木編集距離の宣言的意味に基づく階層とその計算に関する研究

著者	芳野　拓也
学位授与年度	平成29年度
学位授与番号	17104甲情工第332号
URL	http://hdl.handle.net/10228/00006825

木編集距離の宣言的意味に基づく階層とその計算に関する研究

芳野 拓也

概要

Web における HTML データや XML データ, バイオインフォマティクスにおける RNA や糖鎖データのような根付きラベル付き木 (以後, 木という) として表現される木構造データを比較することは, 構造データからのデータマイニングや機械学習における重要な研究の一つである. そのような木同士の距離として有名なものの一つに木編集距離がある. 木編集距離は, ノードの削除, 挿入, 置換からなる編集操作を用いて, 一方の根付き木から他方の木への変換に必要な編集操作列の最小コストとして定式化される.

2 つの木の間の編集操作列は無数に存在するため, 操作列をすべて計算して木編集距離を求める方法は現実的ではない. そこで Tai は, 木編集距離計算の指針として, 木編集距離に宣言的意味を与える Tai マッピング (以後単にマッピングともいう) を導入した. この Tai マッピングは, 先祖子孫関係 (および順序木の場合は兄弟関係) を保持する木のノード間の一対一対応であり, Tai マッピングの最小コストは木編集距離と一致する.

木編集距離の計算時間は, 順序木の場合はノード数 n に対して $O(n^3)$ 時間であるが, 無順序木の場合は MAX SNP 困難である. 一方, 糖鎖データではノードのつながりに意味があるためそのつながりを崩さないような制約が求められ, XML データでは根ノードから一定のノードはどの木にも共通する場合があり, より葉ノードに重点を置いた距離が求められる. このように, 対象によっては木編集距離は過度に一般的となるため, 他方では計算効率を上げるという目的の下に, 宣言的意味であるマッピングに制限を加えることで木編集距離のさまざまな変種が研究されている.

特に, RNA 解析などで利用され, 削除の前に挿入を行う木編集距離でもある木アライメ

ント距離の計算は, 順序木の場合はノード数 n に対して $O(n^4)$ 時間, 無順序木の場合は一般に MAX SNP 困難であるが, 次数が限定されている木のときは多項式時間で計算できる. このアライメント距離は, 2 つの木の超木となるアライメント木の最小コストとして定式化することができ, Tai マッピングに制限を加えた劣制限マッピングの最小コストと一致する.

本論文では, まず, マッピングへの制限を Tai マッピングの階層として捉え, この階層を共通部分森, 特に, 共通部分森中のノードの接続と部分木の並びの観点から見直すことで, 木編集距離の変種の計算における本質について研究する. また, これらの観点によって新たに導入されるマッピングについて, それらの最小コストとなる編集距離の変種の時間計算時間を解析する.

また, 木アライメント距離に対して, 森アライメント構築の高速化を目的として導入されたアンカーアライメント問題が提唱されている. これは, アンカーと呼ばれるマッピングを入力とし, そのアンカーでの対応を保持したアライメント木を構築する問題であるが, このアンカーは Tai マッピングであり, 劣制限マッピングでないマッピングがアンカーとして入力されると木が構築することができない. そこで本論文では, 木アライメント距離の宣言的意味が劣制限マッピングとなることの構成的な別証明を与え, その構成方法を利用することで, アンカーアライメント問題の出力を, アライメント木が構築できない場合は”no”を返す形に定式化する. また, それに基づくアンカーアライメント距離を定式化し, アンカーアライメント距離とアライメント距離を実データをもとに比較する. さらに, 順序木より一般的であり, 無順序木より制限された巡回的順序木を提案し, 巡回的順序木間でのアライメント距離を計算するアルゴリズムを設計する.

最後に, 木編集距離に関するさまざまな内容として, 無順序木編集距離を計算する動的 A* アルゴリズムの設計, Tai マッピングの根無し木への拡張, 巡回的順序木と次数制限無順序木のマッピングカーネルの設計を行う.

無順序木編集距離を計算するアルゴリズムとしては, 既に, 複数の下限関数を用いるHiguchi らの A* アルゴリズムが導入されているが, これには計算の重複が存在するため, 改善の余地がある. 本論文では, その重複計算を動的計画法を用いて省いた動的 A* アルゴリズムを導入する. また, 実験により, 下限関数の効率を確認する.

また, 根付き木 Tai マッピングは木編集距離に対応する重要な概念であるが, この Tai マッピングを根無し木に拡張するためには, 単射であることに加えて, 先祖子孫関係に代わる条件を導入する必要がある. そこで, Zhang らが LCA 保存マッピングを根無し木に拡張する際に用いた中心に着目し, 根無し木のマッピングを導入する. 特に, 根無し木としてよく表現される進化系統樹を特徴づける条件である 4 点条件と 3 点条件を木のトポロジーを特徴づける条件に変更し, それぞれの条件を保存するようなマッピングを導入する.

さらに, サポートベクターマシンを利用して木を分類するための基本的な方法の 1 つである木カーネルは順序木について多く研究がおこなわれており, そのほとんどが, 順序木間のマッピングを数え上げるマッピングカーネルのフレームワークに分類される. 一方で, 無順序木のカーネルは, その計算の難しさからほとんど研究がなされていない. そこで, 巡回的順序木と, 次数を定数 D に制限した無順序木に対するマッピングカーネルを設計し, それらの計算時間について議論する.

目 次

第1章　はじめに

1.1　木編集距離と木アライメント距離

Web マイニングに対する HTML, XML データや, バイオインフォマティクスに対する RNA 二次構造, 糖鎖データのような, 木構造データの比較はデータマイニングにおける重要なタスクの 1 つである. ここで, 木構造としては根を持ちすべてのノードにラベルが付与された**根付きラベル付き木** (*rooted labeled tree*)(以後, 単に木という) を対象とする. 特に, 子供の順番を考慮する木を**順序木** (*ordered tree*), 考慮しない木を**無順序木** (*unordered tree*) という.

そのような木同士の距離として最も有名なものの 1 つとして, **木編集距離** (*tree edit distance*) [1, 4, 29, 45] が挙げられる. 木編集距離は文字列編集距離の拡張であり, ノードの**削除** (*deletion*), **挿入** (*insertion*), **置換** (*substitution*) からなる**編集操作** (*edit operation*) を用いて, 一方の木から他方の木への変換に必要な編集操作列の最小コストとして定式化される.

