5. Busch C, Herlihy M, Wattenhofer R (2000) Hard-potato routing. In: Proceedings of the 32nd annual ACM symposium on theory of computing, Portland, pp 278–285
6. Busch C, Magdon-Ismail M, Mavronicolas M, Spirakis P (2006) Direct routing: algorithms and complexity. Algorithmica 45(1):45–68
7. Cypher R, Meyer auf der Heide F, Scheideler C, Vöcking, B (1996) Universal algorithms for store-and-forward and wormhole routing. In: Proceedings of the 28th ACM symposium on theory of computing, Philadelphia, pp 356–365
8. Feige U, Raghavan P (1992) Exact analysis of hot-potato routing. In: IEEE (ed) Proceedings of the 33rd annual, symposium on foundations of computer science, Pittsburgh, pp 553–562
9. Kaklamanis C, Krizanc D, Rao S (1993) Hot-potato routing on processor arrays. In: Proceedings of the 5th annual ACM, symposium on parallel algorithms and architectures, Velen, pp 273–282
10. Leighton FT (1992) Introduction to parallel algorithms and architectures: arrays – trees – hypercubes. Morgan Kaufmann, San Mateo
11. Leighton FT, Maggs BM, Rao SB (1994) Packet routing and jobscheduling in O(congestion+dilation) steps. Combinatorica 14:167–186
12. Leighton T, Maggs B, Richa AW (1999) Fast algorithms for finding O(congestion + dilation) packet routing schedules. Combinatorica 19:375–401
13. Symvonis A (1996) Routing on trees. Inf Process Lett 57(4):215–223
14. Valiant LG (1982) A scheme for fast parallel communication. SIAM J Comput 11:350–361
15. Valiant LG, Brebner GJ (1981) Universal schemes for parallel communication. In: Proceedings of the 13th annual ACM, symposium on theory of computing, Milwaukee, pp 263–277

Directed Perfect Phylogeny (Binary Characters)

Jesper Jansson
Laboratory of Mathematical Bioinformatics, Institute for Chemical Research, Kyoto University, Gokasho, Uji, Kyoto, Japan

Keywords

Binary character; Character state matrix; Perfect phylogeny; Phylogenetic reconstruction; Phylogenetic tree

Years and Authors of Summarized Original Work

1991; Gusfield
1995; Agarwala, Fernández-Baca, Slutzki
2004; Pe'er, Pupko, Shamir, Sharan

Problem Definition

Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of elements called *objects* and let $C = \{c_1, c_2, \ldots, c_m\}$ be a set of functions from S to $\{0, 1\}$ called *characters*. For each object $s_i \in S$ and character $c_j \in C$, we say that s_i *has* c_j if $c_j(s_i) = 1$ or that s_i *does not have* c_j if $c_j(s_i) = 0$, respectively (in this sense, characters are *binary*). Then the set S and its relation to C can be naturally represented by a matrix M of size $(n \times m)$ satisfying $M[i, j] = c_j(s_i)$ for every $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, m\}$. Such a matrix M is called a *binary character state matrix*.

Next, for each $s_i \in S$, define the set $C_{s_i} = \{c_j \in C : s_i \text{ has } c_j\}$. A *phylogeny for* S is a tree whose leaves are bijectively labeled by S, and a *directed perfect phylogeny for* (S, C) (if one exists) is a rooted phylogeny T for S in which each $c_j \in C$ is associated with exactly one edge of T in such a way that for any $s_i \in S$, the set of all characters associated with the edges on the path in T from the root to leaf s_i is equal to C_{s_i}. See Figs. 1 and 2 for two examples.

Now, define the following problem.

Problem 1 (The Directed Perfect Phylogeny Problem for Binary Characters)

INPUT: An $(n \times m)$-binary character state matrix M for some S and C.

OUTPUT: A directed perfect phylogeny for (S, C), if one exists; otherwise, *null*.

