Appendix

Algorithms for Regular Tree Grammar based search and their application to mining human-viral infection patterns

Ilan Smoly[1*], Amir Carmel[1*], Yonat Shemer-Avni[2], Esti Yeger-Lotem[3,**] and Michal Ziv-Ukelson[1,**]

[1]Department of Computer Science, Ben-Gurion University of the Negev, Israel.
[2]Department of Virology, Ben-Gurion University of the Negev, Israel.
[3]Department of Clinical Biochemistry and Pharmacology, Ben-Gurion University of the Negev, Israel.

Supplementary Materials (pages 2-8):

1) Running-time complexity per color-coding iteration (page 2)
2) Bounding the error probability by iterative color-coding (page 4)
3) Scoring and Monotonicity (page 5)
4) Search Space Pruning (page 6)
5) Grammars used in the results (page 6)
6) Extended results (page 7)

Figures (pages 9-11):

Figure S1. Curryfication of a Regular Tree Grammar and the corresponding curryfication of the tree to be parsed (page 9)

Figure S2. The basic workflow of our framework (page 10)

Figure S3. The canonical code (page 10)

Figure S4. Extended results (page 11)

Code:

The code of RTGnet, implemented in Python, is available at the following URL: http://www.cs.bgu.ac.il/~smolyi/RTGnet/

1. Running-time Complexity per Color-coding Iteration

The time complexity of the work done per each color-coding iteration will be measured by the following parameters:

1. g – the number of production rules in *Curry(A).*
2. $N -$ the number of non-terminal symbols in $Curry(A)$.
3. $|V|, |E|$ – the number of nodes and edges in the input directed graph.
4. m – the maximum number of nodes in the output trees.
5. $k -$ the number of optimal solutions to report.
6. $\alpha -$ the maximal number of production rules with the same non-terminal on the left side.

For a given coloring of the input graph $G = (V, E)$, and a Curry-Encoded RTG $A = \{N, \Sigma, P, S\}$, the algorithm runs as follows: First, it constructs the hypergraph $H = (V_H, E_H)$ in which every hyper-node, denoted by a tuple $x = (X, v, q)$, represents a class of rooted subtrees in G, with the following properties: $X \in P$ is a non-terminal symbol, $v \in V$ is the root vertex and $q \subseteq C$ is the set of colors of the vertices of any subtree represented by x, for which $C(v) \in q$. Hence, the number of hypernodes is at most $N \cdot |V| \cdot 2^m$.

In what follows, we analyze the time complexity of each of the two stages of our algorithm.

1.1. Stage 1: construction of the hypergraph

Theorem 1: *The time complexity for the construction of the hypergraph H is* $O(m \cdot g \cdot |E| \cdot 3^m)$.

Proof: We say that two hypernodes x,y are partners if there exists a hyperedge $e \in E_H$ such that $tail(e) = < x, y >$. From the definition of edges in the hypergraph we get that hypernodes $x = (X, v, q_x)$, $y = (Y, u, q_y)$ are partners if and only if: (1) $(u, v) \in E$ (2) there exists a derivation rule and a non-terminal $W \in P$ such that $W \to X@Y$ and (3) $q_x \cap q_y = \emptyset$. For each hypernode, we want to efficiently obtain its candidate partner hypernodes. For this, we implement the following data-structures:

- For a non-terminal $X \in N$, let $L_X = \{Y \in N : W \to X@Y \in P\}$. Construction of L_X for each $X \in N$ is done, in pre-processing, in linear time in the size of the grammar. For each non-terminal $X \in N$, L_X is retrieved in constant time.
- $F(v, W)$ - A table of size $g \cdot |V|$, such that for each $v \in V$ and for each non-terminal $W \in N$, $F(v, W)$ will store all hypernodes (W, v, q), where $q \subseteq C$ can be any subset of colors. The hypernodes in $F(v, W)$ are stored in a binary tree of height $m + 1$, according to the binary representation of q, in the following manner. For any node in level i of the binary tree in $F(v, W)$, the left edge corresponds to color-sets that contain the color $i + 1$ and the right edge corresponds to color-sets that don't contain the color $i + 1$. The hypernodes are stored in the leaves of $F(v, W)$, such that the path from the root to a leaf u spells out the colors of the subtrees represented by the hypernodes stored in u.

The construction of the hypergraph is carried out as follows. Given a hypernode $x = (X, v, q_x)$, for each edge $(u, v) \in E$, and for each Y in L_X, we use the table $F(u, Y)$ to retrieve the relevant partners of x. The time for reporting each relevant partner is $O(m)$, due to the binary tree traversal. We denote by γ_x the number of potential legal partners of x. Thus, we can retrieve all possible relevant partners of x, in $O(m \cdot \gamma_x)$. Overall, the running time of the hypergraph construction is bounded by $O(|V_H| + \sum_{x \in V_H} m \cdot \gamma_x)$ (i.e. construction of the nodes plus construction of the edges in the hypergraph). We define $d_v = |\{v' : (v, v') \in E\}|$ and $l_X = |L_X|$, then $\gamma_x \le l_X \cdot d_v \cdot 2^{m-|q_x|}$. Then we get:

$$\sum_{x \in V_H} m \cdot \gamma_x \le m \cdot \sum_{(X,v,q_x) \in V_H} l_X \cdot d_v \cdot 2^{m-|q_x|}$$

$$\le m \sum_{v \in V} d_v \sum_{X \in N} l_X \sum_{q \subseteq C} 2^{m-|q|} \overset{(1)}{\le} m \sum_{v \in V} d_v \sum_{X \in N} l_X \sum_{i=0}^{m} \binom{m}{i} 2^{m-i} \overset{(2)}{=} m|E|g \sum_{i=0}^{m} \binom{m}{i} 2^{m-i} \overset{(3)}{=} mg|E|3^m$$

1. We sum the values of q, where q varies from 0 to m, and there are at most $\binom{m}{i}$ coloring subset of size i.
2. $\sum_{v\in V} d_v = |E|$, $\sum_{X\in N} l_X = g$,
3. From the binomial theorem: $(x+a)^n = \sum_{k=0}^{n} \binom{n}{k} x^k a^{n-k}$, now take $x = 2$ and $a = 1$.