2 つの木の間の編集操作列は無数に存在するため, 操作列をすべて計算することで編集距離を求める方法は現実的ではない. そこで, Tai [45] は編集距離計算の指針として, 編集距離の宣言的意味を与える **Tai マッピング** (*Tai mapping*) を導入した. この Tai マッピングは, 先祖子孫関係 (順序木の場合は加えて兄弟関係) を保存する木のノード間の一対一対応であり, 編集距離と密接に関連している [4, 29, 45]. Tai マッピングに対して, 一方の木に含まれる未対応のノードは編集操作の削除に, 他方の木に含まれる未対応のノードは編

集操作の挿入に, 異なるラベルを持つノードの対応は編集操作の置換に対応する. また, すべての可能な Tai マッピングの最小コストは, 編集距離と一致する [45].

順序木についての木編集距離の計算は, 1979 年に Tai [45] が $O(n^6)$ 時間アルゴリズムを設計してから, 1989 年に Zhang と Shasha [62] の $O(n^4)$ 時間, 1998 年に Klein [28] の $O(n^3 \log n)$ 時間, 2007 年に Demaine ら [8] の (n^3) 時間と改良されてきた. ここで, n は木の最大ノード数である. 加えて, Demaine ら [8] は, 彼らのアルゴリズムを使う限り, 木編集距離の計算には $\Omega(n^3)$ 時間が必要となる木が存在することも示した. さらに, Bringmann ら [5] は, 任意の $\varepsilon > 0$ に対して, APSP(All Pairs Shortest Paths) が $O(n^{3-\varepsilon})$ 時間で解けなければ, 木編集距離も $O(n^{3-\varepsilon})$ 時間で解けないことを示した. すなわち, APSP の仮定の下で, 木編集距離の計算における $O(n^3)$ 時間は最適であると言える.

一方, 無順序木についての木編集距離の計算は, NP 困難であり [63, 64], MAX SNP 困難である [2, 18, 61] ことが知られている. この MAX SNP 困難性は次数が 2 のとき, および高さが 2 のときも成り立つ [18, 61].

また編集距離と同様に, 特に RNA 二次構造比較において, 文字列アライメント距離の拡張である**木アライメント距離** (*tree alignment distance*) もよく知られている [21, 37]. これは, 2 つの木それぞれに, ラベルを考慮しない状態で同型になるまで空文字を持つノードを挿入し, 出来上がった 2 つの木を重ね合わせることでできるアライメント木の最小コストとして定式化される. 文字列アライメント距離は文字列編集距離と一致することが知られているが, 一般に, 木アライメント距離は木編集距離以上の値を取る. また, 木アライメント距離は, 編集操作の観点からは, すべての挿入が削除の前に行われる編集距離となる [21].

アライメント距離の計算は, 順序木では, 木の最大ノード数 n と木の最大次数 D に対して $O(n^2D^2)$ 時間であり, 無順序木では, MAX SNP 困難である [21]. ただし, 木の次数を定数以下に制限すれば, 多項式時間で計算が可能となる.

表 1.1 に, 木編集距離と木アライメント距離計算の時間計算量をまとめる.

表 1.1: 木編集距離と木アライメント距離の計算時間. ここで, n はノード数の最大値, D は次数の最大値である.

距離	順序木	無順序木	
		一般	D が定数
木編集距離	$O(n^6)$ 時間 [45] $O(n^4)$ 時間 [62] $O(n^3 \log n)$ 時間 [28] $O(n^3)$ 時間 [8]	MAX SNP 困難 [61]	MAX SNP 困難 [18]
木アライメント距離	$O(n^2D^2)$ 時間 [21]	MAX SNP 困難 [21]	$O(n^2D^2D!)$ 時間 [21]

1.2　Tai マッピングと木編集距離の変種

編集距離は木を比較するための標準的な尺度であるが, 下に述べるようにいくつかの適用範囲によっては一般的過ぎる場合がある. そのため, それらの適用範囲には, より構造的特徴を踏まえた編集距離が必要となる. このような距離は, Tai マッピングに制約を加えた変種を考え, その変種の最小コストとして定式化することができる.

1. Selkow [38] は, 削除や挿入を部分木単位で行う編集距離である**トップダウン距離** (*top-down mapping*)(Top) を導入した. Yang [52] は, 開発途中のプログラムの差異を発見するという問題に対して, テキストの差分を出力するだけでは不十分と考え, プログラムの構文に沿った比較指標として, このトップダウン距離を用いた. また, Wang ら [50] は, XML 文書の更新検出を高速に行うための X-Diff を導入しているが, これはトップダウン距離と同等のものである.

2. Lu [33] は, 誤差変換を用いた木間の距離として木編集距離と同様の距離を計算するアルゴリズムを導入し, 文字画像の分類に用いた. しかし, そのアルゴリズムは実際には木編集距離を求めるものではなかった. Kuboyama [29] は, この距離を特徴づけ

る Tai マッピングの変種を導入し, この距離を**調和的距離** (*accordant distance*)(あるいは **Lu 距離** (*Lu's distance*))(Acc) として定式化した.

3. Zhang [59] は, RNA2 次構造の位相的形状を木とみなし, **制限距離** (*constrained distance*) (あるいは**孤立部分木距離** (*isolated-subtree mapping*) [49])(Ilst) を用いた距離を用いたクラスタリングによって, 編集距離とおおよそ同等のクラスタが得られることを実験的に示した. また, Ferraro と Godin [10] は, 植物の位相が木になることを用いて, 植物間の距離に孤立部分木距離を用いている.

4. Zhang ら [64] は, 生化学情報システムの一般的な用途である, 化合物データベースにおける類似化合物の検索において, それまでによく用いられてきた部分構造の包含数による類似度と比較してもより視覚的な類似度を求めることができる **LCA 保存距離** (*LCA-preserving distance*)(あるいは**次数 2 距離** (*degree-2* **距離**))(Lca) を導入した. ここで LCA とは, **最近共通先祖** (*least common ancestor*) のことである.

5. Jiang ら [21] は, RNA2 次構造を木構造で表すことができることに着目し, RNA1 次構造で用いられていた文字列アライメント距離を木に拡張した, **アライメント距離** (*alignment distance*)(Aln) を導入した. その中で, アライメント距離は最小の**超木** (*supertree*) を与えることに触れている.