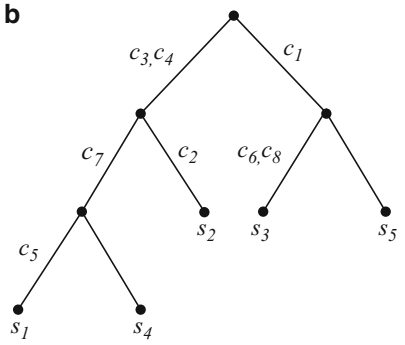
Key Results

In the presentation below, define a set S_{c_j} for each $c_j \in C$ by $S_{c_j} = \{s_i \in S : s_i \text{ has } c_j\}$. The next lemma is the key to solving the Directed

a

M	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
s_1	0	0	1	1	1	0	1	0
s_2	0	1	1	1	0	0	0	0
s_3	1	0	0	0	0	1	0	1
s_4	0	0	1	1	0	0	1	0
s_5	1	0	0	0	0	0	0	0

b



Directed Perfect Phylogeny (Binary Characters), Fig. 1 (**a**) A (5×8)-binary character state matrix M. (**b**) A directed perfect phylogeny for (S, C)

M	c_1	c_2
s_1	1	0
s_2	1	1
s_3	0	1

Directed Perfect Phylogeny (Binary Characters), Fig. 2 This binary character state matrix admits no directed perfect phylogeny

Perfect Phylogeny Problem for Binary Characters efficiently. It is also known in the literature as *the pairwise compatibility theorem* [5].

Lemma 1 *There exists a directed perfect phylogeny for (S, C) if and only if for each pair $c_j, c_k \in C$, it holds that $S_{c_j} \cap S_{c_k} = \emptyset$, $S_{c_j} \subseteq S_{c_k}$, or $S_{c_k} \subseteq S_{c_j}$.*

Short constructive proofs of the lemma can be found in, e.g., [8] and [14]. An algebraic proof of a slightly more general version of the lemma was given earlier by Estabrook, Johnson, and McMorris [3,4].

Using Lemma 1, it is trivial to construct a top-down algorithm for the problem that runs in $O(nm^2)$ time. As one might expect, a faster algorithm is possible. Gusfield [7] observed that after sorting the columns of M in nonincreasing lexicographic order, all duplicate copies of a column appear in a consecutive block of columns and column j is to the right of column k if S_{c_j} is a proper subset of S_{c_k}, and then exploited these two facts together with Lemma 1 to obtain the following result:

Theorem 1 ([7]) *The Directed Perfect Phylogeny Problem for Binary Characters can be solved in $O(nm)$ time.*

For a description of the original algorithm and a proof of its correctness, see [7] or [14]. A conceptually simplified version of the algorithm based on keyword trees can be found in Chapter 17.3.4 in [8]. Gusfield [7] also gave an adversary argument to prove a corresponding lower bound of $\Omega(nm)$ on the running time, showing that his algorithm is time optimal:

Theorem 2 ([7]) *Any algorithm that decides if a given binary character state matrix M admits a directed perfect phylogeny must, in the worst case, examine all entries of M.*

Agarwala, Fernández-Baca, and Slutzki [1] noted that the input binary character state matrix is often sparse, i.e., in general, most of the objects will not have most of the characters. In addition, they noted that for the sparse case, it is more efficient to represent the input (S, C) by all the sets S_{c_j} for $j \in \{1, 2, \ldots, m\}$, where each set S_{c_j} is defined as above and each S_{c_j} is specified as a linked list, than by using a binary character state matrix. Agarwala et al. [1] proved that with this alternative representation of S and C, the algorithm of Gusfield can be modified to run in time proportional to the total number of 1s in the corresponding binary character state matrix:

Theorem 3 ([1]) *The variant of the Directed Perfect Phylogeny Problem for Binary*

Characters in which the input is given as linked lists representing all the sets S_{c_j} *for* $j \in \{1, 2, \ldots, m\}$ *can be solved in* $O(h)$ *time, where* $h = \sum_{j=1}^{m} |S_{c_j}|$.

For a description of the algorithm, refer to [1] or [6]. Observe that Theorem 3 does not contradict Theorem 2; in fact, Gusfield's lower bound argument for proving Theorem 2 considers an input matrix consisting mostly of 1s.

