will outline the approach of the proof and introduce a few new definitions. We can calculate the *P*-values, Eq. (6), by computing the frequency F_i of each score S_i in the score distribution, i.e., by calculating the number of queries that result in score S_i for each possible score. We will consider all query sets $\mathcal{Q}$ that result in score S, denoted as $\mathcal{Q}_S$ later in Eq. (15). These initial query sets consist of the nodes from the Ontology DAG $\mathcal{G} = (V, E)$. Subsequently, we will substitute sets of nodes $\mathcal{Q}$ by multisets $M^q(\mathcal{Q})$ over their target set similarity scores in Eq. (16). This is the important switch that establishes the independence of the number of nodes in the graph by only considering their target set similarity scores. At this step, changing from sets to multisets is necessary, because the same target set similarity score may occur more than once given nodes in a single $\mathcal{Q}$. However, the induced multisets from all sets in $\mathcal{Q}_S$ are themselves not unique and therefore we will use the multiset frequency, Eq. (12), over the set of unique multisets M_S^q given $\mathcal{Q}_S$ to compute the desired quantity F in the proof.

We are interested in the set $\mathcal{Q}_S$ of all sets $\{n_1,..., n_q\}$ of nodes $\{n_1,..., n_q\} \subseteq V$, which result in the same average score S. That is, $\mathcal{Q}_S$ is the set of all queries of size q that result in the same average score S:

$$\mathcal{Q}_S = \{\{n_1, \ldots, n_q\} | \{n_1, \ldots, n_q\} \subseteq V, sim^{avg}(\{n_1, \ldots, n_q\}, \mathcal{T}) = S\}. \quad (15)$$

The core message of Theorem 1 is that we can define a multiset M^q over the target set similarity scores s whose frequency can be used to compute the frequency F of scores S in the score distribution. A necessary first step therefore is to express a query set $\mathcal{Q} = \{n_1, \ldots, n_q\} \subseteq V$ as a multiset $M^q(\mathcal{Q})$:

$$M^q(\mathcal{Q}) = \{(s_1, m_1), \ldots, (s_o, m_o) | s_i \in U_{\mathcal{Q}}, m_i = m_{s_i}^{\mathcal{Q}}\}, \quad (16)$$

where

$$U_{\mathcal{Q}} = \{s_i | n_i \in \mathcal{Q}, sim(n_i, \mathcal{T}) = s_i\} \quad (17)$$

and

$$m_{s_i}^{\mathcal{Q}} = |\{n_i | n_i \in \mathcal{Q}, sim(n_i, \mathcal{T}) = s_i\}|. \quad (18)$$

The underlying set $U\mathcal{Q}$ for a multiset $M^q(\mathcal{Q})$ consists of all existing distinct target set similarity scores s_i of the nodes in $\mathcal{Q}$, Eq. (17), and their multiplicity is the number of nodes in $\mathcal{Q}$ that share the same score s_i, Eq. (18).

Now that we know how to create a multiset of target set similarity scores from any given set of nodes in V, we need another variable M_S^q to represent all distinct multisets that can be generated using Eq. (16) from the set $\mathcal{Q}_S$. The set of distinct multisets M_S^q generated for a

given $\mathcal{Q}_S$ is defined as:

$$M_S^q = \{M^q(\mathcal{Q}) | \mathcal{Q} \in \mathcal{Q}_S\}. \quad (19)$$

We can now state the proof of Theorem 1 as follows.
Proof.

$$F = |\mathcal{Q}_S| \quad (20)$$

$$= \sum_{M^q \in M_S^q} \prod_{(s,m) \in M^q} \binom{|N_s|}{m} \quad (21)$$

$$= \sum_{M^q \in M_S^q} freq(M^q) \quad (22)$$

$$= \sum_{M^q \in M_{all}^q,\ sim^{avg}(M^q)=S} freq(M^q) \quad (23)$$

Eq. (20) merely restates the definition of the Frequency F given by Eq. (15), namely the number of all queries $\mathcal{Q} \subseteq V$ that result in $sim^{avg} = S$. Note that Eq. (15) is representing the number of such queries in terms of sets of nodes of the ontology. Eq. (21) switches the representation from nodes in V to multisets M_S^q over the similarity scores of nodes in V using Eq. (19) and the definition of multiset frequency given in Eq. (12). Eq. (22) follows directly from the definition of the multiset frequency in Eq. (12). The equality between Eq. (22) and (23) is a direct consequence of Eq. (15) and (19).

Additional material

Additional file 1: Additional File 1contains some additional plots showing the differences in ranking by exact and sampled *P*-values for Clinical Diagnostics with the HPO.

Acknowledgements
We thank Martin Vingron for his insights that led to an earlier version of this manuscript. The authors would also like to thank the two anonymous reviewers for insightful comments.
Funding
MHS was funded by the International Max Planck Research School for Computational Biology and Scientific Computing. SK and PNR were supported by the Berlin-Brandenburg Center for Regenerative Therapies (BCRT) (Bundesministerium für Bildung und Forschung, project number 0313911). SB and PNR were supported by the Deutsche Forschungsgemeinschaft (DFG RO 2005/4-1).

Author details
[1]Max Planck Institute for Molecular Genetics, Ihnestr. 73, 14195 Berlin, Germany. [2]Ray and Stephanie Lane Center for Computational Biology, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, 15213 Pennsylvania, USA. [3]Institute for Medical Genetics and Human Genetics, Charité-Universitätsmedizin Berlin, Augustenburger Platz 1, 13353 Berlin, Germany. [4]Berlin-Brandenburg Center for Regenerative Therapies (BCRT), Charité-Universitätsmedizin Berlin, Augustenburger Platz 1, 13353 Berlin, Germany.

