True Path Rule and H-Bayes hierarchical cost-sensitive ensembles for gene function prediction

Giorgio Valentini
e-mail: valentini@dsi.unimi.it

DSI - Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano

Outline

- The Gene Function Prediction (GFP) problem
- The True Path Rule (TPR) ensemble algorithm
- The Hierarchical Bayesian (H-BAYES) ensemble algorithm
- Integration of H-BAYES and data fusion methods

Hierarchical classification problems

- Several interesting real-world classification problems are characterized by hierarchical relationships between classes
- E.g. : textual classification of web content [Rousu et al., 2005], music categorization and semantic scene classification [Tsoumakas et al. 2009], bioinformatics (Barutcuoglu et al. 2006).
- Different approaches: a) methods restricted to multilabels with single and no partial paths [Dekel et al. 2004]; b)methods extended to multiple and also partial paths [Cesa-Bianchi et al. 2006].

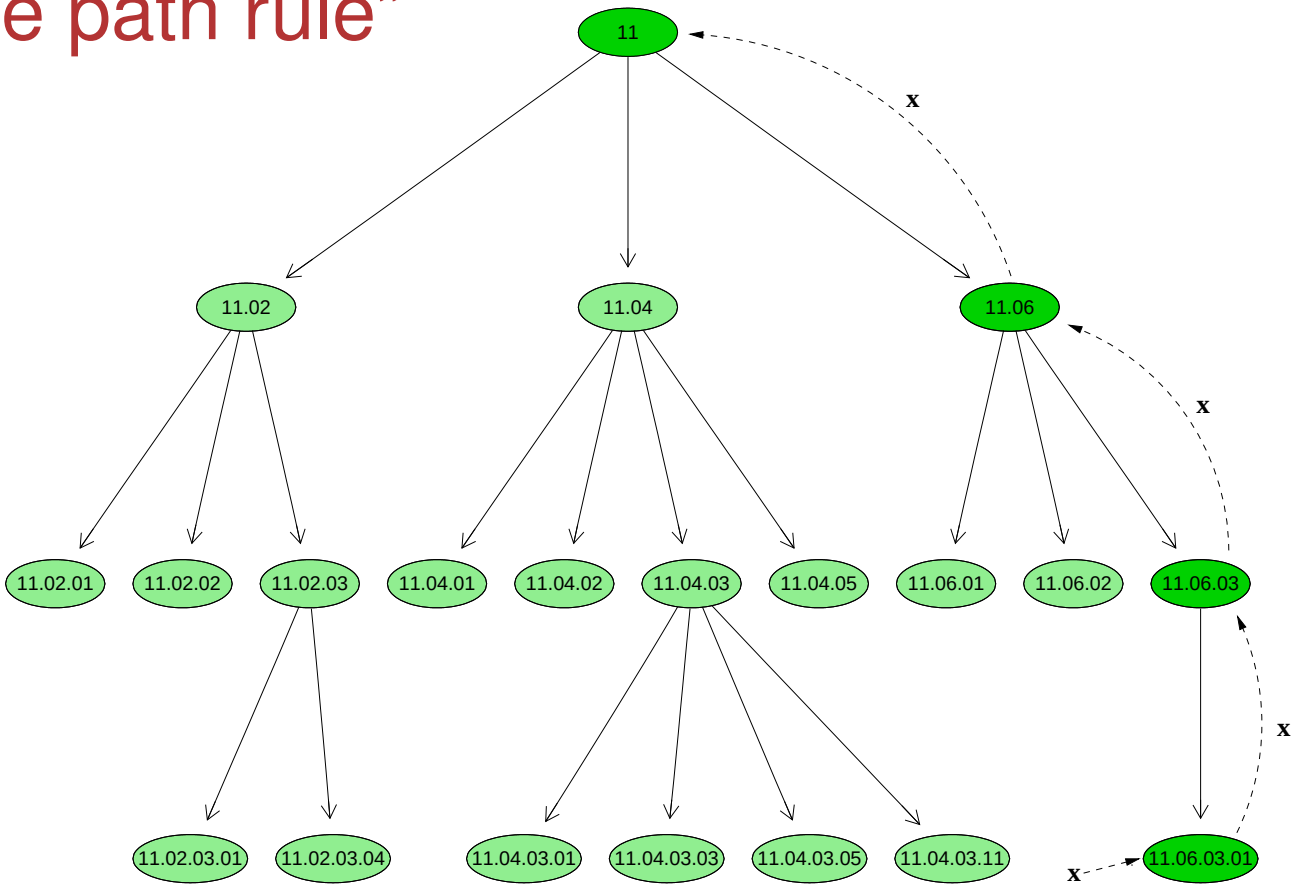
Genome and ontology-wide GFP

- Novel high-throughput biotechnologies accumulated a wealth of data about genes and gene products
- Manual annotation of gene function is time consuming and expensive and becomes infeasible for growing amount of data.
- For most species the functions of several genes are unknown or only partially known: “in silico” methods represent a fundamental tool for gene function prediction at genome-wide and ontology-wide level [Friedberg, 2006].
- Computational analysis provide predictions that can be considered hypotheses to drive the biological validation of gene function [Pena-Castillo et al. 2008].

Main characteristics of the GFP problem

- Large number of functional classes: hundreds (*FunCat*) or thousands (*Gene Ontology (GO)*).
- Multiple annotations for each gene (multilabel classification)
- Hierarchical relationships between functional classes (tree forest for FunCat, direct acyclic graph for GO)
- Different level of evidence for functional annotations
- Class frequencies are unbalanced: positive examples are usually largely lower than negatives
- Different strategies can be applied to choose negative examples
- Multiple sources of data available: each type captures specific functional characteristics of genes/gene products

The “true path rule”