When only a portion of an $(n \times m)$-binary character state matrix is available, an $\tilde{O}(nm)$-time algorithm by Pe'er et al. [13] can fill in the missing entries with 0s and 1s so that the resulting matrix admits a directed perfect phylogeny, if possible. A ZDD-based algorithm for enumerating all such solutions was recently developed by Kiyomi et al. [11].

Theorem 4 ([13]) *The variant of the Directed Perfect Phylogeny Problem for Binary Characters in which the input consists of an incomplete binary character state matrix can be solved in* $\tilde{O}(nm)$ *time.*

Applications

Directed perfect phylogenies for binary characters are used to describe the evolutionary history for a set of objects (e.g., biological species) that share some observable traits and that have evolved from a "blank" ancestral object which has none of the traits. Intuitively, the root of a directed perfect phylogeny corresponds to the blank ancestral object, and each directed edge $e = (u, v)$ corresponds to an evolutionary event in which the hypothesized ancestor represented by u gains the characters associated with e, transforming it into the hypothesized ancestor or object represented by v. For simplicity, it may be assumed that each character can emerge once only during the evolutionary history and is never lost after it has been gained, so that a leaf s_i is a descendant of the edge associated with a character c_j if and only if s_i has c_j. When this requirement is too strict, one can relax it to permit errors, for example, by letting each character be associated with more than one edge in the phylogeny (i.e., allow each character to emerge many times) while minimizing the total number of such associations (*Camin-Sokal optimization*) or by keeping the requirement that each character emerges only once but allowing it to be lost multiple times (*Dollo parsimony*) [5, 6]. Such relaxations generally increase the computational complexity of the underlying computational problems; see, e.g., [2] and [15].

Binary characters are commonly used by biologists and linguists. Traditionally, morphological traits or directly observable features of species were employed by biologists as binary characters, and recently, binary characters based on genomic information such as substrings in DNA or protein sequences, SNP markers, protein regulation data, and shared gaps in a given multiple alignment have become more and more prevalent. Chapter 17.3.2 in [8] mentions several examples where phylogenetic trees have been successfully constructed based on such types of binary character data. In the context of reconstructing the evolutionary history of natural languages, linguists often use phonological and morphological characters with just two states [10].

The Directed Perfect Phylogeny Problem for Binary Characters is closely related to *the Perfect Phylogeny Problem*, a fundamental problem in computational evolutionary biology and phylogenetic reconstruction [5, 6, 14]. This problem (also described in more detail in Encyclopedia entry ▸ Perfect Phylogeny (Bounded Number of States)) introduces nonbinary characters so that each character $c_j \in C$ has a set of allowed states $\{0, 1, \ldots, r_j - 1\}$ for some integer r_j, and for each $s_i \in S$, character c_j is in one of its allowed states. Generalizing the notation used above, define the set $S_{c_j,\alpha}$ for every $\alpha \in \{0, 1, \ldots, r_j - 1\}$ by $S_{c_j,\alpha} = \{s_i \in S :$ the state of s_i on c_j is $\alpha\}$. Then, the objective of *the Perfect Phylogeny Problem* is to construct (if possible) an *unrooted* phylogeny T for S such that the following holds: for each $c_j \in C$ and distinct states α, β of c_j, the minimal subtree of T that connects $S_{c_j,\alpha}$ and the minimal subtree of T that connects $S_{c_j,\beta}$ are vertex-disjoint. McMorris [12] showed that the special case with $r_j = 2$ for all $c_j \in C$ can be reduced to the

Directed Perfect Phylogeny Problem for Binary Characters in $O(nm)$ time: for each $c_j \in C$, if the number of 1s in column j of M is greater than the number of 0s, then set entry $M[i, j]$ to $1 - M[i, j]$ for all $i \in \{1, 2, \ldots, n\}$. Therefore, another application of Gusfield's algorithm [7] is as a subroutine for solving the Perfect Phylogeny Problem in $O(nm)$ time when $r_j = 2$ for all $c_j \in C$. Even more generally, the Perfect Phylogeny Problem for directed as well as undirected *cladistic* characters can be solved in polynomial time by a similar reduction to the Directed Perfect Phylogeny Problem for Binary Characters (see [6]).