6. Lu ら [32] は, Zhang [59] の導入した制限距離の条件をより緩和した編集距離の変種として, **劣制限距離** (*less-constrained distance*)(Less) を導入した.

7. Valiente [47] は, 木パターンマッチングにおける同型な完全部分木のマッチング問題に対して, 木の最大共通部分森に基づく距離として, **ボトムアップマッピング** (*bottom-up mapping*)(Bot) を導入した. このボトムアップマッピングは, Hamou-Lhadj と Lethbridge [15] によって, システムの実行時解析を行う際の手続き呼び出しパターン検出のために用いられた.

8. Kan ら [22] は, 糖鎖構造の木表現ではノードの接続関係が糖鎖の機能を決定することから, 接続関係に基づいた距離である**断片距離** (*segmental distance*)(Sg) を導入した.

編集操作としては, LCA 保存距離は, 葉か次数 1 のノードだけに削除, 挿入を許す距離であり, トップダウン距離は, 葉だけに削除, 挿入を許す距離となる. また, ボトムアップ距離は, 共通部分森以外を削除, 挿入する距離であり, アライメント距離は, すべての挿入を終えた後に削除を行うような距離である. 表 1.2 は, 上述した距離の適用範囲である.

表 1.2: 編集距離の変種の木への適用範囲

編集距離の変種	適用範囲
アライメント距離	RNA2 次構造, 最小超木 [21]
孤立部分木距離	RNA2 次構造 [59], 植物の構造 [10]
調和的距離	文字画像の分類 [33]
LCA 保存距離	化合物の類似検索 [64]
トップダウン距離	プログラムの解析木 [7, 52], XML 文書 [50]
断片距離	糖鎖 [22]
ボトムアップ距離	手続き呼び出しのパターン追跡 [15]

これらの距離を特徴づけるマッピングは, 図 1.3 に示された **Tai マッピング階層** (*Tai mapping hierarchy*) を与える. これまでに, 新たなマッピングを追加することによって, 階層は左から右へと拡張されてきている. ここで, Iso は同型写像を指し, “Less=Aln” [29], “Top=TopSg” [22] となる. この図では, 上位のマッピングクラス A と下位のマッピングクラス B に対して, マッピング M が B に属するならば, M は A にも属する.

1.3　本論文の構成

本論文では次の内容について研究を進めている.

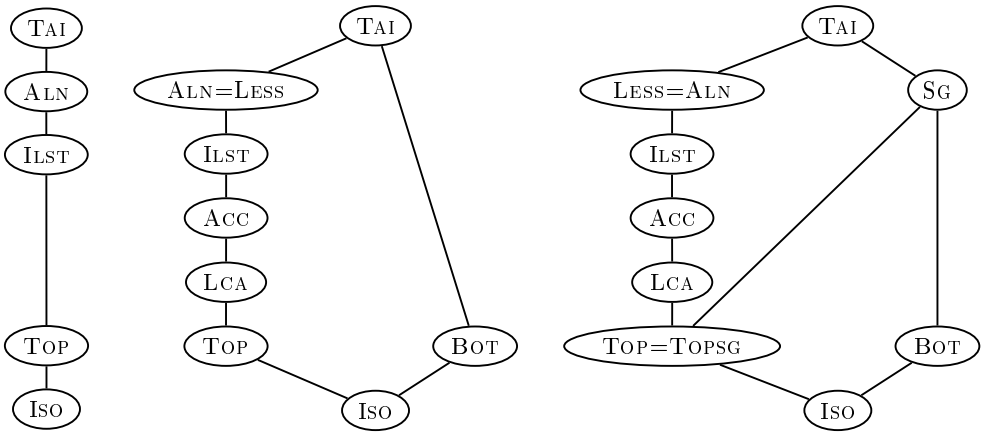


図 1.3: それぞれ Wang と Zhang [49] (左), Kuboyama [29] (中央), Kan ら [22] (右) によって導入された Tai マッピング階層.

1. 共通部分森に基づく Tai マッピング階層. 階層内のマッピングから定義される木編集距離の変種とその時間計算量の解析.

2. 木アライメント距離の計算. 特に, アンカーアライメント問題とアンカーアライメント距離, および, 巡回的順序木の木アライメント距離.

3. さまざまな拡張. 無順序木編集距離計算の A*アルゴリズム, 編集距離の根無し木への拡張, 巡回的順序木と字数制限無順序木のマッピングカーネル.

1.3.1　共通部分森に基づく Tai マッピング階層

3 章では, 新たな Tai マッピングの階層を導入する. また, それに対応した木編集距離の変種を導入し, それらの計算時間を解析する.

図 1.3 の Tai マッピング階層はマッピングの包含関係を示しているが, マッピングの他の特徴は示していない. 特に, 図 1.3 (右) の右の列は左の列よりも疎である. そこで, マッピングの変種間に統一的な性質が存在するかどうかといった疑問が生じる.

本論文では, この疑問を解決するために, Tai マッピングの階層を, 2 つの木のノードの組からなる**共通部分森** (*common subforest*) で特徴づける. このとき, 共通部分森中の**ノー**

ドの接続と**部分木の並び**に着目する．ここで, 最大共通部分森 (あるいは木) は, 別の木の類似度を与えることができる [2, 18, 29, 48, 61][1].

まず, 共通部分森中のノードの接続に着目する．**共通埋め込み部分森** (*common embedded subforest*) は, 先祖子孫関係に従ってマッピングのノードを接続している共通部分森である．**共通誘導部分森** (*common induced subforest*) は, あるノードの集合に誘導される共通部分森である．**共通完全部分森** (*common complete subforest*) は, あるノードの集合と, その子孫すべてを含む共通部分森である．これらを用いて, Tai, Aln, Ilst, Acc, Lca に含まれるマッピングは共通埋め込み部分森, Sg, Top に含まれるマッピングは共通誘導部分森, Bot に含まれるマッピングは共通完全部分森としてそれぞれ特徴づけることができる．共通誘導部分森によって特徴づけられるマッピングは, 削除および挿入を根か葉にのみ適用できるといった操作的条件を必要とし, 共通完全部分森によって特徴づけられるマッピングは, 削除および挿入を根のみに適用できるといった操作的条件を必要とする．