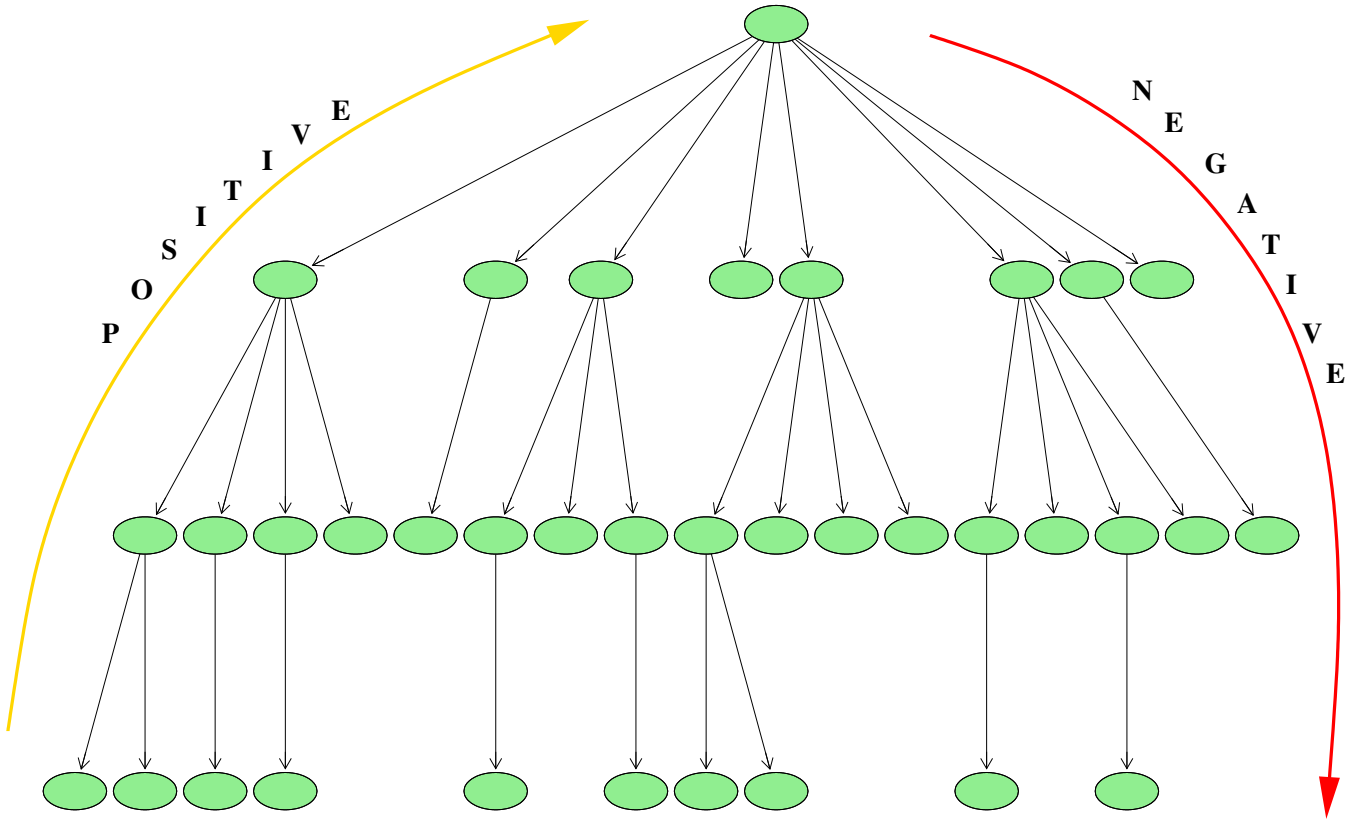
“An annotation for a class in the hierarchy is automatically transferred to its ancestors, while genes unannotated for a class cannot be annotated for its descendants”.

TPR: an algorithm that obeys the True Path Rule

(Valentini, 2010)

1. Training of the base learners (1 for each node)
2. Evaluation phase: the trained classifiers provide a local decision for each node/class
3. Positive decisions may propagate from bottom to top across the graph: they influence the decisions of the parent nodes and of their ancestors.
4. Negative decisions do no affect decisions of the parent node
5. Negative predictions for a given node are propagated to the descendants. Positive decisions do not influence decisions of child nodes.

TPR ensembles: an asymmetric flow of information



From bottom to top : positive predictions influence ancestor nodes/classifiers

From top to bottom : negative predictions influence descendant nodes/classifiers

Basic notation and definitions

- A gene/gene product x can be assigned to one or more functional classes:
  $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$
- Assignments can be coded through a vector of multilabels
  $\mathbf{y} =< y_1, y_2, \ldots, y_m > \in \{0,1\}^m$.
  If x belongs to class c_i, then $y_i = 1$, otherwise $y_i = 0$.
- Nodes corresponding to the class c_i can be simply denoted by i.
- $\mathrm{child}(i)$ represents the set of children nodes of i;
  $\mathrm{par}(i)$ the set of its parents.
- The TPR ensemble classifier $D : X \to \{0,1\}^m$ computes the multilabel associated to each gene $x \in X$
- $d_i(x) \in \{0,1\}$ is the label predicted by the TPR classifier at node i. If there is no ambiguity we represent $d_i(x)$ simply by d_i.

The rules a TPR ensemble must obey

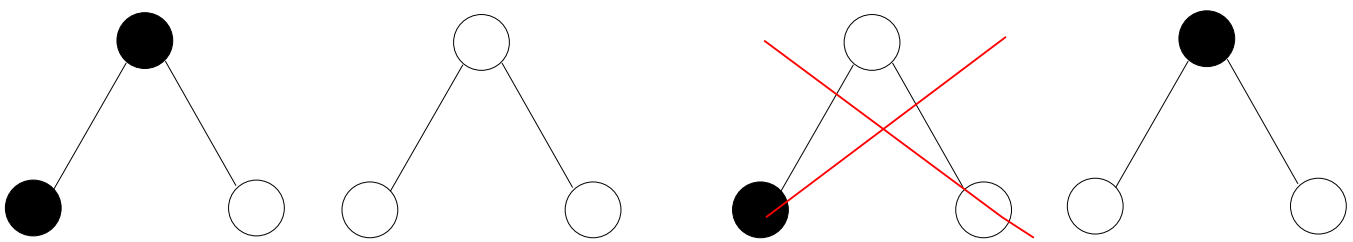
Considering the parents of a given node i:

$$\begin{cases} d_i = 1 & \Rightarrow \quad d_{par(i)} = 1 \\ d_i = 0 & \not\Rightarrow \quad d_{par(i)} = 0 \end{cases}$$

