Applications

More and more research effort has been done to study the non-cooperative games recently. Among these various forms of games, the unicast/multicast routing game [2,5,6] and multicast cost sharing game [1,3,4] have received a considerable amount of attentions over the past few year due to its application in the Internet. However, both unicast/multicast routing game and multicast cost sharing game are one folded: the unicast/multicast routing game does not take the receivers into account while the multicast cost sharing game does not treat the links as non-cooperative. In this paper, they study the scenario, which was called *multicast system*, in which both the links and the receivers could be non-cooperative. Solving this problem paving a way for the real world commercial multicast and unicast application. A few examples are, but not limited to, the multicast of the video content in wireless mesh network and commercial WiFi system; the multicast routing in the core Internet.

Open Problems

A number of problems related to the work of Wang, Li and Chu [7] remain open. The first and foremost, the upper bound and lower bound on α still have a gap of r if the multicast system is α-stable; and a gap of $2r$ if the multicast system is α-Nash stable.

The second, Wang, Li and Chu only showed the existence of the Nash Equilibrium under their systems. They have not characterized the convergence of the Nash Equilibrium and the strategies of the user, which are not only interesting but also important problems.

Cross References

► Non-approximability of Bimatrix Nash Equilibria

Recommended Reading

1. Feigenbaum, J., Papadimitriou, C.H., Shenker, S.: Sharing the cost of multicast transmissions. J. Comput. Syst. Sci. **63**, 21–41 (2001)
2. Kao, M.-Y., Li, X.-Y., Wang, W.: Towards truthful mechanisms for binary demand games: A general framework. In: ACM EC, pp. 213–222, Vancouver, Canada (2005)
3. Herzog, S., Shenker, S., Estrin, D.: Sharing the "cost" of multicast trees: an axiomatic analysis. IEEE/ACM Trans. Netw. **5**, 847–860 (1997)
4. Moulin, H., Shenker, S.: Strategyproof sharing of submodular costs: budget balance versus efficiency. Econ. Theory **18**, 511–533 (2001)
5. Wang, W., Li, X.-Y., Sun, Z., Wang, Y.: Design multicast protocols for non-cooperative networks. In: Proceedings of the 24th IEEE INFOCOM. vol. 3, pp. 1596–1607, Miami, USA (2005)
6. Wang, W., Li, X.-Y., Wang, Y.: Truthful multicast in selfish wireless networks. In: Proceedings of the 10th ACM MOBICOM, pp. 245–259, Philadelphia, USA (2004)
7. Wang, W., Li, X.-Y., Chu, X.: Nash equilibria, dominant strategies in routing. In: Workshop for Internet and Network Economics (WINE). Lecture Notes in Computer Science, vol. 3828, pp 979–988. Springer, Hong Kong, China (2005)

Navigation

► Mobile Agents and Exploration
► Robotics

Nearest Neighbor Interchange and Related Distances

1999; DasGupta, He, Jiang, Li, Tromp, Zhang

Bhaskar DasGupta[1], Xin He[2], Tao Jiang[3], Ming Li[4], John Tromp[5], Louxin Zhang[6]

[1] Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA
[2] Department of Computer Science and Engineering, University at Buffalo The State University of New York, Buffalo, NY, USA
[3] Department of Computer Science and Engineering, University of California at Riverside, Riverside, CA, USA
[4] Department of Computer Science, University of Waterloo, Waterloo, ON, Canada
[5] CWI, Amsterdam, Netherlands
[6] Department of Mathematics, National University of Singapore, Singapore, Singapore

Keywords and Synonyms

Comparison of phylogenies; Network models of evolution

Problem Definition

In this chapter, the authors state results on some transformation based distances for *evolutionary trees*. Several distance models for evolutionary trees have been proposed in the literature. Among them, the best known is perhaps the *nearest neighbor interchange* (nni) distance introduced independently in [10] and [9]. The authors will focus on the nni distance and a closely related distance called the *subtree-transfer* distance originally introduced in [5,6]. Several papers that involved DasGupta, He, Jiang,

Li, Tromp and Zhang essentially showed the following results:

- A correspondence between the nni distance and the linear-cost subtree-transfer distance on unweighted trees;
- Computing the nni distance is NP-hard, but admits a fixed-parameter tractability and a logarithmic ratio approximation algorithms;
- A 2-approximation algorithm for the linear-cost subtree-transfer distance on weighted evolutionary trees.

The authors first define the nni and linear-cost subtree-transfer distances for unweighted trees. Then the authors extend the nni and linear-cost subtree-transfer distances to weighted trees. For the purpose of this chapter, an evolutionary tree (also called *phylogeny*) is an *unordered* tree, has uniquely labeled leaves and unlabeled interior nodes, can be *unrooted* or *rooted*, can be *unweighted* or *weighted*, and has all internal nodes of degree 3.