次に, 共通部分森中の部分木の並びに着目する．一方の木で 1 つ目と 2 つ目のノードの最近共通先祖 (LCA) が 2 つ目と 3 つ目のノードの LCA の先祖のとき, およびそのときに限り, 他方の木で 2 つ目と 3 つ目のノードの LCA が 1 つ目と 2 つ目のノードの LCA の先祖であるような共通部分森中の 3 つのノードが存在しないとき, その共通部分森を**ねじれが無い** (*non-twisting*) という．共通部分森中の任意の 3 つのノードに対して, 一方の木で 1 つ目と 2 つ目のノードの LCA が 1 つ目と 3 つ目のノードの LCA であるとき, およびそのときに限り, 他方の木で 1 つ目と 2 つ目のノードの LCA が 1 つ目と 3 つ目のノードの LCA であるとき, その共通部分森を**並列である** (*parallel*) という．共通部分森が木であれば, その共通部分森を**部分木** (*subtree*) という．共通部分木の根が 2 つの木の根であるとき, その共通部分木を**根保存である** (*root-preserving*) という．これらを用いて, Tai, Sg, Bot に含まれるマッピングは特に制約のない共通部分森, Aln に含まれるマッピングはねじれ

[1] 標準的な定義 [2, 18, 29, 48, 61] における**最大共通部分木** (*largest common subtree*) は, 同じラベルを持つノードの組からなる **LCA 保存断片マッピング** (後に LcaSg と定義する) の最小コストと対応する．

のない共通部分森, Ilst, Acc に含まれるマッピングは並列な共通部分森, Lca に含まれるマッピングは共通部分木, Top, Iso に含まれるマッピングは根保存共通部分木として, それぞれ特徴付けることができる.

その結果, 図 1.3 の Tai マッピング階層は, 共通部分森中のノードの接続と部分木の並びに対して不完全であることが分かる. この隙間を埋めるため, 本論文では, 図 1.3 の右列にある Sg, Bot と, 図 1.3 の左側にある Aln, Ilst, Acc, Lca の共通部分をとることで, 新たなマッピングを導入する.

Sg との共通部分として, **断片アライメント可能マッピング** (*segmental alignable mapping*) (SgAln), **孤立部分木断片マッピング** (*isolated-subtree segmental mapping*) (IlstSg), **調和的断片マッピング** (*accordant segmental mapping*) (AccSg), **LCA 保存断片マッピング** (*LCA-preserving segmental mapping*) (LcaSg) を, Bot との共通部分として, **ボトムアップアライメント可能マッピング** (*bottom-up alignable mapping*) (BotAln), **孤立部分木ボトムアップマッピング** (*isolated-subtree bottom-up mapping*) (IlstBot), **調和的ボトムアップマッピング** (*accordant bottom-up mapping*) (AccBot), **LCA 保存ボトムアップマッピング** (*LCA-preserving bottom-up mapping*) (LcaBot) をそれぞれ導入する. さらに, **LCA 保存根保存マッピング** (*LCA- and root-preserving mapping*) (LcaRt) を導入する.

これらのマッピングの導入により, 図 1.4 に示す新たな Tai マッピング階層を得ることができる. ここで, 図 1.3 ですでに導入されていたマッピングは灰色の線で, 本論文で導入したマッピングは黒色の線で囲んである. 特に, “AccSg=IlstSg”, “AccBot=IlstBot” である.

図 1.4 の縦方向について, 左の列 (Tai, Aln, Ilst, Acc, Lca, LcaRt), 中央列 (Sg, SgAln, AccSg, LcaSg, Top), 右の列 (Bot, BotAln, AccBot, LcaBot, Iso) のマッピングは, それぞれ, 共通埋め込み部分森, 共通誘導部分森, 共通完全部分森として特

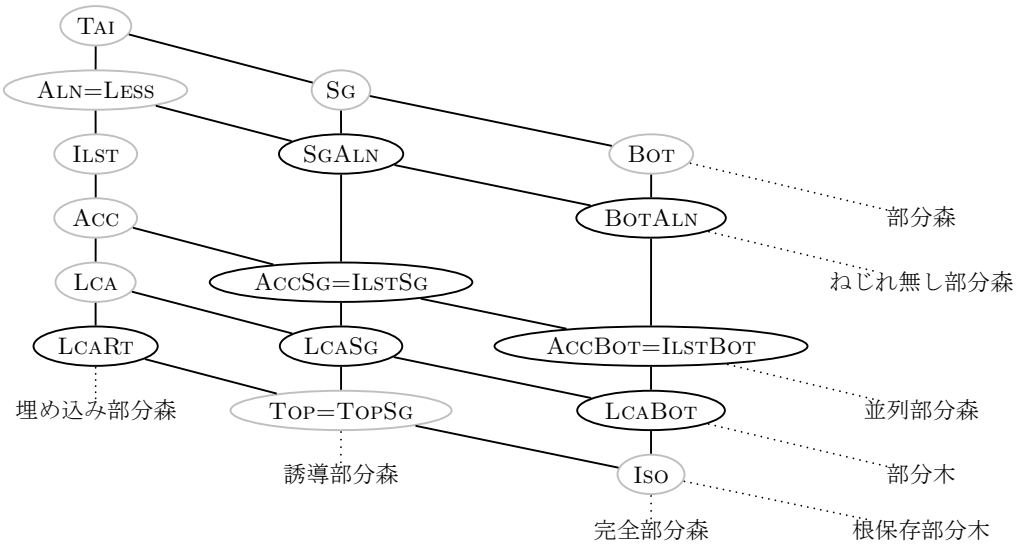


図 1.4: 図 1.3 のマッピング (灰色の線) と新たに導入したマッピング (黒色の線) による新たな Tai マッピング階層.

徴付けることができる. 一方, 図 1.4 の横方向について, 上の行 (Tai, Sg, Bot), 2 番目の行 (Aln, SgAln, BotAln), 3 番目の行 (Ilst, Acc, AccSg, AccBot), 4 番目の行 (Lca, LcaSg, LcaBot), 最後の行 (LcaRt, Top and Iso) のマッピングは, それぞれ, 制約無し部分森, ねじれ無し部分森, 並列部分森, 部分木, 根保存部分として特徴付けることができる.

次に, SgAln, AccSg, LcaSg, BotAln, AccBot, LcaBot, LcaRt に対して, すべての可能なマッピングの最小値として編集距離の変種を導入する. これらをそれぞれ, **断片アライメント距離** (*segmental alignment distance*), **調和的断片距離** (*accordant segmental distance*), **LCA 保存断片距離** (*LCA-preserving segmental distance*), **ボトムアップアライメント距離** (*bottom-up alignment distance*), **調和的ボトムアップ距離** (*accordant bottom-up distance*), **LCA 保存ボトムアップ距離** (*LCA-preserving bottom-up distance*), **LCA 保存根保存距離** (*LCA- and root-preserving distance*) という. このとき, ねじれ無し部分森によって特徴付けられる距離はメトリックではなく, そのほかの距離はメトリックであることを示す.

