RANGER-DTLx Manual

https://compbio.engr.uconn.edu/ranger-dtlx/

Description

RANGER-DTLx is an extended version of the Ranger-DTL program from the RANGER-DTL 2.0 software package and can be used for inferring gene family evolution by speciation, gene duplication, horizontal gene transfer, and gene loss. The prototype implementation of RANGER-DTLx available here takes as input a rooted gene tree and a rooted species tree and reconciles the two by postulating speciation, duplication, transfer, and loss events. The distinguishing feature of RANGER-DTLx is that it implements an extended version of the Duplication-Transfer-Loss (DTL) reconciliation model, called the DTLx reconciliation model, that can account for more complex horizontal gene transfer scenarios, namely transfer-loss and transfers from extinct or unsampled species lineage, denoted as TL and TX events, respectively. Further details on the DTLx reconciliation model appear in the manuscript cited below.

> *Improved Duplication-Transfer-Loss Reconciliation with Extinct and Unsampled Lineages*
> Samson Weiner, Mukul S. Bansal
> Under review

RANGER-DTLx is freely available from https://compbio.engr.uconn.edu/ranger-dtlx/

Usage

Core Programs: RANGER-DTLx consists of two programs:

- *Ranger-DTLx* takes as input a rooted species tree and a rooted gene tree and computes an optimal DTLx reconciliation for the gene tree and species tree. In the case of multiple optimal reconciliations, the output is an optimal reconciliation sampled uniformly at random among all optimal reconciliations. This program can (and should) be run multiple times on the same species tree and gene tree pair to sample the space of all optimal reconciliations.
- *AggregateRanger* takes as input a series of output reconciliation files computed using Ranger-DTLx on the same gene tree and species tree. The files must be created by executing Ranger-DTLx on the same input gene tree and species tree multiple times with either the same or different event cost assignments. AggregateRanger is useful for computing support values for individual events and mappings by accounting for variance due to multiple optimal reconciliations and alternative event cost assignments.

Input File Format for Ranger-DTLx: Ranger-DTLx takes as input a single file (specified using the –i command line option) containing a rooted species tree on the first line and a rooted gene tree on the second line. All input trees must be expressed using the Newick format terminated by a semicolon, and they must be fully binary (fully resolved). Species names in the species tree must be unique.

E.g., `((speciesA, speciesB), speciesC);`

Each leaf in the gene tree must be labeled with the name of the species from which that gene was sampled. If desired, the gene name can be appended to the species name separated by an underscore '_' character.

E.g., `(((speciesA_gene1, speciesC_gene1), speciesB_geneX), speciesC_gene2);`

Input File Format for AggregateRanger: AggregateRanger takes as input a set of reconciliation files computed using Ranger-DTLx. Each of the reconciliation files must be named using a particular naming scheme: All of the reconciliation files to be aggregated must have the same file name prefix, appended with consecutive numbers starting from 1. For example, if there are 200 reconciliation files to be used in the analysis and the chosen file name prefix is "reconciliation", then the 200 files must be named reconciliation1, reconciliation2, …, reconciliation200. To have AggregateRanger analyze and aggregate these files, one must pass the file name prefix as a command line option to the program.

Specifying Event Costs and Usage for Ranger-DTLx: A numerical cost is required for each of the event types in duplication, transfer, loss, transfer-loss, and transfer from extinct/unsampled lineage (specified by using the -D, -T, -L, -TL, and -TX options, respectively) to be used in the reconciliation. If no cost is specified, the default value is used instead. Allowing for TL and TX events in the reconciliation can be toggled (specified by using --noTL and --noTX, respectively). If no event usage is specified, TL and TX events are enabled by default. Note that TL events cannot be disabled if TX events are enabled. Lastly, each of the five event costs must be positive integers.

Variable Transfer Cost Usage for Ranger-DTLx: Ranger-DTLx allows the user to apply variable cost assignments to transfer events. In particular, the program features two variable transfer cost options (specified by using --type 1 and --type 2) in addition to the default fixed cost assignment, which is non-distance-dependent (--type 0). Importantly, the chosen transfer type is not only applied to standard transfer events, but also to TL and TX events. When using Type 1 transfer costs, the user specifies a threshold and an "additional transfer cost" parameter. If the distance (in the number of edges) between the donor and recipient species for the transfer (resp. TL or TX event) is less than the threshold, the regular transfer cost (resp. TL or TX cost) is used, and for any distance larger than or equal to the threshold, the regular transfer cost (resp. TL or TX cost) + additional transfer cost is used. Type 2 variable transfer costs are a generalized version of type 1 costs, in that the transfer cost (resp. TL or TX cost) for a transfer (resp. TL or TX event) is given by regular transfer cost (resp. TL or TX cost) + \floor((distance between donor and recipient) /threshold) * (additional transfer cost). Note that a single additional transfer cost is used for transfers, TL events, and TX events.