1.2. Stage 2: computing k-best derivations of the hypergraph

Theorem 2: *The time complexity for the second stage of our algorithm is* $O(|E_H| + |V_H| \cdot km(m\log k + \log\alpha)) = O(g|E|3^m + g|V|2^m km(m\log k + \log\alpha))$.

Proof: In Stage 2, the algorithm computes the k-best derivations. For this purpose, we describe how to modify Huang and Chiang's[2] Algorithm 2 to avoid isomorphic subtrees (see definition below). We do not address their lazy Algorithm 3 since some of its assumptions do not hold in our case, as discussed in the paper. Remember that in our application we want to return k-best non-isomorphic rooted subtrees. As shown in Figure S3, there could be several top-scoring derivations of the very same unordered subtree T, differing only by their node ordering. In order to overcome this obstacle, we produce the canonical representation of the tree, prior to inserting it to a k-best list. By this, we guarantee that in each k-best list we have exactly one subtree of a given isomorphism type. To compute the new bound of the modified algorithm, we define:

- $r = \max_{v\in V} r_v$, where r_v denotes the maximal number of children of any node v in any tree that is considered as a candidate solution to *Problem 2*.
- $\alpha_X = \left|\{X \to Y@Z \in P \mid Y, Z \in N\}\right|$ and $\alpha = \max_{X\in N} \alpha_X$.

Rooted Unordered Tree Isomorphism: *In Rooted Unordered Isomorphism, trees τ_1 and τ_2 are isomorphic if τ_1 can be obtained from τ_2 by subtree reordering operations. Formally, we define $\tau_1 \equiv \tau_2$ by induction on the number of nodes in τ_1 and τ_2 by defining that $\tau_1 \equiv \tau_2$ holds if and only if:*

1. *$|\tau_1| = |\tau_2| = 1$ and $\tau_1 = \tau_2$ or*
2. *τ_1 and τ_2 both have the same number, m, of immediate subtrees, and there is some ordering $\tau_1^1,\ldots,\tau_1^m$ of the immediate subtrees of τ_1 and some ordering $\tau_2^1,\ldots,\tau_2^m$ of the immediate subtrees of τ_2 such that $\tau_1^i \equiv \tau_2^i$ for all $1 \le i \le m$.*

By definition, in Rooted Unordered Tree Isomorphism, any two isomorphic trees have the same canonical encoding (Figure S3).

Lemma 2: *For hypernode $x = (X, v, q_x)$, the number of distinct incoming derivations that may produce the same tree (up to isomorphism) is bounded by $\alpha_X \cdot r$.*

<u>*Claim 1:*</u> any two derivations induced by the same hyperedge, correspond to non-isomorphic trees.

<u>*Proof of Claim 1:*</u> Let $e = (h \leftarrow < x, y >)$ be some hyperedge and denote $x = (X, v, q_x), y = (Y, u, q_y)$. Assume, in contradiction, that there are two distinct derivation d_1 and d_2, induced in e, that produce the same isomorphism type $T = (V', E')$ and denote $d_1 = (e, < i_1, j_1 >)$, $d_2 = (e, < i_2, j_2 >)$. Without loss of generality, let $j_1 \neq j_2$. Denote τ_1 and τ_2 the subtrees corresponding to $D^{j_1}\left(t_2(e)\right)$ and $D^{j_2}\left(t_2(e)\right)$, respectively. Denote by V_1, V_2 the set of nodes of τ_1 and τ_2, respectively. Since τ_1 and τ_2 come from the same hypernode, they consist of the same colors q_y. Also note that $V_1, V_2 \subseteq V'$, thus $V_1 = V_2$ (otherwise, there are two different vertices in T with the same color). We get that both τ_1 and τ_2 correspond to a subtree of T over the vertices $V_1 = V_2$ and therefore correspond to isomorphic trees. Since the k-best list does not contain isomorphic trees, we get in contradiction $j_1 = j_2$. Q.E.D

<u>*Proof of Lemma 2:*</u> Following *Claim 1*, given a hypernode $x = (X, v, q_x)$, all we need to do is to count the number of distinct hyperedges in $BS(x)$ that can produce isomorphic trees. Consider an ordered rooted tree T in the class of subtrees represented by x, thus T is rooted in v. Assume, in contradiction, $\alpha_X \cdot r + 1$ distinct incoming hyperedges that produce trees isomorphic to T. Since X could be generated by at most α_X different rules, from the pigeon-hole principle there is a rule, say $X \to W@Y$, that corresponds to at least $r + 1$ distinct hyperedges. Note that every hyperedge corresponds to some

edge in G. Also, there are at most r children of v in T, (and thus, at most r edges adjacent to v). Therefore, once again, from the pigeon-hole principle, there exist two distinct hyperedges e_1, e_2, denoted by, $e_1 = \left((x_1 = (W, v, q_{x_1}), y_1 = (Y, u, q_{y_1})\right)$ and $e_2 = \left(x_2 = (W, v, q_{x_2}), y_2 = (Y, u, q_{y_2})\right)$, that correspond to the same edge $(u, v) \in E$. Since $q_x = q_{x_1} \cup q_{y_1} = q_{x_2} \cup q_{y_2}$ and $q_{x_1} \cap q_{y_1} = q_{x_2} \cap q_{y_2} = \emptyset$, we get that $q_{x_1} \neq q_{x_2}, q_{y_1} \neq q_{y_2}$ (otherwise $q_{x_1} = q_{x_2} \leftrightarrow q_{y_1} = q_{y_2}$ which means $x_1 = x_2, y_1 = y_2$). Assume W.L.O.G that $q_{x_1} \setminus q_{x_2} \neq \emptyset$, therefore, there exists a vertex w which is a descendant of v in some rooted subtree represented by (W, v, q_{x_1}) and also a descendant of u in some rooted subtree represented by (Y, u, q_{y_2}). Since (u, v) is an edge in the tree T, we get two different paths from v to w in T (one contains u and the other doesn't), this is a contradiction to the fact that T is a tree and therefore there exists only one unique path to any descendant of the root. Q.E.D.