さらに, 編集距離の変種の計算に関する時間計算量を解析する. ここで, m, n はそれぞれ 2 つの木のノード数 ($m \leq n$), D は 2 つの木の最大次数, d は 2 つの木の最小次数を表す.

順序木に対しては, ねじれ無し部分森に特徴付けられる距離の時間計算量は $O(nmD^2)$ であり, 通常の編集距離や木同型を除く, 他の部分森や部分木に特徴付けられる距離の時間計算量は $O(nm)$ である. 無順序木に対しては, 制約無し部分森に特徴付けられる距離の計算問題は MAX SNP 困難であり [61], これは二分木であったとしても成り立つ [18, 51]. 一方, アライメント距離 [21] の場合と同様にして, ねじれ無し部分森に特徴付けられる距離の計算問題は MAX SNP 困難であるが, 次数がある定数以下の場合, 多項式時間での計算が可能になる. さらに, 並列部分森, 部分木, 根保存部分木に特徴付けられる距離の計算は $O(nmd)$ 時間で可能である. ただし, LCA 保存ボトムアップ距離は $O(nm)$ 時間で計算が可能である. これらをまとめると, 表 1.5 のようになる.

表 1.5: マッピング $\mathcal{M}_{\mathtt{A}}(T_1, T_2)$ に対する距離 $\tau_{\mathtt{A}}(T_1, T_2)$ のメトリック性と, その計算の時間計算量. ここで, $n = \max\{|T_1|, |T_2|\}$, $m = \min\{|T_1|, |T_2|\}$, $D = \max\{d(T_1), d(T_2)\}$, $d = \min\{d(T_1), d(T_2)\}$ である.

マッピング $\mathcal{M}_{\mathtt{A}}(T_1, T_2)$	メトリック	順序木 $\tau^o_{\mathtt{A}}$		無順序木 $\tau^u_{\mathtt{A}}$		(次数を制限)	
Tai	yes	$O(nm^2(1+\log\frac{n}{m}))$	[8]	MAX SNP	[61]	MAX SNP	[18]
Aln	no	$O(nmD^2)$	[21]	MAX SNP	[21]	多項式時間	[21]
Ilst	yes	$O(nm)$	[59]	$O(nmd)$	[51]	$O(nm)$	←
Acc	yes	$O(nm)$	[29]	$O(nmd)$	[51]	$O(nm)$	←
Lca	yes	$O(nm)$	[64]	$O(nmd)$	[64]	$O(nm)$	←
LcaRt	yes	$O(nm)$	定理 3.15	$O(nmd)$	定理 3.15	$O(nm)$	←
Sg	yes	$O(nm)$	[22]	MAX SNP	[51]	MAX SNP	[51]
SgAln	no	$O(nmD^2)$	定理 3.17	MAX SNP	定理 3.26	多項式時間	定理 3.27
AccSg	yes	$O(mn)$	定理 3.17	$O(nmd)$	定理 3.18	$O(nm)$	←
LcaSg	yes	$O(nm)$	定理 3.16	$O(nmd)$	定理 3.16	$O(nm)$	←
Top	yes	$O(nm)$	[7, 38]	$O(nmd)$	[51]	$O(nm)$	←
Bot	yes	$O(nm)$	[51]	MAX SNP	[51]	MAX SNP	[51]
BotAln	no	$O(nmD^2)$	定理 3.21	MAX SNP	定理 3.26	多項式時間	定理 3.27
AccBot	yes	$O(nm)$	定理 3.21	$O(nmd)$	定理 3.22	$O(nm)$	←
LcaBot	yes	$O(nm)$	定理 3.20	$O(nm)$	定理 3.20	←	
Iso	yes	$O(n+m)$	[48] 参照	$O(n+m)$	[48] 参照	←	

1.3.2　木アライメント距離の計算

4 章では, アンカーアライメント問題を定式化し, アンカーアライメント距離を導入する. また, 順序木よりも一般的かつ無順序木よりも制限の強い巡回的順序木を導入し, 巡回的順序木間のアライメント距離を求めるアルゴリズムを設計する.

無順序木において, 編集距離の変種であるアライメント距離の計算問題は MAX SNP 困難であるが, 2 つの木の最大次数が定数 D 以下の場合 $O(n^2D^2D!)$ 時間で計算可能である [21]. ここで, n は 2 つの木の最大ノード数である.

一方, Schiermer と Giegerich [37] は, Tai マッピングを**アンカー** (*anchoring*) として用いる**アンカーアライメント** (*anchored alignment*) を導入した. ただし, このアンカーは劣制限となる保証はなく, したがって, アライメント距離が計算できない場合もある. そこで本論文では, Kuboyama [29] が示した "LESS=ALN" の構成的証明を与えることで, その構成方法に基づいてアンカー M が劣制限でないならば "no" を返し, 劣制限ならばアンカーアライメントを $O(H|M|^2 + n)$ 時間で計算するアルゴリズムを設計する. ここで, H は 2 つの木の最大の高さである.

このアンカーアライメントを計算するためには, アンカーを与える必要がある. 次に本論文では, Tai マッピングの階層で劣制限マッピングに最も近く, 最小コストを $O(n^2d)$ 時間で計算可能な孤立部分木マッピング [51] からアンカーを構築する. ここで, d は 2 つの木の次数の最小値である.

最小コストの孤立部分木マッピング M に対して, M で対応付けられていない葉の対の集合 M' を選ぶ. そして, **アンカーアライメント距離** (*anchored alignment distance*) τ_{ACH} を, アンカー $M \cup M'$ が劣制限ならば最小コストの $M \cup M'$ によるアンカーアライメントのコストとして, そうでなければ孤立部分木距離 τ_{ILST} として定式化する. そして, τ_{ACH} を $O(n^2(d + H2^v))$ 時間で計算するアルゴリズムを設計する. ここで, v は 2 つの木の葉の数の最小値である.

このアルゴリズムの実行には, 理論上 v に依存した指数時間かかるが, KEGG [26] の提供する N 型糖鎖データに対しては, アンカーアライメント距離 τ_{ACH} を効果的に計算できることを実験的に確認する. また, アンカーアライメント距離 τ_{ACH}, アライメント距離 τ_{ALN}, 孤立部分木距離 τ_{ILST} を実例をもとに比較する.