In addition to the above, it is possible to apply Gusfield's algorithm to determine whether two given trees describe compatible evolutionary history, and if so, merge them into a single tree so that no branching information is lost (see [7] for details). Finally, Gusfield's algorithm has also been used by Hanisch, Zimmer, and Lengauer [9] to implement a particular operation on documents defined in their Protein Markup Language (ProML) specification.

Cross-References

- ▸ Directed Perfect Phylogeny (Binary Characters)
- ▸ Perfect Phylogeny Haplotyping

Acknowledgments JJ was funded by the Hakubi Project at Kyoto University and KAKENHI grant number 26330014.

Recommended Reading

1. Agarwala R, Fernández-Baca D, Slutzki G (1995) Fast algorithms for inferring evolutionary trees. J Comput Biol 2(3):397–407
2. Bonizzoni P, Braghin C, Dondi R, Trucco G (2012) The binary perfect phylogeny with persistent characters. Theor Comput Sci 454:51–63
3. Estabrook GF, Johnson CS Jr, McMorris FR (1976) An algebraic analysis of cladistic characters. Discret Math 16(2):141–147
4. Estabrook GF, Johnson CS Jr, McMorris FR (1976) A mathematical foundation for the analysis of cladistic character compatibility. Math Biosci 29(1–2): 181–187
5. Felsenstein J (2004) Inferring phylogenies. Sinauer Associates, Sunderland
6. Fernández-Baca D (2001) The perfect phylogeny problem. In: Cheng X, Du DZ (eds) Steiner trees in industry. Kluwer Academic, Dordrecht, pp 203–234
7. Gusfield DM (1991) Efficient algorithms for inferring evolutionary trees. Networks 21:19–28
8. Gusfield DM (1997) Algorithms on strings, trees, and sequences. Cambridge University Press, New York
9. Hanisch D, Zimmer R, Lengauer T (2002) ProML – the protein markup language for specification of protein sequences, structures and families. In Silico Biol 2:0029. http://www.bioinfo.de/isb/2002/02/0029/
10. Kanj IA, Nakhleh L, Xia G (2006) Reconstructing evolution of natural languages: complexity and parameterized algorithms. In: Proceedings of the 12th annual international computing and combinatorics conference (COCOON 2006). Lecture notes in computer science, vol 4112. Springer, Berlin/Heidelberg, pp 299–308
11. Kiyomi M, Okamoto Y, Saitoh T (2012) Efficient enumeration of the directed binary perfect phylogenies from incomplete data. In: Proceedings of the 11th international symposium on experimental algorithms (SEA 2012). Lecture notes in computer science, vol 7276. Springer, Berlin/Heidelberg, pp 248–259
12. McMorris FR (1977) On the compatibility of binary qualitative taxonomic characters. Bull Math Biol 39(2):133–138
13. Pe'er I, Pupko T, Shamir R, Sharan R (2004) Incomplete directed perfect phylogeny. SIAM J Comput 33(3):590–607
14. Setubal JC, Meidanis J (1997) Introduction to computational molecular biology. PWS Publishing Company, Boston
15. Sridhar S, Dhamdhere K, Blelloch GE, Halperin E, Ravi R, Schwartz R (2007) Algorithms for efficient near-perfect phylogenetic tree reconstruction in theory and practice. IEEE/ACM Trans Comput Biol Bioinform 4(4):561–571

Discrete Ricci Flow for Geometric Routing

Jie Gao[1], Xianfeng David Gu[1], and Feng Luo[2]
[1]Department of Computer Science, Stony Brook University, Stony Brook, NY, USA
[2]Department of Mathematics, Rutgers University, Piscataway, NJ, USA

Keywords

Geometric routing; Greedy routing; Greedy embedding; Virtual coordinates; Wireless networks