Now, following *Lemma* 2, we can analyze the running-time of *findKBest* for a given hypernode $x = (X, v, q_x)$, as follows. In algorithm *findKBest*, in every iteration, a new candidate is inserted to the k-best list. Therefore, the number of iterations is bounded by *k*. In our variation, for each tree τ in the k-best list of x, there might be $\alpha_X \cdot r$ iterations in which we compute τ. Therefore we maintain a priority queue of the top-$kr\alpha_X$ candidate derivations (instead of top-k as in Huang and Chiang's algorithm). For each such candidate in the k-best selecting method, we construct its canonical representation in *O(m)* time. We use a binary search tree sorted lexicographically to store the canonical codes of all subtrees in the k-best list, in order to verify that any candidate subtree is not already in the top-k list. Hence operations cost $(mlogk)$ time (i.e. *O(logk)* for traversing the binary tree of size *k*, multiplied by an *O(m)* factor for comparator operations). Since for every $v \in V, r_v \leq m$ we get $r \leq m$, and therefore the running time per hypernode *x* is computed as follows:

$$
\begin{aligned}
O\big(|BS(x)| + kr\alpha_X(\log(kr\alpha_X) + m\log k)\big) &= O\big(|BS(x)| + kr\alpha_X(mlogk + logr + \log\alpha_X)\big) \\
&\leq O\big(\,|BS(x)| + km\alpha_X(\,m\log k\ + \log\alpha_X)\big) \\
&\leq O(|BS(x)| + km\alpha_X(mlogk + log\alpha))
\end{aligned}
$$

The bound on the number of hyperedges in E_H, based on *Theorem 2*, is $O(g \cdot |E| \cdot 3^m)$. Thus, the total running-time of Step 2, for computing *findKBest* for all hypernodes, is computed as follows.

$$
\sum_{x\in V_H} O\big(\,|BS(x)| + km\alpha_X(m\log k + \log\alpha)\big) = O\big(\textstyle\sum_{x\in V_H} BS(x) + km\alpha_X(m\log k + \log\alpha)\big) = O\big(\textstyle\sum_{x\in V_H} BS(x) + \sum_{x\in V_H} km\alpha_X(m\log k + \log\alpha)\big) = O\big(|E_H| + \sum_{v\in V}\sum_{q\subseteq C}\sum_{X\in N} km\alpha_X(m\log k + \log\alpha)\big) = O(|E_H| + \sum_{v\in V}\sum_{q\subseteq C} km(m\log k + \log\alpha)\sum_{X\in N}\alpha_X) \overset{(1)}{=} O(|E_H| + \sum_{v\in V}\sum_{q\subseteq C} km(m\log k + \log\alpha)g) = O(|E_H| + g\cdot|V|\cdot 2^m km(m\log k + \log\alpha)) = O\big(g\cdot|E|\cdot 3^m + g\cdot|V|\cdot 2^m km(m\log k + \log\alpha))\big)
$$

1. $\sum_{X\in N}\alpha_X = g$ since every production rule is counted exactly one time.

Q.E.D

2. Bounding the Error Probability by Iterative Color-coding

Theorem : *For any* $\epsilon > 0$, *after* $e^m ln\left(\frac{k}{\epsilon}\right)$ *iterations, the output list contains all the k-best subtrees with error probability* $\leq \epsilon$.

Proof: Let $V_1, \dots, V_k$ be the sets of nodes in the k-best subtrees. For any i, $|V_i| \leq m$, thus V_i is colorful in iteration j, with probability $\geq \frac{m!}{m^m} \geq e^{-m}$ (since $|V_i|!$ is the number of legal colorings, $|V_i|^{|V_i|}$ is the total number of colorings, and $|V_i| \leq m$). Denote by A_i^j the event that V_i is not colorful in iteration j, thus: $P\left[A_i^j\right] \leq 1 - e^{-m}$.

Note that if V_i is colorful in at least one iteration, then V_i is part of the output. Therefore, the k subtrees are not the k-best if and only if there exists some V_i such that V_i is not colorful in any iteration. Denote by A_i the event that V_i is not colorful in any iteration (i.e. $A_i = \bigcap_{j=1}^{t} A_i^j$), then

$$P[Failure] = P[A_1 \cup A_2 \cup \ldots \cup A_k] \leq \sum_{i=1}^{k} P[A_i]$$

$$P[A_i] = P\left[A_i^1 \cap A_i^2 \cap \ldots \cap A_i^t\right] \overset{(1)}{=} \prod_{j=1}^{t} P\left[A_i^j\right] \leq \prod_{j=1}^{t} (1 - e^{-m}) = (1 - e^{-m})^t \overset{(2)}{\leq} \left(e^{-(e^{-m})}\right)^t$$

$$\Rightarrow P[Failure] \leq \sum_{i=1}^{k} P[A_i] = \sum_{i=1}^{k} \left(e^{-(e^{-m})}\right)^t = k \cdot e^{-(e^{-m}) \cdot t}$$

$$k \cdot e^{-e^{-m} \cdot e^{m} ln\left(\frac{k}{\epsilon}\right)} = \epsilon \Rightarrow P[Failure] \leq \epsilon$$

Notes:
(1) The colorings of distinct iterations are independent.
(2) $(1 - x) \leq e^{-x}$.
Q.E.D

Thus, for a failure probability ϵ, the total running time of our algorithm is:

$$O(e^m ln\left(\frac{k}{\epsilon}\right)\left(mg|E|3^m + g|V|2^m mk\ (m \log k + \log \alpha)\right))$$

3. Scoring Functions and Monotonicity

Monotonicity of derivations of hypernodes. A function $f: \Re^m \to \Re$ is monotonic with regards to $\preccurlyeq$, if for all $i \in \{1 \ldots m\}$, $(a_i \preccurlyeq a_i') \Rightarrow f(a_1, \ldots, a_m) \leq f(a'_1, \ldots, a'_m)$.