上で述べた通り, 無順序木アライメント距離は次数を定数以下に制限することで計算可能になる. 無順序木は兄弟間のすべての並びを許す一方, 応用例によっては兄弟間の並びすべてを必要としない場合もある. 例えば, 化学成分のようなサイクルを持つグラフを木で表現するとき, サイクル内でのノードの隣接関係だけを木表現で保持すればよい. そこで特定の並びのみを許容する木として, 次の 3 つの**巡回的順序木** (*cyclically ordered tree*) を導入し, そのアライメント距離について考察する.

$v_1, \ldots, v_n$ を, 兄弟を左から右に並べたノードの列とする. このとき, $v_1, \ldots, v_n$ と $v_n, \ldots, v_1$ の 2 つの並びを許容する木を**両順序木** (*biordered tree*) という. また, すべての i $(1 \leq i \leq n)$ に対して, 巡回順序 $v_i, \ldots, v_n, v_1, \ldots, v_{i-1}$ を許容する木を**巡回順序木** (*cyclic-ordered tree*) という. さらに, すべての i $(1 \leq i \leq n)$ に対して, 巡回順序 $v_i, \ldots, v_n, v_1, \ldots, v_{i-1}$ と $v_i, \ldots, v_1, v_n, \ldots, v_{i-1}$ を許容する木を**巡回両順序木** (*cyclic-biordered tree*) という.

そして, 順序木間のアライメント距離を計算するアルゴリズム [21] を拡張することによって, 両順序木間のアライメント距離を順序木 [21] と同じ $O(n^2D^2)$ 時間で計算するアルゴリズムを設計する. ここで, n は与えられた 2 つの木の最大ノード数, D は最大次数である. また, 巡回順序木と巡回両順序木のアライメント距離を $O(n^2D^4)$ 時間で計算するアルゴリズムも設計する. なお, 巡回的順序木の編集距離の計算は MAX SNA 困難である.

1.3.3　さまざまな拡張

5 章では, 木編集距離についてのさまざまな拡張について, 無順序木距離の動的 A* アルゴリズム, Tai マッピングの根無し木への拡張, 多項式時間計算可能な無順序木カーネルの

設計についての研究である.

Horesh ら [19] は, 2 つの進化系統樹を比較するために (計算が困難である) 根付き**ラベルなし**無順序木間の編集距離を計算する **A* アルゴリズム** (A* *algorithm*) を開発した. この A*アルゴリズムは, 無順序木編集距離の定数倍以下となる 3 つの**下限関数** (*lower bounding functions*) を用いている. Higuchi ら [16] は, この A* アルゴリズムを根付き**ラベル付き**無順序木間の編集距離を計算するアルゴリズムに拡張している. 彼らは, 上記の 3 つの下限関数に加えて, さらに 2 つの下限関数を採用した. また, 標準的な A*アルゴリズムの連結グラフにおける最短経路を求める問題を無順序木編集距離を計算するものに変換する**編集距離探索木** (*edit distance search tree*) を導入した. そして, 上記 5 つの下限関数の最大値をヒューリスティック関数に設定し, 最良サーチにより編集距離探索木に対して必要に応じて辺を構築する A*アルゴリズムを設計した. しかし, この A*アルゴリズムは, 何度も部分的な Tai マッピングを複数回計算しているという欠点が残っている.

このような冗長性を避けるため, 本論文では, 動的計画法と再帰呼び出しを用いた新しい A*アルゴリズムである**動的 A* アルゴリズム** (*dynamic programming A* algorithm*) を設計する. そして, 糖鎖データを用いて, 動的 A* アルゴリズムによる無順序木編集距離の計算時間を A*アルゴリズム [16], Shasha らの網羅的アルゴリズム [40], Fukagawa らのクリークアルゴリズム [12] による計算時間と比較する. また, 5 つの下限関数のどれが有効であるかを評価する.

また, 根付き木において Tai マッピングは木編集距離に対応する重要な概念であるが, この Tai マッピングを**根無し木** (*unrooted tree*) に拡張するためには, 一対一対応であることに加えて, 先祖子孫関係に代わる条件を導入する必要がある. Zhang ら [64] は, 挿入と削除を次数 2 以下のノードに制限した**次数 2 距離** (*degree-2 distance*) を根無し木に拡張する際, 根付き木の LCA 保存マッピングの拡張として, 3 つのノードの**中心** (*center*) を保存する**中心保存マッピング** (*center-preserving mapping*) を導入した. ここで, 3 つのノードの

中心とは, 2 つのノード間の 3 つのパスの共通ノードである. 本研究では, この中心に着目する.

葉に生物種のラベルが付いた根無し木もしくは根付き木として生物種の進化を表す進化系統樹 が, すべての生物種 (葉) 間の距離である距離行列から, その距離がパス上の辺の重みの総和となるように再構成できるとき, 距離行列は**加法的である** (*additive*) という. また, 距離行列から葉の高さがすべて等しい根付き進化系統樹を構成できるとき, 距離行列は**超計量的である** (*ultrametric*) という [44]. このとき, 距離行列における任意の異なる 4 つの生物種が **4 点条件** (*4-point condition*) [6] を満たすとき, そのときに限り, 距離行列は加法的となる [44]. また, 加法的距離行列における任意の異なる 3 つの生物種が **3 点条件** (*3-point condition*) を満たすとき, そのときに限り, 距離行列は超計量的になる [44].