Considering the children of a given node i:

$$\begin{cases} d_i = 1 & \not\Rightarrow \quad d_{child(i)} = 1 \\ d_i = 0 & \Rightarrow \quad d_{child(i)} = 0 \end{cases}$$

Example: (black $d_i = 1$, white $d_i = 0$)



The basic hierarchical True Path Rule (*TPR*) ensemble

- Base classifiers estimate local probabilities $\hat{p}_i(x)$ that a given example x belongs to class c_i
- The ensemble provides an estimate of the "consensus" global probability $p_i(x)$

The "consensus" global probability $p_i(x)$ is estimated in two steps:

Bottom-up step $p_i(x)$ is computed by averaging the local probabilities of the "positive" predictions of computed at node i and $\mathrm{child}(i)$

Top-down step If $p_i(x) < 1/2$ then the subtree "is set to 0": $\forall j \in \mathrm{desc}(i), d_j(x) = 0$

The basic hierarchical True Path Rule (*TPR*) ensemble

Given the set $\phi_i(x)$ of all the children nodes of node i for which we have a positive prediction for the gene x:

$$\phi_i(x) = \{j | j \in \text{child}(i), d_j(x) = 1\}$$

The *consensus probability* $p_i(x)$ that a gene x belongs to the node/class i is:

$$p_i(x) = \frac{1}{1 + |\phi_i(x)|} \left(\hat{p}_i(x) + \sum_{j \in \phi(x)} p_j(x) \right)$$

- The $p_i(x)$ are computed in a bottom-up fashion, visiting "per-level" the tree from bottom to top, starting from the leaves, and continuing up to the root.
- At each level and for each node we have an asymmetric flow of information: bottom-up “positive” information, and top-down “negative” information.

The TPR algorithm

`Input`:
- example x whose classes need to be predicted
- tree T of the m hierarchical classes
- set of m classifiers (one for each node) each predicting $\hat{p}_i(x),\ 1 \leq i \leq m$

`begin algorithm`
01: `for each` level k of the tree T from bottom to top `do`
02: `for each` node i at level k `do`
03: `if` i is a leaf
04: $p_i(x) \leftarrow \hat{p}_i(x)$
05: `if` $(p_i(x) > t)\ d_i(x) \leftarrow 1$
06: `else` $d_i(x) \leftarrow 0$
07: `else`
08: $\phi_i(x) \leftarrow \{j | j \in \text{child}(i), d_j(x) = 1\}$
09: $p_i(x) \leftarrow \frac{1}{1+|\phi_i(x)|}\left(\hat{p}_i(x) + \sum_{j \in \phi_i(x)} p_j(x)\right)$
10: `if` $(p_i(x) > t)\ d_i(x) \leftarrow 1$
11: `else`
12: $d_i(x) \leftarrow 0$
13: `for each` $j \in$ subtree(i) `do`
14: $d_j(x) \leftarrow 0$
15: `if` $(p_j(x) > p_i(x))\ p_j(x) \leftarrow p_i(x)$
16: `end for`
17: `end for`
18: `end for`
`end algorithm`.

`Output`: for each node i
- the ensemble decisions $d_i(x) = \begin{cases} 1 & \text{if } x \text{ belongs to node } i \\ 0 & \text{otherwise} \end{cases}$
- the probabilities $p_i(x)$ that $x \in X$ belongs to the node $i \in T$

The weighted hierarchical True Path Rule (*TPR-w*) ensemble

TPR-w is a variant of the basic *TPR*: we can modulate the role of the local predictor w.r.t. the predictions of its children and descendants.

We introduce a *parent weight* w_p, $0 \leq w_p \leq 1$, such that the prediction is shared proportionally to w_p and $1 - w_p$ between respectively the local parent predictor and the set of its children:

$$p_i(x) = w_p \cdot \hat{p}_i(x) + \frac{1 - w_p}{|\phi(x)|} \sum_{j \in \phi(x)} p_j(x)$$

This learning behaviour is recursively reproduced from the leaves up to the root of the overall taxonomy.

Analysis of the propagation of the positive decisions (1)

What about the influence of the positive decisions of the descendants on the decision of their ancestors ?

We define:

- q_k: the posterior probability computed by the ensemble for a generic node at level k
- $\hat{q}_k$: the probability computed by the base learner local to a node at level k
- q^j_{k+1}: the probability of a child j of a node at level k, where the index $j \geq 1$ refers to different children of a node at level k

Analysis of the propagation of the positive decisions (2)

By defining the average probability computed by the positive children of a node at level i:

$$a_{i+1} = \frac{1}{|\phi_i|} \sum_{j \in \phi_i} \hat{q}^j_{i+1} \tag{1}$$

and the average of the probability averages of the positive grandchildren of a node at level i is:

$$a_{i+2} = \frac{1}{|\phi_i|} \sum_{j \in \phi_i} \frac{1}{|\phi^j_{i+1}|} \sum_{k \in \phi^j_{i+1}} \hat{q}^k_{i+2} \tag{2}$$

By iterating this procedure at the next level we obtain:

$$a_{i+3} = \frac{1}{|\phi_i|} \sum_{j \in \phi_i} \frac{1}{|\phi^j_{i+1}|} \sum_{k \in \phi^j_{i+1}} \frac{1}{|\phi^k_{i+2}|} \sum_{l \in \phi^k_{i+2}} \hat{q}^l_{i+3} \tag{3}$$