The inter-derivations relation $\preccurlyeq$ over the derivations of the hypergraph is defined as follows. Given two derivation $d_1 = (e_1, (i_1, j_1))$ and $d_2 = (e_2, (i_2, j_2))$, we define $d_1 \preccurlyeq d_2 \Leftrightarrow e_1 = e_2, i_1 \leq i_2, j_1 \leq j_2$. Note that $\preccurlyeq$ is the relation that must preserve monotonicity to ensure the correctness of the algorithms proposed by Huang and Chiang.

Scoring functions. Below we exemplified two types of monotonic scoring functions $s_c^1, s_c^2 \in \Re$, defined for any derivation $d = (e, < i, j >)$ and obeying monotonicity according to the inter-derivations relation.

1) The addition function,
$$s_c^1(d) = D^i\big(t_1(e)\big) + D^j\big(t_2(e)\big) + c(e)$$
2) The weighted average function, $s_c^2(d) = \frac{size(t_1(e)) \times D^i(t_1(e)) + size(t_2(e)) \times D^j(t_2(e)) + c(e)}{size(head(e))}$, where $size(v)$ is defined as the number of edges in the subtrees corresponding to the hypernode v. Since all derivations competing for the same hypernode (except for the pseudo root hypernode, see note below) correspond to subtrees that have the same number of edges, the weighted average function is monotonic.

Notes:
- For the pseudo root hypernode, the scoring scheme is defined $s_c(d) = D^i\big(t_1(e)\big)$, hence preserves monotonicity.
- $D^i\big(t_l(e)\big)$ is the score of the *i'th* derivation in the k-best list $D\big(t_l(e)\big)$.
- $t_l(e)$ is the *l'th* hypernode in the *tail* of e.

4. Search Space Pruning

Based on the dynamic realization, during the search, we can prune away hypernodes that cannot take part in a full solution that is scored high enough to be included in the k-best list across all color-coding iterations. For this, the optimal score of each hypernode *β(v)* is augmented with an optimistic estimation *α(v)* of the remaining score potential. For each hypernode *v*, the merit of *v*, *αβ(v) = α(v) + β(v)* (defined for the addition function scoring scheme), is an optimistic upper bound on the score of a full solution in which hypernode *v* participates. The merit *αβ(v)* is compared with a threshold c^k, which is monotonically updated between consecutive color-coding iterations to store the value of the highest scoring *k*-ranked entry observed so far. While cost *β(v)* is computed during the bottom-up construction of the hypergraph for each hypernode *v*, *α(v)* is computed in pre-processing as follows.
In the grammar-based scoring approach, we use an Inside-Outside algorithm to compute the Viterbi inside cost *β(v)* and the Viterbi outside cost *α(v)* for each hypernode *v* in the hypergraph. The outside parse-score [3-4], *α(v),* is computed in preprocessing, as in Klein and Manning et al. [4] on a leaf-set consisting of all possible terminal productions for the given grammar, rather than on the specific network.
In the network-based scoring approach, we compute *α(v)* as a naive estimation of the number of edges that could be added, multiplied by the maximal edge-score in *G*. Specifically, for hypernode *(W,v,q)*, we define $\alpha(v) = (m - |q|) * (E^H_{max} + V^H_{max})$, where E^H_{max} and V^H_{max} are the maximal edge- and node-score in *G*, respectively.

5. Grammars Used in the Results

A. The grammar used on KEGG pathways

This grammar accepts trees composed of pathway-proteins interacting with the virus (marked with *x*). Each such protein *x* contributes +4 to the total score. The viral-interactors are either connected directly or connected indirectly via a gap pathway-protein that does not interact with the virus (marked with *g*). Gap pathway proteins can be augmented with viral-interactors that do not belong to the pathway (marked with *v*). Each such protein *v* contributes +1 to the total score.

$$A = \{N, \Sigma, P, S\} \qquad N = \{S, G, X, V\} \qquad P = \left\{\begin{array}{c} S \to x\left(\{X\}^* \{S \mid G \mid \epsilon\}^*\right), \quad score = 0 \\ G \to g(S\,\{V\}^*), \quad score = 0 \\ X \to x, \quad score = 4 \\ V \to v\,, \quad score = 1 \end{array}\right\} \qquad \Sigma = \{x\,, g\,, v\}$$

x : *pathway protein interacting with the virus*
g: *pathway protein not interacting with any virus*
v: *non pathway protein interacting with the virus*

B. The grammars used on the differentially expressed genes of influenza virus

The regular tree grammars used for the application on the up-regulated (a) and the down-regulated (b) genes. The grammar is designed to accept trees, in which for each protein-node, its child-nodes are proteins that are either classified to the same time-point or to the following time-point. For example, rule 1 in grammar 1 defines that the derivation must start at the non-terminal symbol S_{15}, meaning that the accepted tree must be rooted in a protein-node labelled t_1 (i.e. classified to time-point 1). In addition, the accepted trees must have at least one leaf labeled t_5, in order to obtain trees that depict a full perspective of cellular dynamics across all time-points. This constraint is implemented in the non-terminal symbols S_{15}, S_{25}, S_{35} and S_{45}, each of which defines a subtree that has at least one leaf labelled t_5 and is rooted in protein-node labelled t_1, t_2, t_3 or t_4, respectively.