4 点条件と 3 点条件は, 辺の重みの総和としての距離についての条件である. 本論文では, この 2 つの条件を, 中心を用いることで木のトポロジーを特徴づける条件に変更する. そして, 根無し木 T_1 と T_2 に対する単射 $M \subseteq V(T_1) \times V(T_2)$ に対して, 任意の $(u_1, u_2), (v_1, v_2), (w_1, w_2), (x_1, x_2) \in M$ に対して, (u_1, v_1) と (w_1, x_1) が 4 点条件を満たすことと (u_2, v_2) と (w_2, x_2) が 4 点条件を満たすことが同値となるとき, M を **4 点保存マッピング** (*4-point-preserving mapping*) という. 特に, 根 r_1 と r_2 を持つ根付き木 T_1 と T_2 の場合, 任意の $(u_1, u_2), (v_1, v_2), (w_1, w_2) \in M$ に対して, (u_1, v_1) と (w_1, r_1) が 4 点条件を満たすことと (u_2, v_2) と (w_2, r_2) が 4 点条件を満たすことが同値となるとき, M を**根付き 4 点保存マッピング** (*rooted 4-point-preserving mapping*) という. さらに, 任意の $(u_1, u_2), (v_1, v_2), (w_1, w_2) \in M$ に対して, (u_1, v_1, w_1) が 3 点条件を満たすことと (u_2, v_2, w_2) が 3 点条件を満たすことが同値となるとき, M を **3 点保存マッピング** (*3-point-preserving mapping*) という. M が T_1 の部分木と T_2 の部分木を対応付けるとき M を**部分木保存マッピング** (*subtree-preserving mapping*) という. また, 任意の $(u_1, u_2), (v_1, v_2), (w_1, w_2) \in M$ に対して, w_1 が u_1 と v_1 のパス上にあることと w_2 が u_2 と v_2 のパス上にあることが同値

となるとき, M を**パス保存マッピング** (*path-preserving mapping*) という.

これらの準備の下で, 本論文では, M が根付き木のマッピングの場合, 以下が成り立つことを示す.

$$M \text{ は孤立部分木マッピング } [59,\ 60] \Leftrightarrow M \text{ は 3 点保存マッピング.}$$

$$M \text{ は劣制限マッピング } [32] \Leftrightarrow M \text{ は 4 点保存マッピング}$$

$$\Leftrightarrow M \text{ は根付き 4 点保存マッピング.}$$

また, M が根無し木のマッピングの場合, 以下の含意関係が成り立つことを示す. 一般に逆は成り立たない.

$$M \text{ は部分木保存マッピング} \Rightarrow M \text{ は中心保存マッピング}$$

$$\Rightarrow M \text{ は 4 点保存マッピング} \quad \Rightarrow M \text{ はパス保存マッピング.}$$

木カーネル (*tree kernel*) は, サポートベクターマシン (SVM) を利用して木を分類するための基本的な方法の 1 つである. 順序木に対する木カーネルを設計するための多くの研究が行われている ([13, 25, 39, 41, 42, 43] 参照). 本論文ではこれを, **順序木カーネル** (*ordered tree kernels*) と呼ぶ.

マッピングカーネル (*mapping kernel*) [41, 42, 43] は, すべてのマッピング (およびその変種) を一対一のノードの対応の集合として数え上げる, 強力かつ一般的なフレームワークである. マッピングカーネルの性質として, ほとんどの順序木カーネルはマッピングカーネル [41] のフレームワークに分類され, マッピングが**推移的** (*transitive*) である, すなわち, **写像の合成の元で閉じている** (*closed under the composition*) とき, かつそのときに限り, マッピングカーネルは**正定値** (*positive definite*) となる [42, 43]. ここで, 集合 X 上のカーネル K が正定値であるとは, 任意の n と $x_1, \ldots, x_n \in X$ に対して, グラム行列 $\big(K(x_i, x_j)\big)_{(i,j)\in\{1,\ldots,n\}\times\{1,\ldots,n\}}$ が正定値であることをいう. カーネルが正定値であれば, そのカーネルを内積として持つヒルベルト空間が存在し, カーネルはその空間での類似度を

与えることになる.

一方, 無順序木に対するカーネルを設計する研究はほとんどなされていない. このカーネルを**無順序木カーネル** (*unordered tree kernels*) と呼ぶ. その理由の 1 つは無順序木の部分木をすべて数え上げる問題が#P 困難であることである [25].

このような困難を避けるために, 無順序木カーネルは部分木ではなく, 特定の部分構造をすべてカウントすることで開発されている. たとえば, Kuboyama ら [30] と木村ら [27] はそれぞれ, すべての**二葉 q グラム** (*bifoliate q-gram*) を数え上げる無順序木カーネルとすべての**部分パス** (*subpath*) を数え上げる無順序木カーネルを設計した.

無順序木のためのマッピングカーネルとして, Hamada ら [14] は進化系統樹のための**合致部分木マッピングカーネル** (*agreement-subtree mapping kernel*) を導入した. また, 彼らは鹿島ら [25] よりも簡単な, 無順序木に対するマッピングカーネルの計算困難性 (#P 完全性) の新たな証明を与えている.

マッピングのすべての変種は, 編集距離の変種を与える [22, 29, 51, 54] だけでなく, すべてのマッピングを数え上げる木カーネルを与える [29, 31, 41]. ここで, 孤立部分木距離 [59, 60], 調和的距離 [29, 31, 51], LCA 保存距離 [64], LCA 保存断片距離 [54], トップダウン距離 [7, 38] といった, 無順序木間の多項式時間計算可能な編集距離の変種を計算する問題は, 二部グラフの最小重み付き最大マッチングを解く必要がある [51, 60, 64]. 一方, 上記の#P 完全性 [14, 25] に対しては, 二部グラフのすべてのマッチングの数え上げ問題からの還元が本質的であるため, 一般の無順序木のマッピングカーネルの計算は困難であると考えられる.

そこで本論文では, 巡回的順序木に対する**トップダウンマッピングカーネル** (*top-down mapping kernel*), **LCA 保存断片マッピングカーネル** (*LCA-preserving segmental mapping kernel*), **LCA 保存マッピングカーネル** (*LCA-preserving mapping kernel*), **調和的マッピングカーネル** (*accordant mapping kernel*), **孤立部分木マッピングカーネル** (*isolated-subtree*

mapping kernel) について議論する. そして, これらすべてのマッピングカーネルが, 両順序木に対して $O(nm)$ 時間, 巡回順序木と巡回両順序木に対して $O(nmdD)$ 時間でそれぞれ計算できることを示す. ここで, n, m は 2 つの木のノード数, D は 2 つの木の次数の最大値, d は 2 つの木の次数の最小値である. また, 次数を定数 D に制限した無順序木に対するこれらのマッピングカーネルが $O(nmD^D)$ 時間で計算できることを示す. 一方, 無順序木に対しては今までの#P 完全性の証明 [14, 25] がトップダウンマッピングカーネルとボトムアップマッピングカーネルには直接適用できないため, **ラベル保存葉拡張** (*label-preserving leaf-extended*) トップダウンマッピングカーネルと**ラベル保存** (*label-preserving*) ボトムアップマッピングカーネルの計算が#P 完全であることを示す.