Authors' contributions
MHS, SK, and PNR planned the research work. MHS, and SB designed the algorithm. MHS, SK, and SB implemented the software and made the analysis. All authors wrote the paper and approved the final manuscript.

Received: 20 May 2011 Accepted: 12 November 2011
Published: 12 November 2011

References

1. Robinson PN, Bauer S: *Introduction to Bio-Ontologies* Chapman & Hall/CRC Mathematical & Computational Biology., Chapman & Hall; 2011.
2. Rosse C, Mejino JLV: **A reference ontology for biomedical informatics: the Foundational Model of Anatomy.** *J Biomed Inform* 2003, **36(6)**:478-500.
3. Eilbeck K, Lewis SE, Mungall CJ, Yandell M, Stein L, Durbin R, Ashburner M: **The Sequence Ontology: a tool for the unification of genome annotations.** *Genome Biol* 2005, **6(5)**:R44.
4. Bard J, Rhee SY, Ashburner M: **An ontology for cell types.** *Genome Biol* 2005, **6(2)**:R21.
5. Degtyarenko K, de Matos P, Ennis M, Hastings J, Zbinden M, McNaught A, Alc'antara R, Darsow M, Guedj M, Ashburner M: **ChEBI: a database and ontology for chemical entities of biological interest.** *Nucleic Acids Res* 2008, **36(Database issue)**:D344-D350.
6. Smith CL, Goldsmith CAW, Eppig JT: **The Mammalian Phenotype Ontology as a tool for annotating, analyzing and comparing phenotypic information.** *Genome Biol* 2005, **6**:R7.
7. Robinson PN, Köhler S, Bauer S, Seelow D, Horn D, Mundlos S: **The Human Phenotype Ontology: a tool for annotating and analyzing human hereditary disease.** *Am J Hum Genet* 2008, **83(5)**:610-615.
8. Hancock JM, Mallon AM, Beck T, Gkoutos GV, Mungall C, Schofield PN: **Mouse, man, and meaning: bridging the semantics of mouse phenotype and human disease.** *Mamm Genome* 2009, **20(8)**:457-461.
9. Pesquita C, Faria D, Falcao AO, Lord P, Couto FM: **Semantic similarity in biomedical ontologies.** *PLoS Comput Biol* 2009, **5(7)**:e1000443.
10. Yu H, Jansen R, Stolovitzky G, Gerstein M: **Total ancestry measure: quantifying the similarity in tree-like classification, with genomic applications.** *Bioinformatics* 2007, **23(16)**:2163-2173.
11. Lord PW, Stevens RD, Brass A, Goble CA: **Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation.** *Bioinformatics* 2003, **19(10)**:1275-1283.
12. Couto F, Silva MJ, Coutinho PM: **Measuring Semantic Similarity between Gene Ontology Terms.** *Data and Knowledge Engineering, Elsevier* 2007, **61.**
13. Joshi T, Xu D: **Quantitative assessment of relationship between sequence similarity and function similarity.** *BMC Genomics* 2007, **8**:222.
14. Sevilla JL, Segura V, Podhorski A, Guruceaga E, Mato JM, Martínez-Cruz LA, Corrales FJ, Rubio A: **Correlation between gene expression and GO semantic similarity.** *IEEE/ACM Trans Comput Biol Bioinform* 2005, **2(4)**:330-338.
15. Xu T, Du L, Zhou Y: **Evaluation of GO-based functional similarity measures using S. cerevisiae protein interaction and expression profile data.** *BMC Bioinformatics* 2008, **9**:472.
16. Lei Z, Dai Y: **Assessing protein similarity with Gene Ontology and its use in subnuclear localization prediction.** *BMC Bioinformatics* 2006, **7**:491.
17. Köhler S, Schulz MH, Krawitz P, Bauer S, Dölken S, Ott CE, Mundlos C, Horn D, Mundlos S, Robinson PN: **Clinical diagnostics in human genetics with semantic similarity searches in ontologies.** *Am J Hum Genet* 2009, **85(4)**:457-464.
18. Resnik P: **Using Information Content to Evaluate Semantic Similarity in a Taxonomy.** *Proceedings of the 14th International Joint Conference on Artificial Intelligence* 1995, 448-453.
19. Resnik P: **Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language.** *Artificial Intelligence Research* 1999, **11**:95-130.
20. Jiang J, Conrath D: **Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy.** *Proc of the 10th International Conference on Research on Computational Linguistics* 1997, **10.**
21. Lin D: **An information-theoretic definition of similarity.** *Proc of the 15th International Conference on Machine Learning* 1998, **15.**
22. Mistry M, Pavlidis P: **Gene Ontology term overlap as a measure of gene functional similarity.** *BMC Bioinformatics* 2008, **9**:327.

23. Wang J, Zhou X, Zhu J, Zhou C, Guo Z: **Revealing and avoiding bias in semantic similarity scores for protein pairs.** *BMC Bioinformatics* 2010, **11**:290.
24. Schulz MH, Köhler S, Bauer S, Vingron M, Robinson PN: **Exact Score Distribution Computation for Similarity Searches in Ontologies.** In *Algorithms in Bioinformatics, 9th International Workshop, WABI 2009. Volume 5724.* Edited by: Warnow T, Salzberg S. Springer LNBI; 2009.
25. Blizard WD: **Multiset Theory.** *Notre Dame Journal of Formal Logic* 1989, **30**:36-66.

doi:10.1186/1471-2105-12-441
Cite this article as: Schulz *et al.*: Exact score distribution computation for ontological similarity searches. *BMC Bioinformatics* 2011 **12**:441.