1. The grammar applied on the up-regulated genes

$$A = \{N, \Sigma, P, S_{15}\} \quad N = \{S_{15}, S_{25}, S_{35}, S_{45}, S_1, S_2, S_3, S_4, S_5\} \quad P = \left\{\begin{array}{c} S_{15} \to t_1(\{S_1|S_2\}^* \cdot \{S_{15}|S_{25}\}) \\ S_{25} \to t_2(\{S_2|S_3\}^* \cdot \{S_{25}|S_{35}\}) \\ S_{35} \to t_3(\{S_3|S_4\}^* \cdot \{S_{35}|S_{45}\}) \\ S_{45} \to t_4(\{S_4|S_5\}^* \cdot \{S_{45}|S_5\}) \\ S_1 \to t_1(\{S_1|S_2\}^*) \\ S_2 \to t_2(\{S_2|S_3\}^*) \\ S_3 \to t_3(\{S_3|S_4\}^*) \\ S_4 \to t_4(\{S_4|S_5\}^*) \\ S_5 \to t_5(\{S_5\}^*) \end{array}\right\} \quad \Sigma = \{t_1, t_2, t_3, t_4, t_5\}$$

2. The grammar applied on the down-regulated genes

$$A = \{N, \Sigma, P, S_{13}\} \quad N = \{S_{13}, S_{23}, S_1, S_2, S_3\} \quad P = \left\{\begin{array}{c} S_{13} \to t_1(\{S_1|S_2\}^* \cdot \{S_{15}|S_{25}\}) \\ S_{23} \to t_2(\{S_2|S_3\}^* \cdot \{S_{25}|S_{35}\}) \\ S_1 \to t_1(\{S_1|S_2\}^*) \\ S_2 \to t_2(\{S_2|S_3\}^*) \\ S_3 \to t_3(\{S_3\}^*) \end{array}\right. \quad \Sigma = \{t_1, t_2, t_3\}$$

$$\begin{array}{l} \boldsymbol{t_1} : \textbf{gene up} - \textbf{regulated at time} - \textbf{point 1} \\ \boldsymbol{t_2} : \textbf{gene up} - \textbf{regulated at time} - \textbf{point 2} \\ \boldsymbol{t_3} : \textbf{gene up} - \textbf{regulated at time} - \textbf{point 3} \\ \boldsymbol{t_4} : \textbf{gene up} - \textbf{regulated at time} - \textbf{point 4} \\ \boldsymbol{t_5} : \textbf{gene up} - \textbf{regulated at time} - \textbf{point 5} \end{array}$$

6. Extended Results

Identifying human pathways that specific viruses heavily rely on. In the manuscript we detailed the example of the TGF-β signaling pathway interaction with several viruses. Another example for a human pathway that interacts with multiple viruses is the antigen processing and presentation through the major histocompatibility complex class I (MHC1). This pathway is of special importance to viruses as it triggers the immune system to eliminate infected cells [5] (Figure S4A). Indeed, four viruses, EBV, HCV, PYV and HIV, were found to interact with both MHC1 and with chaperons that enable its assembly (Figure S4B). The fact that H1N1 is only weakly linked to this pathway is due to the fact that this virus escapes the immune system via increased mutations in viral proteins, rather than via viral-host interactions [6]. Similarly to the example in the paper, RTGnet identified top-scoring subtrees with different topologies for each virus, demonstrating the power of RTG to identify flexible patterns in a single query. To rigorously identify pathways that viruses densely interact with, we applied the grammar described in the manuscript to each pathway in the KEGG database. We focused on four evolutionarily-distant viruses whose entire proteome was screened for interactions with the human proteins: H1N1 and HCV (two RNA viruses), EBV (a DNA virus), and HIV1 (a retro-virus). As can be expected, several pathways scored high for all viruses, including "cell cycle" and "chemokine signaling" pathways, while other pathways scored high for a subset of the viruses. For example, H1N1, an RNA virus that does not establish chronic or latent infections, was the only virus not interacting with the "TGF-β signaling pathway", while HIV, the AIDS virus, was the only virus that densely interacted with the "primary immunodeficiency" pathway. Hierarchical clustering (Figure S4C) of these results by their Spearman correlation revealed that the three persistent viruses, EBV, HCV and HIV, are clustered together, while the acute virus, H1N1, has a relatively distinct profile. Therefore, the choice of which cellular process to perturb is mainly affected by the type of infection (persistent vs. acute) caused by the virus.

Methods: Human protein-protein interactions were gathered from the STRING database[7]. These contained ~69,000 interactions between ~11,000 proteins. Virus-host interactions downloaded from publicly available interaction databases (INTACT, BIOGRID, MINT, MENTHA, APID ,BIND) using PSIQUIC.

Virus	**Total number of interacting human proteins in the human network**
EBV	1169
HCV	710
HIV	967
H1N1	270
HPV	846
PYV	348