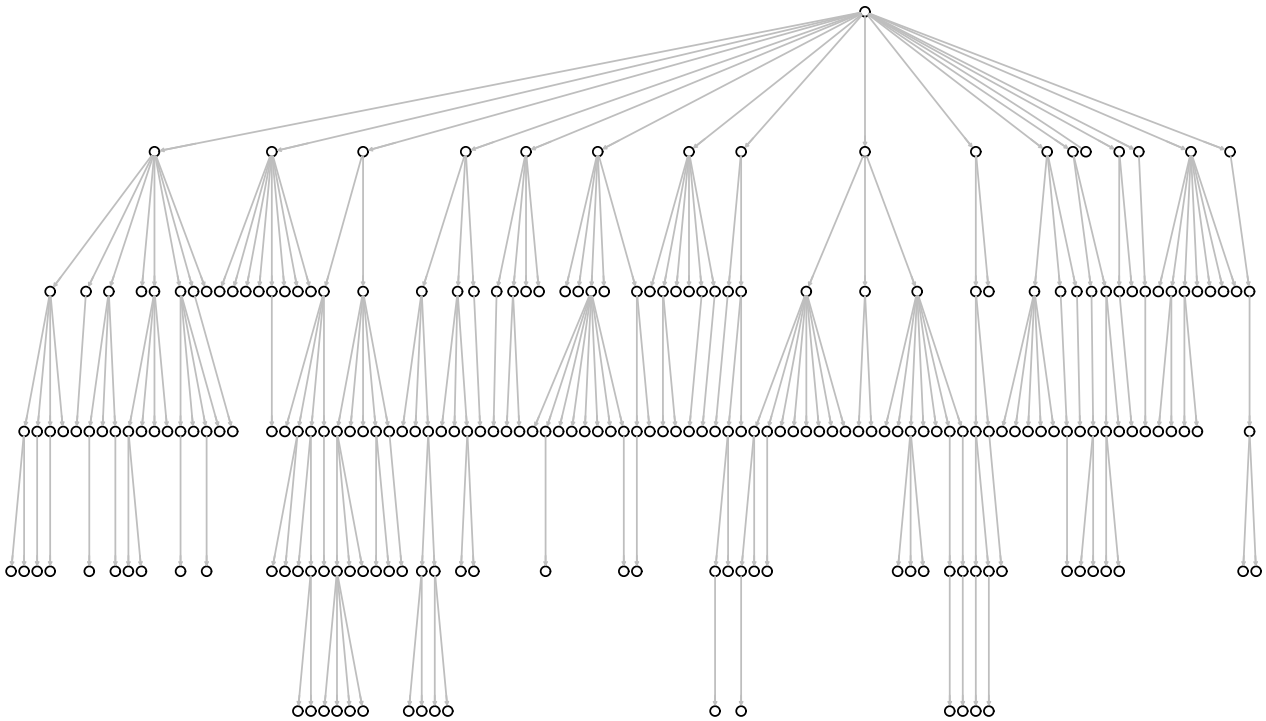
Theorem: Influence of positive descendant nodes

In a *TPR-w* ensemble, for a generic node at level i, with a given parameter $w, 0 \leq w \leq 1$, balancing the weight between parent and children predictors, and having a variable number larger or equal than 1 of positive descendants for each of the m lower levels below, the following equality holds for each $m \geq 1$:

$$q_i = w\hat{q}_i + \sum_{j=1}^{m-1} w(1-w)^j a_{i+j} + (1-w)^m a_{i+m}$$

The contribution of descendant nodes decays exponentially w.r.t. their depth

An application of TPR ensembles to a genome and ontolgy-wide GFP problem



- Genome-wide gene function prediction in *S. cerevisiae*
- About 200 FunCat classes (5 hierarchical levels)
- About 6000 genes to be classified (1000 unknown)

Data sets and experimental set-up

Table: Yeast "omics" data sets

Data set	Description	n. examples	n. feat.	n.classes
Pfam-1	protein domain binary data from *Pfam*	3529	4950	211
Pfam-2	protein domain log E data from *Pfam*	3529	5724	211
Phylo	phylogenetic data	2445	24	187
Expr	gene expression data	4532	250	230
PPI-BG	PPI data from *BioGRID*	4531	5367	232
PPI-VM	PPI data from *STRING*	2338	2559	177
SP-sim	Sequence pairwise similarity data	3527	6349	211

- Experimental comparison between *Flat* and Hierarchical Top-Down (*HTD*, see next slide) ensembles versus the proposed True Path Rule in both basic (*TPR*, and weighted (*TPR-w*) form.
- Probabilistic linear SVMs as base learners [Lin et al., 2007]
- 5-fold cross validation
- F-per-class and F-hierarchical measures [Verspoor et al., 2006]

The Hierarchical Top-Down (HTD) algorithm

HTD classifies examples in a single top-down pass, starting from root and down to the leaves.

- x: a singe example
- $d_i(x)$: the classifier decision at node i
- $root(T)$: set of nodes at the first level of the tree T

$$y_i = \begin{cases} d_i(x) & \text{if } i \in root(T) \\ d_i(x) & \text{if } i \notin root(T) \text{ } AND \text{ } y_{par(i)} = 1 \\ 0 & \text{if } i \notin root(T) \text{ } AND \text{ } y_{par(i)} = 0 \end{cases}$$