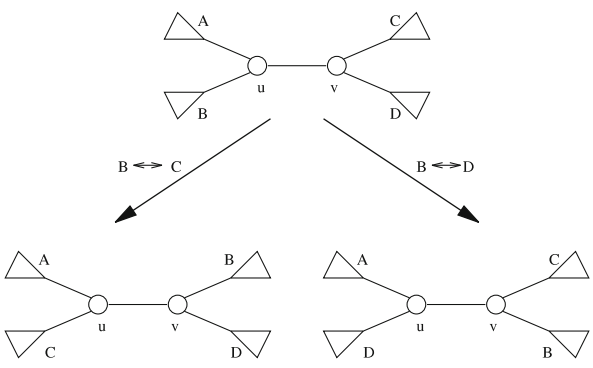
Unweighted Trees

An *nni* operation swaps two subtrees that are separated by an internal edge (u, v), as shown in Fig. 1.

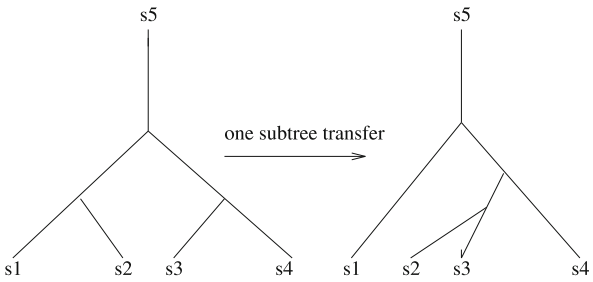
The nni operation is said to *operate* on this internal edge. The nni distance, $D_{\text{nni}}(T_1, T_2)$, between two trees T_1 and T_2 is defined as the *minimum* number of nni operations required to transform one tree into the other.

An nni operation can also be viewed as moving a subtree past a neighboring internal node. A more general operation is to transfer a subtree from one place to another arbitrary place. Figure 2 shows such a *subtree-transfer* operation.

The subtree-transfer distance between two trees T_1 and T_2 is the minimum number of subtrees one need to move to transform T_1 into T_2 [5,6,7]. It is sometimes



Nearest Neighbor Interchange and Related Distances, Figure 1
The two possible nni operations on an internal edge (u, v): exchange $B \leftrightarrow C$ or $B \leftrightarrow D$



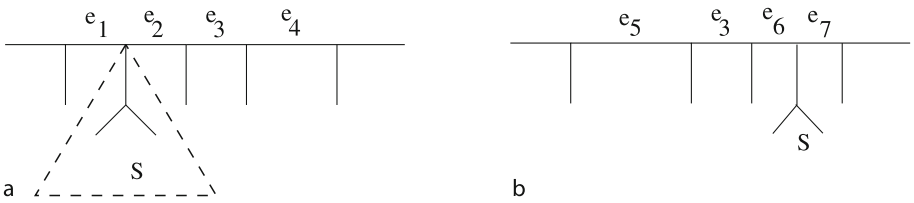
Nearest Neighbor Interchange and Related Distances, Figure 2
An example of subtree-transfer

appropriate in practice to discriminate among subtree-transfer operations as they occur with different frequencies. In this case, one can charge each subtree-transfer operation a cost equal to the distance (the number of nodes passed) that the subtree has moved in the current tree. The *linear-cost* subtree-transfer distance, $D_{\text{lcst}}(T_1, T_2)$, between two trees T_1 and T_2 is then the minimum total cost required to transform T_1 into T_2 by subtree-transfer operations [1,2].

Weighted Trees

Both the linear-cost subtree-transfer and nni models can be naturally extended to weighted trees. The extension for nni is straightforward: an nni operation is simply charged a cost equal to the weight of the edge it operates on. For feasibility of weighted nni transformation between two given weighted trees T_1 and T_2, one also requires that the following conditions are satisfied: (**1**) for each leaf label a, the weight of the edge in T_1 incident on a is the same as the weight of the edge in T_2 incident on a and (**2**) the multisets of weights of internal edges of T_1 and T_2 are the same.