Figures

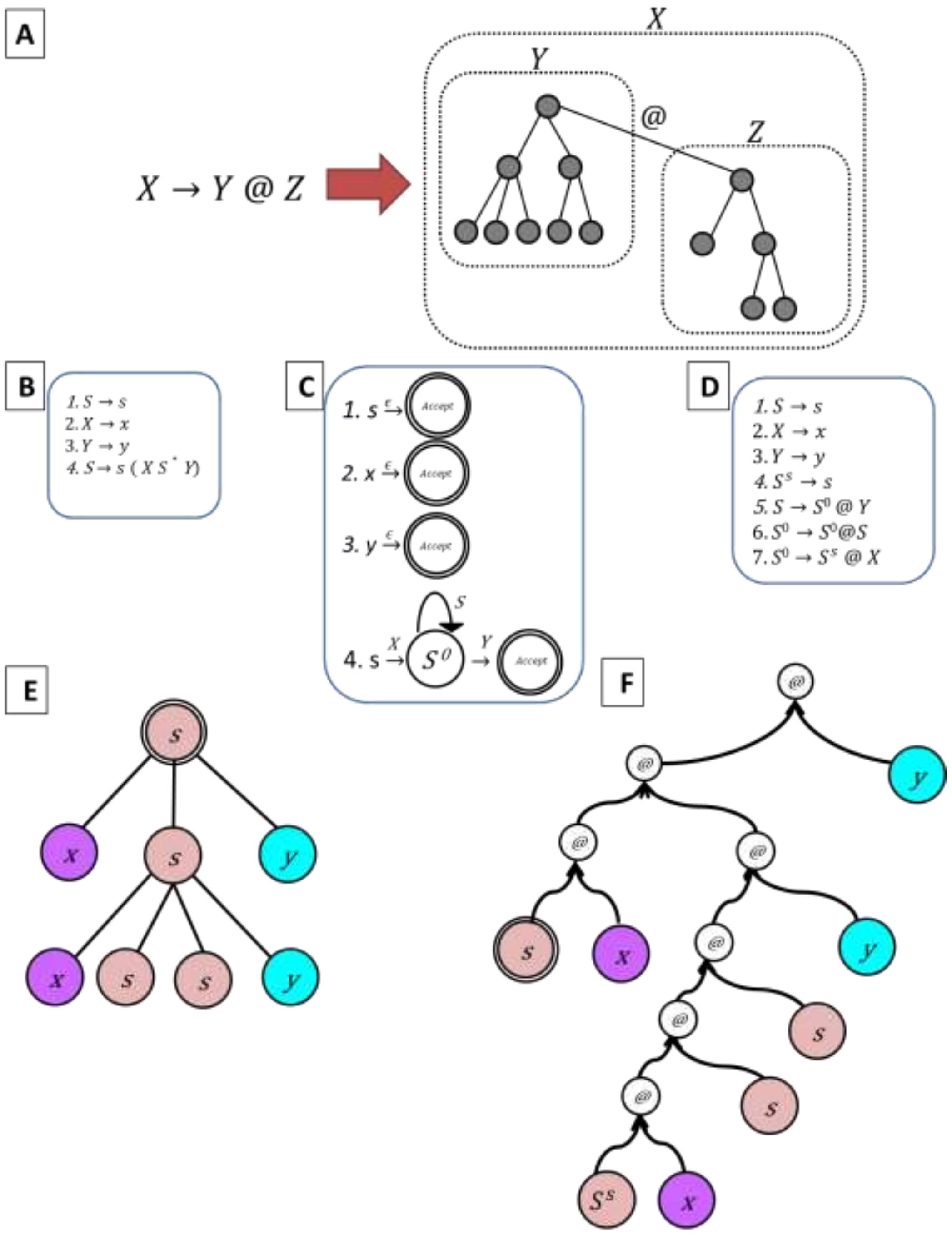

Figure S1. Curryfication of a Regular Tree Grammar and the corresponding curryfication of the tree to be parsed. (A) Each production rule *X→Y@Z* generates a tree, such that the tree represented by the non-terminal symbol *X* is composed of the two subtrees that are represented by non-terminal symbols *Y* and *Z*, connected to each other with an edge (marked with @). The composition of the tree represented by *X*, as an ordered rooted subtree, sets the subtree represented by *Z* as the right-most child of the root of the subtree represented by *Y*. **(B)** A regular tree grammar. **(C)** The regular tree grammar in B, given as a deterministic automaton for each production rule. **(D)** The Curry-Encoding of the RTG in B, produced by de-composing the automatons that are showed in C. Each edge in each automaton corresponds to a unique production-rule in the Curry-Encoded RTG. **(E)** a tree accepted by the RTG in B. **(F)** the curry-encoding of the tree in E is accepted by Curry-Encoded RTG in D.

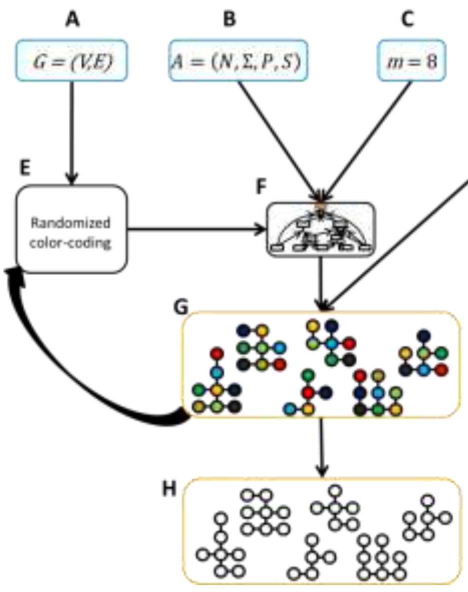

Figure S2. The basic workflow of our framework. The input to our algorithm consists of (A) a node-labeled directed graph *G*, representing a biological network, (B) an unranked regular tree grammar *A*, (C) a parameters *m* that limits the number of vertices in the sought trees, (D) a parameter *k* specifying the number of top-scoring results to compute. (E) The algorithm then applies color-coding to *G*, by coloring each node randomly in one of *m* colors. (F) This is followed by a bottom-up parsing of the colored graph to create a hypergraph that encodes all candidate solutions. (G) The hypergraph is traversed to output the *k*-best highest scoring colorful trees. (H) This procedure (E-G) is repeated iteratively for a given input, in order to probabilistically limit the random error introduced by color-coding. After each iteration, the final list of k-best trees is merged with the new list of k-best trees computed in this iteration.