F-score results

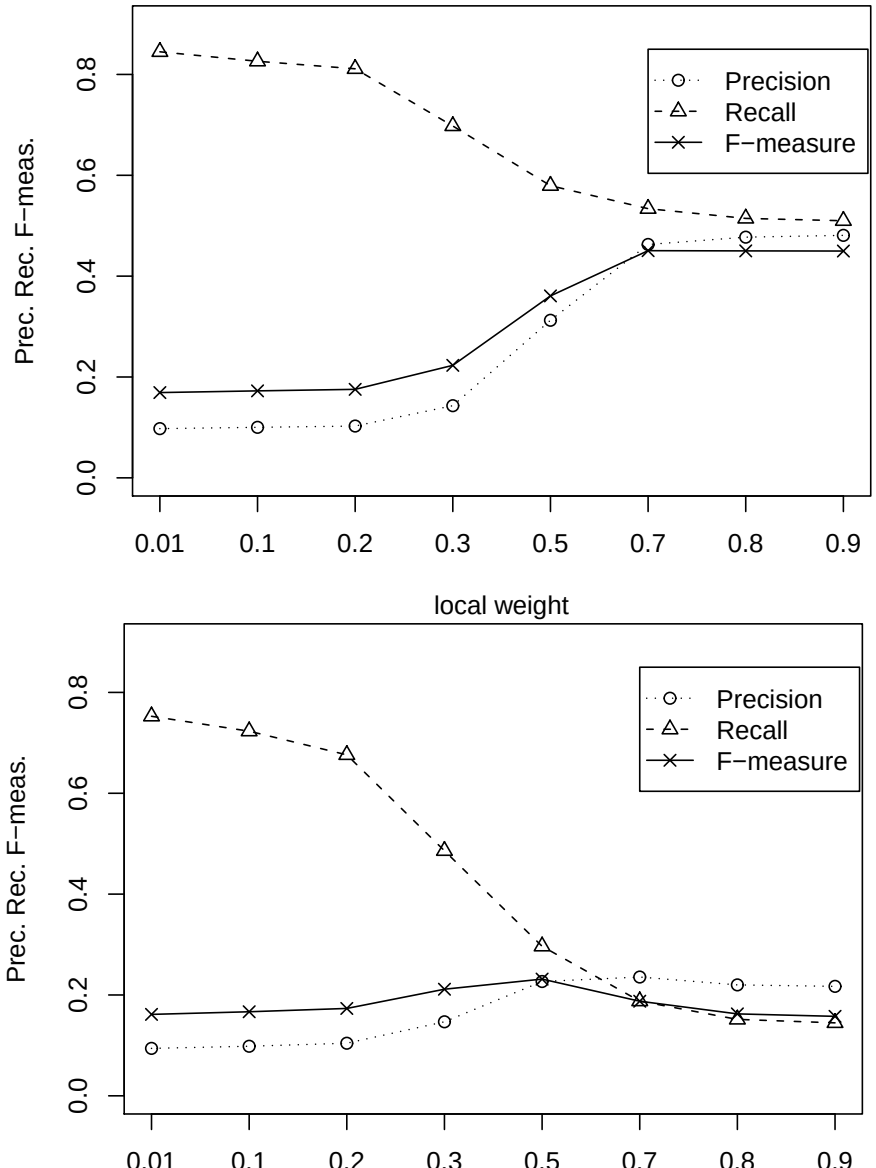
Average per-class F-score results

Flat	HTD	TPR	TPR-w
0.1489	0.2222	0.1824	0.2414

Hierarchical F-score results

Data set	HTD	TPR	TPR-w
Pfam-1	0.4123	0.3080	0.4131
Pfam-2	0.3406	0.2684	**0.3700**
Phylo	0.0497	0.2010	**0.2174**
Expr	0.1166	0.1696	**0.1784**
PPI-BG	0.3226	0.2670	**0.3485**
PPI-VM	0.3977	0.2796	0.4000
SP-sim	0.4251	0.2398	**0.4472**
Average	0.2949	0.2468	0.3392

Tuning hierarchical precision and recall in TPR-w ensembles



- A single global parameter can regulate precision/recall characteristics (useful for different types of gene function prediction tasks)
- Opposite trends of precision and recall w.r.t the parent weight parameter w_p
- Usually better F-scores with $w_p > 0.5$

TPR summary

- TPR is a heuristic ensemble method based on the “true path rule”
- TPR-w achieves significantly better hierarchical F-scores than the basic TPR and Top-down ensembles, and largely bettr results than flat approaches
- Per-level analysis shows that this is the result of a better compromise between precision and recall
- With a single global parameter we may tune the precision/recall characteristics of the overall TPR-w ensemble
- Analysis of the bottom-up propagation of the information across the tree shows that the influence of positive decisions of the descendant nodes decay exponentially with their depth $\Longrightarrow$ room to design and experiment new variants

HBAYES-CS: a hierarchical cost-sensitive algorithm for genome-wide gene function prediction

(Cesa-Bianchi and Valentini, 2010)

- A variant of the basic *HBAYES* algorithm [Cesa-Bianchi et al., 2006]
- *HBAYES-CS* takes into account the unbalance between positive and neagative examples that characterize gene functional classes
- *HBAYES-CS* has been applied to genome and ontology-wide GFP in *S. cerevisiae*

The HBAYES method

- Train a set of independent classifier (one for each functional class)
- Modification of the labels through a bottom-up recursive procedure
- Underlying stochastic model for the multilabels
- Based on an approximation of the bayesian-optimal classifier for the H-loss

The H-loss

The main intuition behind the H-loss:
if a parent class has been predicted wrongly, then errors in its descendants should not be taken into account.

Given the predicted multilabel $\widehat{\boldsymbol{y}} = (\widehat{y}_1, \ldots, \widehat{y}_N)$ and the true multilabel $\boldsymbol{v} = (v_1, \ldots, v_N)$ and the cost coefficients $c_1, \ldots, c_N$:

$$\ell_H(\widehat{\boldsymbol{y}}, \boldsymbol{v}) = \sum_{i=1}^{N} c_i \{\widehat{y}_i \neq v_i \wedge \widehat{y}_j = v_j, j \in \mathrm{anc}(i)\}$$

The stochastic model for multilabels

Given:

- $\boldsymbol{V} = (V_1, \ldots, V_N) \in \{0,1\}^N$: the vector random variable modeling the multilabel of a gene $\boldsymbol{x}$
- $\mathrm{par}(i)$: the unique parent of node i in T
- the probability $p_i(\boldsymbol{x})$ for node i on input $\boldsymbol{x}$ is: $p_i(\boldsymbol{x}) = \mathbb{P}\big(V_i = 1 \mid V_{\mathrm{par}(i)} = 1,\, \boldsymbol{x}\big)$
- *True path rule consistent predictions*: the distribution of the random boolean vector $\boldsymbol{V}$ is assumed to be

$$\mathbb{P}\big(\boldsymbol{V} = \boldsymbol{v}\big) = \prod_{i=1}^{N} \mathbb{P}\big(V_i = v_i \mid V_{\mathrm{par}(i)} = 1,\, \boldsymbol{x}\big) \qquad \text{for all } \boldsymbol{v} \in \{0,1\}^N$$

with:
$\mathbb{P}\left(V_i = 1 \mid V_{\mathrm{par}(i)} = 0,\, \boldsymbol{x}\right) = 0$

The Bayes-optimal classifier

By definition the *Bayes-optimal classifier* is:

$$\boldsymbol{y}^* = \underset{\boldsymbol{y} \in \{0,1\}^n}{\operatorname{argmin}} \, \mathbb{E}\left[\ell_H(\boldsymbol{y}, \boldsymbol{V}) \mid \boldsymbol{x}\right]$$

where the expectation is w.r.t. the distribution of $\boldsymbol{V}$.