In the case of linear-cost subtree-transfer, although the idea is immediate, i. e., a moving subtree should be charged for the weighted distance it travels, the formal definition needs some care and is given below. Consider (unrooted) trees in which each edge e has a weight $w(e) \geq 0$. To ensure feasibility of transforming a tree into another, One requires the total weight of all edges to equal one. A subtree-transfer is now defined as follows. Select a subtree S of T at a given node u and select an edge $e \notin S$. Split the edge e into two edges e_1 and e_2 with weights $w(e_1)$ and $w(e_2)$ $(w(e_1), w(e_2) \geq 0, w(e_1) + w(e_2) = w(e))$, and move S to the common end-point of e_1 and e_2. Finally, merge the two remaining edges e' and e'' adjacent to u into one edge with weight $w(e') + w(e'')$. The cost of this subtree-transfer is the total weight of all the edges over which S



Nearest Neighbor Interchange and Related Distances, Figure 3
Subtree-transfer on weighted phylogenies. Tree b is obtained from tree a with one subtree-transfer

is moved. Figure 1 gives an example. The edge-weights of the given tree are normalized so that their total sum is 1. The subtree S is transferred to split the edge e_4 to e_6 and e_7 such that $w(e_6), w(e_7) \geq 0$ and $w(e_6) + w(e_7) = w(e_4)$; finally, the two edges e_1 and e_2 are merged to e_5 such that $w(e_5) = w(e_1) + w(e_2)$. The cost of transferring S is $w(e_2) + w(e_3) + w(e_6)$.

Note that for weighted trees, the linear-cost subtree-transfer model is more general than the nni model in the sense that one can slide a subtree along an edge with subtree-transfers. Such an operation is not realizable with nni moves.

Key Results

Let T_1 and T_2 be the two trees, each with n nodes, that are being used in the distance computation.

Theorem 1 ([2,3,4]) *Assume that T_1 and T_2 are unweighted. Then, the following results hold:*

- $D_{nni}(T_1, T_2) = D_{lcst}(T_1, T_2)$.
- *Computing $D_{nni}(T_1, T_2)$ is NP-complete.*
- *Suppose that $D_{nni}(T_1, T_2) \leq d$. Then, an optimal sequence of nni operations transforming T_1 into T_2 can be computed in $O(n^2 \log n + n \cdot 2^{23d/2})$ time.*
- *$D_{nni}(T_1, T_2)$ can be approximated to within a factor of $\log n + O(1)$ in polynomial time.*

Theorem 2 ([1,2,3,4]) *Assume that T_1 and T_2 are weighted. Then, the following results hold:*

- *$D_{nni}(T_1, T_2)$ can be approximated to within a factor of $6 + 6 \log n$ in $O(n^2 \log n)$ time.*
- *Assume that T_1 and T_2 are allowed to have leaves that are not necessarily uniquely labeled. Then, computing $D_{lcst}(T_1, T_2)$ is NP-hard.*
- *$D_{lcst}(T_1, T_2)$ can be approximated to within a factor of 2 in $O(n^2 \log n)$ time.*

Applications

The results reported here are on transformation based distances for evolutionary trees. Such a tree is can be *rooted* if the evolutionary origin is known and can be *weighted* if the evolutionary length on each edge is known. Reconstructing the correct evolutionary tree for a set of species is one of the fundamental yet difficult problems in evolutionary genetics. Over the past few decades, many approaches for reconstructing evolutionary trees have been developed, including (not exhaustively) parsimony, compatibility, distance and maximum likelihood approaches. The outcomes of these methods usually depend on the data and the amount of computational resources applied. As a result, in practice they often lead to different trees on the same set of species [8]. It is thus of interest to compare evolutionary trees produced by different methods, or by the same method on different data.

Another motivation for investigating the linear-cost subtree transfer distance comes from the following motivation. When *recombination* of DNA sequences occurs in an evolution, two sequences meet and generate a new sequence, consisting of genetic material taken left of the recombination point from the first sequence and right of the point from the second sequence [5,6]. From a phylogenetic viewpoint, before the recombination, the ancestral material on the present sequence was located on two sequences, one having all the material to the left of the recombination point and another having all the material to the right of the breaking point. As a result, the evolutionary history can no longer be described by a single tree. The recombination event partitions the sequences into two neighboring regions. The history for the left and the right regions could be described by separate evolutionary trees. The recombination makes the two evolutionary trees describing neighboring regions differ. However, two neighbor trees cannot be arbitrarily different, one must be obtainable from the other by a *subtree-transfer operation*. When more than one recombination occurs, one can describe an evolutionary history using a list of evolutionary trees, each corresponds to some region of the sequences and each can be obtained by several subtree-transfer operations from its predecessor [6]. The computation of a linear-cost subtree-transfer distance is useful in reconstructing such a list of trees based on parsimony [5,6].

Open Problems

1. Is there a constant ratio approximation algorithm for the nni distance on unweighted evolutionary trees or is the $O(\log n)$-approximation the best possible?
2. Is the linear-cost subtree-transfer distance NP-hard to compute on weighted evolutionary trees if leaf labels are not allowed to be non-unique?
3. Can one improve the approximation ratio for linear-cost subtree-transfer distance on weighted evolutionary trees?