Sample Commands

```
./Ranger-DTLx --seed 1 -i inputFile.newick -o output_folder/reconciliation1

./Ranger-DTLx --seed 2 -i inputFile.newick -o output_folder/reconciliation2

./AggregateRanger output_folder/reconciliation
```

Available command line options for Ranger-DTLx

-i, --input	Input file.
-o, --output	Output file.
-D,	Duplication cost (whole number only, default value 2).
-T,	Transfer cost (whole number only, default value 3).
-L,	Loss cost (whole number only, default value 1).
-TL,	Transfer-Loss cost (whole number only, default value 4).
-TX,	Cost for transfer from extinct/unsampled lineage (whole number only, default value 4).
--noTL <bool>	Disable TL events (default = False).
--noTX <bool>	Disable TX events (default = False).
--type 0|1|2	Type of transfer to use: 0 - Fixed transfer cost (default). 1 - Simple threshold based variable transfer cost. 2 - General threshold based variable transfer cost.
--thr <whole number>	Threshold value for use with types 1 and 2 (default 10).
--add <whole number>	Additional transfer cost for use with types 1 and 2 above (default value is transfer cost divided by 2, rounded down).
--seed <whole number>	Set a user defined random number generator seed.
-q, --quiet	No process output.
-s, --summary	Only output summary statistics.
-v, --version	Program version.
-h, --help	Brief help message.

Interpreting the output

Ranger-DTLx: The reconciliation output begins by printing both the input species tree and input gene tree, where the internal nodes of each tree are labelled. What follows is the reconciliation itself, which details a DTLx scenario between an augmented gene tree and the augmented species tree. The reconciliation format closely follows that of RANGER-DTL 2.0, where the new format is distinguished by the possibility of the gene tree containing dummy nodes representing either TL or TX events, in addition to the presence of extra branches in the species tree representing extinct or unsampled lineages. Dummy nodes will only be present in the augmented gene tree if they facilitate either a TL or TX event in the reconciliation. Each node of the augmented gene tree is printed sequentially in post order, along with that specific node's event and mapping assignment. Every original gene tree node is labeled either as a leaf node, a speciation node, a duplication node, or a transfer node, and is assigned a mapping to some node from the species tree (this mapping defines the embedding of the gene tree into the species tree). Similarly, every dummy node is labelled as either a TL node (if it maps to a species node from the original species tree) or a TX node (if it maps to an extra branch in the augmented species tree). For any transfer event, standard or complex (i.e. T, TL, or TX), in addition to the mapping (which, in this case, represents the donor species), the recipient species is also specified. Lastly, a summary of the reconciliation is printed detailing the total minimum reconciliation cost, number of optimal solutions, and, importantly, the number of duplication, transfer, loss, TL, and TX events.

When printing the gene tree nodes, if a node is a leaf, the name assigned to that leaf will be used to identify the node.

E.g., `specA_gene1: Leaf Node`

Internal nodes that appear in the original gene tree (i.e. not dummy nodes) are represented by the label assigned previously, and always starting with `m`. What follows is the least common ancestor describing the location of the node in the gene tree, the event type, and the mapping. The recipient is included if it is a transfer event.

E.g., `m5 = LCA[specA_1, specB_2]: Transfer, Mapping → specA, Recipient → specD`

Dummy nodes are identified depending on their location in the gene tree. If the dummy node is the parent of a leaf node, the name assigned to that leaf is used; otherwise if the dummy node is the parent of an internal node, the name assigned to that internal node is used except the `m` is replaced with a `d`. The node is also clearly marked as *dummy node* following the label.

E.g., `d5 = Dummy Node[m5], TL, Mapping → specA, Recipient → specD`

Added nodes and leaves in the augmented species tree always begin with 'e' in the output. If a gene tree node maps to an added branch, the location of the added branch in the augmented species tree is specified by identifying the first original species tree node on the path from the branch to the root, labelled as the *parent* for convenience. In other words, the added branch exists along the edge below the node represented by *parent*. If *parent* appears as *None*, the added branch in question appears above the root of the input species tree. Finally, if the word *edge* appears, the mapped node points to an *extra leaf*. Otherwise, the mapped node points to an *extra node* subdividing an original species tree edge.

E.g., `d5 = Dummy Node[m5], TX, Mapping → e10, Edge, Parent = specB, Recipient → specD`

AggregateRanger: The output of AggregateRanger is similar to that of Ranger-DTLx. Each augmented gene tree node is listed in post order (one node per line) and shows the number of times that node was labeled as a speciation, duplication, transfer, TL event, or TX event in the input set of sampled reconciliations. Because dummy nodes only appear in the reconciliation if they represent a TL or TX event, the number of occurrences may not equal the total number of reconciliations considered. For each node, the most frequent mapping for that node among the input reconciliations is shown, along with the number of times that mapping was seen. The output of AggregateRanger thus provides support values for the event type and mapping of each node in the augmented gene tree. For original gene tree nodes, these support values are computed based on the fraction (percentage) of sampled reconciliations supporting that event assignment or mapping assignment. For dummy nodes, however, the support values are computed based on the fraction of sampled reconciliations the dummy node, event, mapping assignment, and recipient appear in. AggregateRanger can be used to aggregate reconciliations and compute support values across multiple optima as well as across different event cost assignments.

Test File

The software directory includes a sample input file located in the TEST_DATA directory. The software can be executed with default parameters using the following command:

```
./Ranger-DTLx -i TEST_DATA/testFile_1.newick
```

Contact

If there are any questions related to RANGER-DTLx, please contact Samson Weiner (samson.weiner@uconn.edu) or Mukul Bansal (mukul.bansal@uconn.edu), or visit https://compbio.engr.uconn.edu/ for a complete list of available software and datasets.