Theorem [Cesa-Bianchi et al., 2005]: The multilabel generated by *HBAYES* is the *Bayes-optimal* for the H-loss.

The HBAYES algorithm

- The classification of an example $\boldsymbol{x}$ is performed node-by-node with a *bottom-up strategy* using a per-level (or DFS) visit of the tree.
- For each node i the corresponding label $\widehat{y}_i$ is computed according to:
  - the underlying stochastic model for multilabels
  - the Bayesian-optimal strategy to minimize the H-loss

that is:

$$\widehat{y}_i = 1 \iff p_i(2 - \sum_{k \in \text{child}(i)} H_k(\widehat{\boldsymbol{y}})/c_i) \geq 1$$

where for each node k H_k is recursively defined as follows:

$$H_k(\widehat{\boldsymbol{y}}) = c_k(p_k(1 - \widehat{y}_k) + (1 - p_k)\widehat{y}_k) + \sum_{j \in \text{child}(k)} H_j(\widehat{y})$$

Note that if i is a leaf node the above rule is equivalent to $\widehat{y}_i = \{p_i \geq 1/2\}$.

HBAYES-CS: the cost-sensitive variant

The problem: we have very unbalanced classification tasks

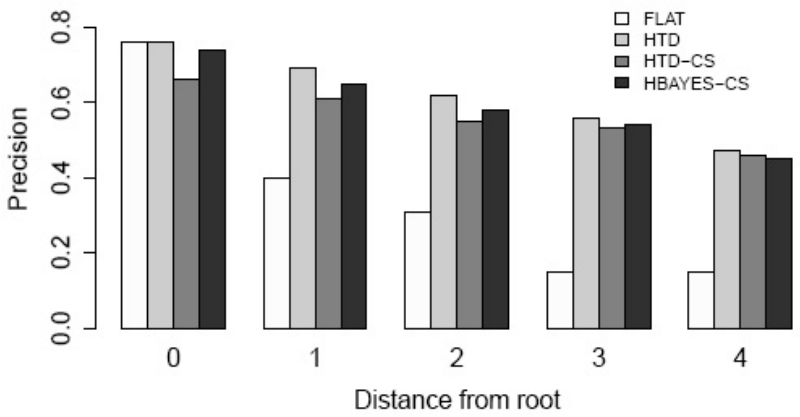
A possible solution: introducing a trade-off between the costs of false positive (FP) c_i^+ and false negative (FN) c_i^- mistakes: $c_i^- = \alpha c_i^+$ while keeping $c_i^+ + c_i^- = 2c_i$ with $\alpha \geq 0$.

In this setting the decision rule for *HBAYES-CS* becomes:

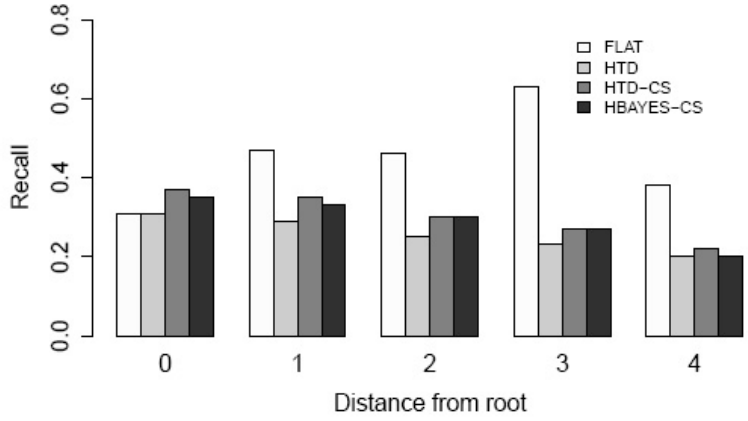
$$\widehat{y}_i = 1 \iff p_i \left(2 - \sum_{j \in \text{child(i)}} H_j / c_i \right) \geq \frac{2}{1+\alpha} \, .$$

- $\alpha > 1$: we put more cost on negative predictions
- $\alpha < 1$: we put more cost on positive predictions
- $\alpha = 1$: we come back to the non cost-sensitive version

Experiments with the yeast: precision and recall



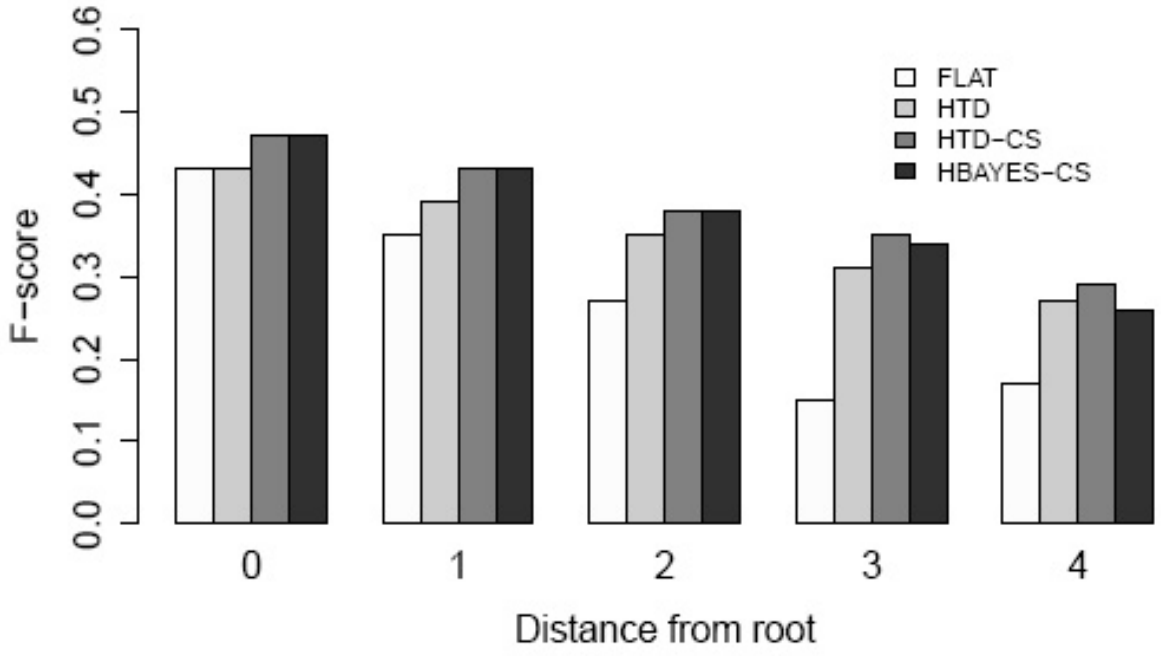
(Precision)