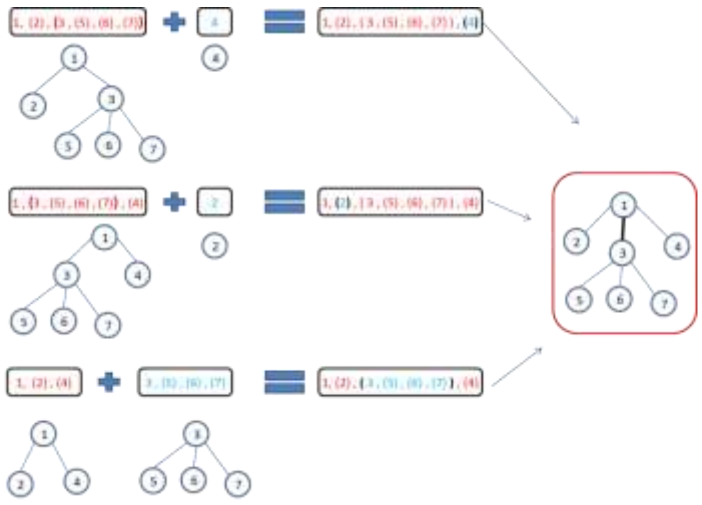

Figure S3. The canonical code. This figure demonstrates how three distinct derivations can produce three isomorphic trees, all encoded with the same canonical code. The canonical code of a tree τ (to the right) is composed from the canonical codes of the subtrees used to compose τ. In this example, the root node of τ has three children encoded as "(2)", "(4)" and "(3, (5), (6), (7))". There are three combinations to compose τ, and in each combination a different child is the right child. For example, in the bottom equation, "(2)" and "(4)" are the children of the root of the left-subtree and can be recovered from its canonical code "1, (2), (4)". The third child, "(3, (5), (6), (7))" is the full canonical code of the right-subtree. While constructing the canonical code of the tree, "(3, (5), (6), (7))" is inserted in the middle of "(2)" and "(4)", due to the lexicographical ordering, while the root-node remains unchanged. This ordering guarantees that any combination to compose τ would yield an identical canonical encoding "1, (2), (3, (5), (6), (7)), (4)".

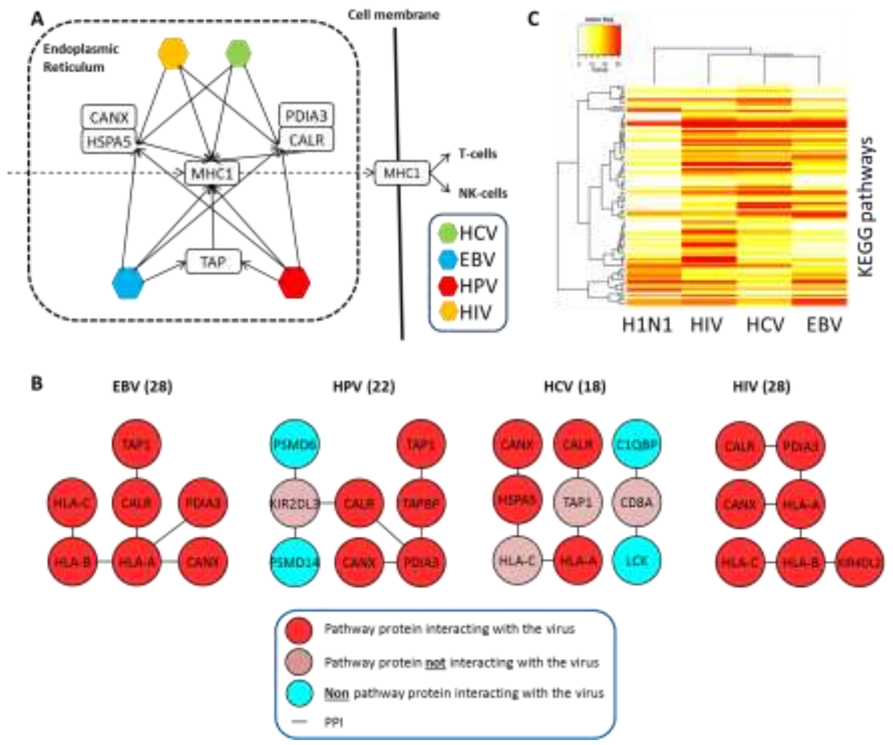

Figure S4. Extended results. (A) MHC1 antigens are assembled within the endoplasmic reticulum lumen and presented on the cellular membrane together with pathogenic peptides to be recognized by T-cells and thus trigger the immune-response. EBV, HIV, HPV and HCV perturb MHC1 processing by binding to MHC1 itself and to chaperone complexes that facilitate its folding. EBV and HPV also directly interact with TAP, a complex responsible for translocation of peptides from the cytosol into the lumen and for peptide-binding during MHC1 assembly. **(B)** Maximal scoring subtrees per virus (scores appear inside parenthesis) in the MHC1 pathway. **(C)** A heatmap showing the scores of the top-scoring subtrees found for each virus and for each KEGG pathway. Rows correspond to KEGG pathways and columns correspond to viruses. Each entry contains a value between 0-1: This value corresponds to the highest score obtained for the corresponding virus and pathway, divided by the highest score obtained for this virus across all pathways. KEGG pathways scoring less than 5 for all viruses were omitted. Hierarchical clustering was performed using the distance function *d = (1-r)/2*, where *r* is Spearman correlation coefficient.