Cross References

- ▶ Constructing a Galled Phylogenetic Network
- ▶ Maximum Agreement Subtree (of 2 Binary Trees)
- ▶ Maximum Agreement Subtree (of 3 or More Trees)
- ▶ Phylogenetic Tree Construction from a Distance Matrix

Recommended Reading

1. DasGupta, B., He, X., Jiang, T., Li, M., Tromp, J.: On the linear-cost subtree-transfer distance. Algorithmica **25**(2), 176–195 (1999)
2. DasGupta, B., He, X., Jiang, T., Li, M., Tromp, J., Zhang, L.: On distances between phylogenetic trees, 8th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 427–436 (1997)
3. DasGupta, B., He, X., Jiang, T., Li, M., Tromp, J., Wang, L., Zhang, L.: Computing Distances between Evolutionary Trees. In: Du, D.Z., Pardalos, P.M. (eds.) Handbook of Combinatorial Optimization. Kluwer Academic Publishers, Norwell, **2**, 35–76 (1998)
4. DasGupta, B., He, X., Jiang, T., Li, M., Tromp, J., Zhang, L.: On Computing the Nearest Neighbor Interchange Distance. In: Du, D.Z., Pardalos, P.M., Wang, J. (eds.) Proceedings of the DIMACS Workshop on Discrete Problems with Medical Applications, DIMACS Series in Discrete Mathematics and Theoretical Computer Science. Am. Math. Soc. **55**, 125–143 (2000)
5. Hein, J.: Reconstructing evolution of sequences subject to recombination using parsimony. Math. Biosci. **98**, 185–200 (1990)
6. Hein, J.: A heuristic method to reconstruct the history of sequences subject to recombination. J. Mol. Evol. **36**, 396–405 (1993)
7. Hein, J., Jiang, T., Wang, L., Zhang, K.: On the complexity of comparing evolutionary trees. Discret. Appl. Math. **71**, 153–169 (1996)
8. Kuhner, M., Felsenstein, J.: A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. Mol. Biol. Evol. **11**(3), 459–468 (1994)
9. Moore, G.W., Goodman, M., Barnabas, J.: An iterative approach from the standpoint of the additive hypothesis to the dendrogram problem posed by molecular data sets. J. Theor. Biol. **38**, 423–457 (1973)
10. Robinson, D.F.: Comparison of labeled trees with valency three. J Combinator. Theory Series B **11**, 105–119 (1971)

Negative Cycles in Weighted Digraphs

1994; Kavvadias, Pantziou, Spirakis, Zaroliagis

Christos Zaroliagis
Computer Engineering & Informatics,
University of Patras, Patras, Greece

Problem Definition

Let $G = (V, E)$ be an n-vertex, m-edge directed graph (digraph), whose edges are associated with a real-valued cost function $wt : E \to \mathbb{R}$. The cost, $wt(P)$, of a path P in G is the sum of the costs of the edges of P. A simple path C whose starting and ending vertices coincide is called a cycle. If $wt(C) < 0$, then C is called a *negative cycle*. The goal of the negative cycle problem is to detect whether there is such a cycle in a given digraph G with real-valued edge costs, and if indeed exists to output the cycle.

The negative cycle problem is closely related to the shortest path problem. In the latter, a minimum cost path between two vertices s and t is sought. It is easy to see that an s-t shortest path exists if and only if no s-t path in G contains a negative cycle [1,13]. It is also well-known that shortest paths from a given vertex s to all other vertices form a tree called *shortest path tree* [1,13].

Key Results

For the case of general digraphs, the best algorithm to solve the negative cycle problem (or to compute the shortest path tree, if such a cycle does not exist) is the classical Bellman–Ford algorithm that takes $O(nm)$ time (see e. g., [1]). Alternative methods with the same time complexity are given in [4,7,12,13]. Moreover, in [11, Chap. 7] an extension of the Bellman–Ford algorithm is described which, in addition to detecting and reporting the existing negative cycles (if any), builds a shortest path tree rooted a some vertex s reaching those vertices u whose shortest s-u path does not contain a negative cycle. If edge costs are integers larger than $-L$ ($L \geq 2$), then a better algorithm was given in [6] that runs in $O(m\sqrt{n}\log L)$ time, and it is based on bit scaling.

A simple deterministic algorithm that runs in $O(n^2 \log n)$ expected time with high probability is given in [10] for a large class of input distributions, where the edge costs are chosen randomly according to the endpoint-independent model (this model includes the common case where all edge costs are chosen independently from the same distribution).