(Recall)

Precision and recall across levels (Pfam data).

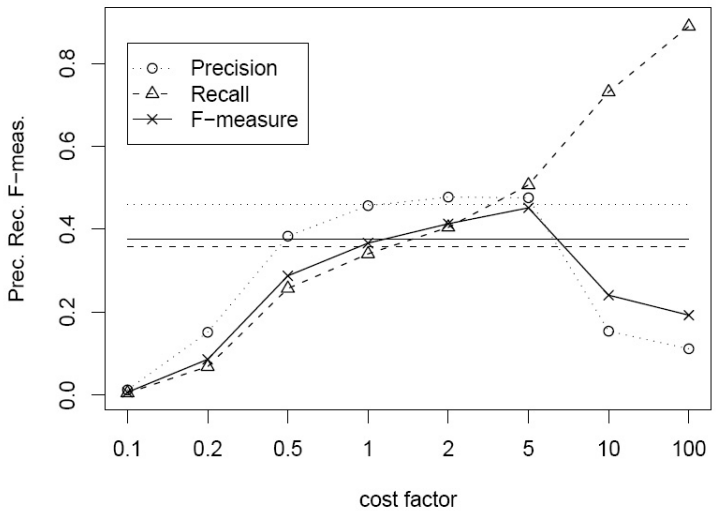
Experiments with the yeast: F-score



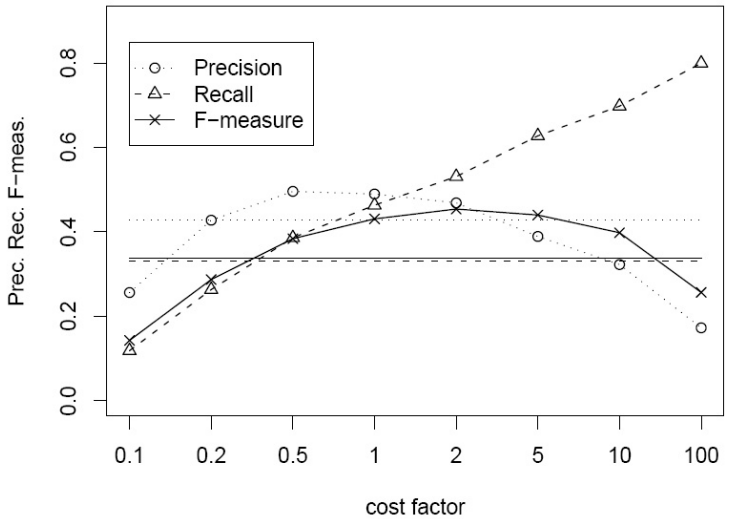
Win-tie-loss between methods:

win-tie-loss		
Methods	HTD-CS	HTD
HBAYES-CS	2-4-1	6-1-0
HTD-CS	-	7-0-0

Tuning precision and recall through the α parameter



(Pfam-1)



(SP-sim)

Hierarchical precision, recall and F-measure as a function of the cost modulator factor α (Pfam and sequence similarity data).

Combination of hierarchical ensembles with data fusion methods

(Cesa-Bianchi, Re and Valentini, 2010)

A two-steps strategy:

1. For each term of the taxonomy, train a classifier using multiple sources of data
2. Combine the predictions at each node to obtain the multi-label predictions according to the *HBAYES-CS* method.

Two-levels of improvements:

- Improvement of flat predictions through the bottom-up Bayesian correction
- Improvement of the single-source predictions by exploiting multiple sources of data

Data Fusion methods

Weighted voting: $\hat{P}(V_i = 1 \mid g) = \frac{1}{\sum_{s=1}^{L} F_s} \sum_{t=1}^{L} F_t \, \widehat{p}_{t,i}(g)$

Kernel Fusion: $K_{\text{ave}}(g, g') = \frac{1}{L} \sum_{t=1}^{L} K_t(\boldsymbol{x}_t, \boldsymbol{x}'_t)$.

where:

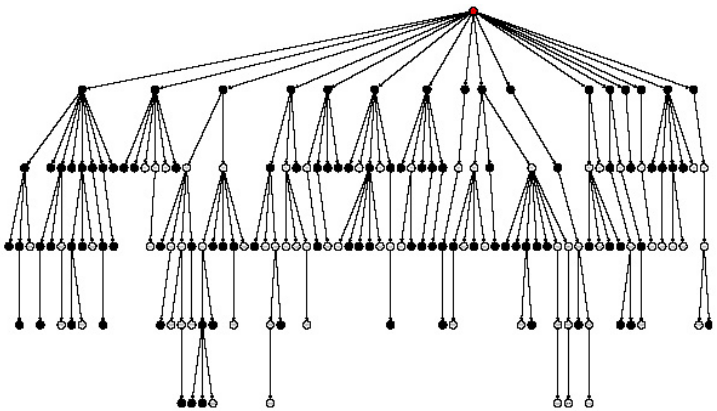
- $V_i \in \{0, 1\}$: random variable that models the labeling of a gene g for the class $\omega_i \in \Omega$
- L different sources of biomolecular data D_t, for $t = 1, \ldots, L$
- $\widehat{p}_{t,i}(g)$: classifier's estimate of the probability that g belongs to ω_i using data D_t
- F_t is the F-measure assessed on the training data for the t-th base learner
- g and g': a pair of genes, and $\boldsymbol{x}_t, \boldsymbol{x}'_t \in D_t$ their corresponding pairs of feature vectors.