References

1. Lange, M. and H. Leiß, *To CNF or not to CNF? An efficient yet presentable version of the CYK algorithm.* Informatica Didactica, 2009. **8**: p. 2008-2010.
2. Huang, L. and D. Chiang. *Better k-best parsing*. in *Proceedings of the Ninth International Workshop on Parsing Technology*. 2005: Association for Computational Linguistics.
3. Nielsen, L.R., D. Pretolani, and K.A. Andersen, *Finding the K shortest hyperpaths using reoptimization.* Operations Research Letters, 2006. **34**(2): p. 155-164.
4. Klein, D. and C.D. Manning, *A parsing: fast exact Viterbi parse selection*, in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*. 2003, Association for Computational Linguistics: Edmonton, Canada. p. 40-47.
5. Heath, W.R. and F.R. Carbone, *Cross-presentation in viral immunity and self-tolerance.* Nat Rev Immunol, 2001. **1**(2): p. 126-34.
6. van de Sandt, C.E., J.H. Kreijtz, and G.F. Rimmelzwaan, *Evasion of influenza A viruses from innate and adaptive immune responses.* Viruses, 2012. **4**(9): p. 1438-76.
7. Franceschini, A., D. Szklarczyk, S. Frankild, M. Kuhn, M. Simonovic, A. Roth, J. Lin, P. Minguez, P. Bork, C. von Mering, and L.J. Jensen, *STRING v9.1: protein-protein interaction networks, with increased coverage and integration.* Nucleic Acids Res, 2013. **41**(Database issue): p. D808-15.
8. Zhou, Y., X. Wang, Y. Huang, Y. Chen, G. Zhao, Q. Yao, C. Jin, Y. Huang, X. Liu, and G. Li, *Down-regulated SOX4 Expression Suppresses Cell Proliferation, Metastasis and Induces Apoptosis in Xuanwei Female Lung Cancer Patients.* J Cell Biochem, 2015. **116**(6): p. 1007-18.
9. Pramoonjago, P., A.S. Baras, and C.A. Moskaluk, *Knockdown of Sox4 expression by RNAi induces apoptosis in ACC3 cells.* Oncogene, 2006. **25**(41): p. 5626-39.
10. Pan, X., H. Li, P. Zhang, B. Jin, J. Man, L. Tian, G. Su, J. Zhao, W. Li, H. Liu, W. Gong, T. Zhou, and X. Zhang, *Ubc9 interacts with SOX4 and represses its transcriptional activity.* Biochem Biophys Res Commun, 2006. **344**(3): p. 727-34.
11. Saltzman, A., G. Searfoss, C. Marcireau, M. Stone, R. Ressner, R. Munro, C. Franks, J. D'Alonzo, B. Tocque, M. Jaye, and Y. Ivashchenko, *hUBC9 associates with MEKK1 and type I TNF-alpha receptor and stimulates NFkappaB activity.* FEBS Lett, 1998. **425**(3): p. 431-5.
12. Hsu, H., H.B. Shu, M.G. Pan, and D.V. Goeddel, *TRADD-TRAF2 and TRADD-FADD interactions define two distinct TNF receptor 1 signal transduction pathways.* Cell, 1996. **84**(2): p. 299-308.
13. Bouwmeester, T., A. Bauch, H. Ruffner, P.O. Angrand, G. Bergamini, K. Croughton, C. Cruciat, D. Eberhard, J. Gagneur, S. Ghidelli, C. Hopf, B. Huhse, R. Mangano, A.M. Michon, M. Schirle, J. Schlegl, M. Schwab, M.A. Stein, A. Bauer, G. Casari, G. Drewes, A.C. Gavin, D.B. Jackson, G. Joberty, G. Neubauer, J. Rick, B. Kuster, and G. Superti-Furga, *A physical and functional map of the human TNF-alpha/NF-kappa B signal transduction pathway.* Nat Cell Biol, 2004. **6**(2): p. 97-105.
14. Tomoiu, A., A. Gravel, R.M. Tanguay, and L. Flamand, *Functional interaction between human herpesvirus 6 immediate-early 2 protein and ubiquitin-conjugating enzyme 9 in the absence of sumoylation.* J Virol, 2006. **80**(20): p. 10218-28.
15. Hateboer, G., E.M. Hijmans, J.B. Nooij, S. Schlenker, S. Jentsch, and R. Bernards, *mUBC9, a novel adenovirus E1A-interacting protein that complements a yeast cell cycle defect.* J Biol Chem, 1996. **271**(42): p. 25906-11.
16. Shapira, S.D., I. Gat-Viks, B.O. Shum, A. Dricot, M.M. de Grace, L. Wu, P.B. Gupta, T. Hao, S.J. Silver, D.E. Root, D.E. Hill, A. Regev, and N. Hacohen, *A physical and regulatory map of host-influenza interactions reveals pathways in H1N1 infection.* Cell, 2009. **139**(7): p. 1255-67.
17. Higuchi, T., T. Nakayama, T. Arao, K. Nishio, and O. Yoshie, *SOX4 is a direct target gene of FRA-2 and induces expression of HDAC8 in adult T-cell leukemia/lymphoma.* Blood, 2013. **121**(18): p. 3640-9.

18. Agarwal, S.K., S.C. Guru, C. Heppner, M.R. Erdos, R.M. Collins, S.Y. Park, S. Saggar, S.C. Chandrasekharappa, F.S. Collins, A.M. Spiegel, S.J. Marx, and A.L. Burns, *Menin interacts with the AP1 transcription factor JunD and represses JunD-activated transcription.* Cell, 1999. **96**(1): p. 143-52.
19. Obungu, V.H., A. Lee Burns, S.K. Agarwal, S.C. Chandrasekharapa, R.S. Adelstein, and S.J. Marx, *Menin, a tumor suppressor, associates with nonmuscle myosin II-A heavy chain.* Oncogene, 2003. **22**(41): p. 6347-58.
20. Watanabe, T., E. Kawakami, J.E. Shoemaker, T.J. Lopes, Y. Matsuoka, Y. Tomita, H. Kozuka-Hata, T. Gorai, T. Kuwahara, E. Takeda, A. Nagata, R. Takano, M. Kiso, M. Yamashita, Y. Sakai-Tagawa, H. Katsura, N. Nonaka, H. Fujii, K. Fujii, Y. Sugita, T. Noda, H. Goto, S. Fukuyama, S. Watanabe, G. Neumann, M. Oyama, H. Kitano, and Y. Kawaoka, *Influenza virus-host interactome screen as a platform for antiviral drug development.* Cell Host Microbe, 2014. **16**(6): p. 795-805.