Results: Impact of data fusion on flat and hierarchical methods (average F-scores)

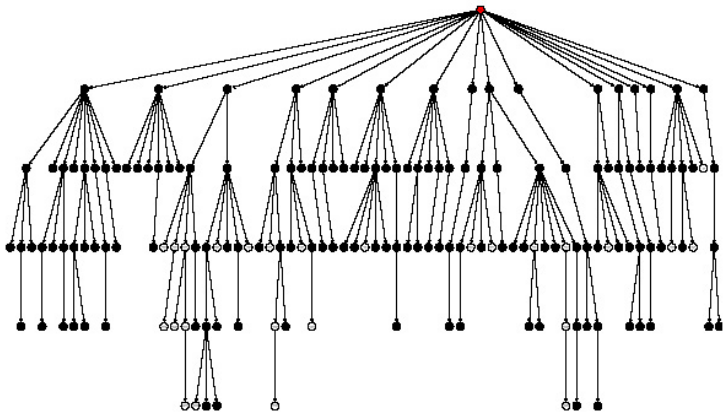
METHODS	FLAT	HTD	HTD-CS	HB	HB-CS
SINGLE-SOURCE					
BIOGRID	0.2643	0.3759	0.4160	0.3385	0.4183
STRING	0.2203	0.2677	0.3135	0.2138	0.3007
PFAM BINARY	0.1756	0.2003	0.2482	0.1468	0.2395
PFAM LOGE	0.2044	0.1567	0.2541	0.0997	0.2500
EXPR.	0.1884	0.2506	0.2889	0.2006	0.2781
SEQ. SIM.	0.1870	0.2532	0.2899	0.2017	0.2825
MULTI-SOURCE (DATA FUSION)					
KERNEL FUSION	**0.3220**	**0.5401**	**0.5492**	**0.5181**	**0.5505**
WEIGH. VOTING	0.2754	0.2792	0.3974	0.1491	0.3532

- 6 data sources
- 2 data fusion techniques: Kernel Fusion and weighted voting
- Flat, HTD, HBAYES and their cost-sensitive variants

Comparison of F-scores with and without data integration



Flat



HBAYES-CS

- Black nodes: better results with data fusion
- White nodes: better results with the best single-source data
- p-value= $2.2 \cdot 10^{-16}$ (Wilcoxon signed-rank sums test)

Synergy between hierarchical, data fusion and cost-sensitive techniques

METHODS	F-SCORE	PREC.	REC.
BIOGRID:			
FLAT	0.1893	0.1253	0.5801
HTD	0.4311	0.5901	0.3827
HTD-CS	0.4732	0.5645	0.4650
HBAYES	0.3776	0.5404	0.3236
HBAYES-CS	0.4738	0.5654	0.4639
KF:			
FLAT	0.2052	0.1293	**0.7026**
HTD	0.5800	**0.7051**	0.5560
HTD-CS	**0.6091**	0.6745	0.6156
HBAYES	0.5512	**0.6915**	0.5086
HBAYES-CS	**0.6073**	0.6759	0.6126

- Best F-score: joint hierarchical cost-sensitive and data fusion techniques
- Best precision: HTD and HBAYES but also HBAYES-CS and HTD-CS perform well
- Best recall: FLAT, but also HBAYES-CS and HTD-CS good results
- Better compromise between precision and recall: HBAYES-CS and HTD-CS.

Conclusions

- Hierarchical strategies show better results than “flat” approaches
- TPR-W and HBAYES-CS achieve significantly better hierarchical F-scores than the basic TPR and HTD ensembles
- This is the result of a better compromise between precision and recall
- With a single global parameter we may tune the precision/recall characteristics of the overall TPR-W and HBAYES-CS ensembles
- Data fusion significantly improve predictions
- We need a sinergy between hierarchical, data fusion and cost-sensitive approaches to achive the best results.

Some open problems ...

- Biomolecular data integration can improve gene function prediction performances: which other methods could be considered?
- How to take into account different evidences of association gene-functional class?
- Can we improve the TPR-W algorithm by choosing non exponential decay rules for the weights?
- Can we extend TPR-W and HBAYES-CS to DAG-structured taxonomies (e.g. GO)?
- Can we introduce active learning techniques in this context?
- Experimental work: comparison with other promising hierarchical ensemble approaches and state-of-the-art methods in the context of genome-wide gene function prediction
- Can these methods to be applied to genome and ontology-wide of multi-cellular eukaryotes? (e.g. *A.thaliana*, mouse or human)

Large room for further research ...

References

- G. Valentini, *True Path Rule hierarchical ensembles for genome-wide gene function prediction*, IEEE ACM Transactions on Computational Biology and Bioinformatics (in press).
- N. Cesa-Bianchi, G. Valentini, *Hierarchical cost-sensitive algorithms for genome-wide gene function prediction*, Journal of Machine Learning Research, vol.8: Machine Learning in Systems Biology, pp.14-29, 2010
- M. Re, G. Valentini, *Simple ensemble methods are competitive with state-of-the-art data integration methods for gene function prediction*, Journal of Machine Learning Research, vol.8: Machine Learning in Systems Biology, pp. 98-111, 2010.
- M. Re, G. Valentini, *An experimental comparison of Hierarchical Bayes and True Path Rule ensembles for protein function prediction*, In: MCS 2010, Lecture Notes in Computer Science, vol. 5997, pp. 294-303, Springer, 2010.
- N. Cesa-Bianchi, M. Re, G. Valentini, *Functional Inference in FunCat through the Combination of Hierarchical Ensembles with Data Fusion Methods*, ICML Workshop on learning from Multi-Label Data, 2010 (in press).

All these papers (and others) are available from:
`http://homes.dsi.unimi.it/~valenti/pub.html`