第2章　木編集距離と木アライメント距離

本章では, 次章以降で必要となる, 木, Tai マッピング, 木編集距離, 木アライメント距離などを導入する.

2.1　木

閉路を持たない連結グラフを**木** (*tree*) という. 木 T に対して, そのノード (頂点) 集合 V と辺集合 E をそれぞれ $V(T), E(T)$ で表し, 木そのものを (V, E) で表す. また, $v \in V(T)$ を単に $v \in T$ と表す. $|V(T)|$ を T の**大きさ** (*size*) といい, $|T|$ で表す. 簡単のため, $v \in V$ を単に $v \in T$ で表し, 空の木を $\emptyset$ で表す. 木 T_1, T_2 に対して, $(u, v) \in E(T_1) \Leftrightarrow (id(u), id(v)) \in E(T_2)$ となる全単射 $id : V(T_1) \to V(T_2)$ が存在するとき, T_1 と T_2 は同型であるという.

$T = (V, E)$ を木とする. $u, v \in V$ に対して, $V' = \{v_1, \ldots, v_n\} \subseteq V$, $E' = \{(v_i, v_{i+1}) \mid 1 \leq i \leq n-1\} \subseteq E$, $v_1 = u, v_n = v$ を満たす木 (V', E') を u から v への**パス** (*path*) といい, $[u, v]$ で表す. また, $[u, v] \setminus \{u, v\}$ を $[\![u, v]\!]$ で表す.

$T = (V, E)$ を木, Σ を有限集合 (アルファベット) とする. T の各ノードに Σ の文字が割り当てられているとき, その T を**ラベル付き木** (*labeled tree*) といい, 割り当てられた文字をラベルという. ラベルの割り当ては, 写像 $l : V \to \Sigma$ に対応する. ノード v のラベルを $l(v)$ で表し, 場合により, v と $l(v)$ を同一視する. また, $\varepsilon \notin \Sigma$ を**空文字** (*blank*) といい, $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$ とする.

木 T のうち, **根** (*root*) ノード $r \in T$ を 1 つ選んだものを**根付き木** (*rooted tree*), そう

でないものを**根無し木** (*unrooted tree*) という. 根付き木 T の根を $r(T)$ と表し, 根ノード $r \in T$ を根として選んだ根付き木を T^r で表す. 2 つの根付き木 T_1, T_2 に対して, 木同型写像 $id : V(T_1) \to V(T_2)$ が存在し, $id(r(T_1)) = r(T_2)$ が成り立つとき, T_1 と T_2 は (根付き木) 同型であるといい, $T_1 \equiv T_2$ で表す. 本論文では特に断りのない限り, 簡単のために根付きラベル付き木を単に木と表現することとする.

T を r を根とする根付き木とする. ノード $u \in T$ に対して, $[r, u]$ に含まれる u 以外のノードを, u の**先祖** (*ancestor*) といい, u の先祖が v であることを $u < v$ で表す. また, $u < v$ または $u = v$ であることを $u \leq v$ で表す. 特に, 任意の $v \in T^r$ に対して, $v \leq r$ である. さらに, $u \leq v$ でも $v \leq u$ でもないことを $u \# v$ で表す. ノード v に対して, 先祖の数を v の**深さ** (*depth*) といい, $depth(v)$ で表す. すなわち, $depth(v) = |\{u \in T \mid v < u\}|$ である. ノード $u, v \in T$ に対して, $u < v$ であるとき, u を v の**子孫** (*descendant*) という. $(u, v) \in E(T)$ を満たす v の先祖 u を v の**親** (*parent*) ノードといい, $par(v)$ で表す. また, v を親に持つノードを v の**子** (*child*) ノードといい, v の子ノードの集合を $ch(v)$ で表す. さらに, 同じ親ノードを持つノードを**兄弟** (*sibling*) ノードという. ノード v に対して, 子ノードの数を v の**次数** (*degree*) といい, $d(v)$ で表す. また, $max\{d(v) \mid v \in T\}$ を T の次数といい, $d(T)$ で表す. さらに, 子ノードを持たないノードを**葉** (*leaf*) ノードといい, T の葉の集合を $lv(T)$ で表す.

木 $T = (V, E)$ に対して, $V' \subseteq V$, $E' \subseteq E$, $T' = (V', E')$ とする. すべての $u, v \in V'$ に対して u から v へのパスが T' 内に存在するとき, T' を T の**部分木** (*subtree*) という. また, ノード $v \in T$ に対して, v と v の子孫からなる部分木を, T の v を根とする部分木という. さらに, v と v の子孫すべてを含む部分木を, T の v を根とする**完全部分木** (*complete subtree*) といい, $T[v]$ で表す. ノード $u, v \in T$ に対して, u と v の共通する先祖のうち, 最も u, v に近いノードを u, v の**最近共通先祖** (*least common ancestor*) といい, $u \sqcup v$ で表す.

根付き木 T に対するノードの走査を考える. あるノードを走査するときに, そのノード

自身を走査してから子ノードの走査に移るような走査を**先行走査** (*pre-order traversal*) という. 逆に, 子ノードの走査を終えてからそのノード自身の走査を行うような走査を**後行走査** (*post-order traversal*) という. 木 T に先行走査を行うとき, ノード $v \in T$ より前に走査されたノードの数を $pre_T(v)$ で表す. 同様に, 木 T に後行走査を行うとき, ノード $v \in T$ より前に走査されたノードの数を $post_T(v)$ で表す.

すべてのノード $v \in T$ に対して, 子ノード集合 $ch(v)$ 上の全順序関係 $\preceq_v$ が与えられたとき, その根付き木 T を**順序木** (*ordered tree*) といい, そうでないときを**無順序木** (*unordered tree*) という. この $\preceq_v$ を v の子ノードの兄弟関係という. $v_1, v_2 \in ch(v)$ に対して, $v_1 \preceq_v v_2$ かつ $v_2 \npreceq_v v_1$ を $v_1 \prec_v v_2$ で表す. 順序木 T の根でないノード u, v に対して, $u \leq u'$, $v \leq v'$ を満たす $u', v' \in ch(u \sqcup v)$ が存在する. このとき, $u' \preceq_{u \sqcup v} v'$ であることを $u \preceq v$ で表し, $u' \prec_{u \sqcup v} v'$ であることを $u \prec v$ で表す. また, この関係 $\preceq$ を, T の $\preceq_v$ から得られるノード間左右関係という.

2 つの順序木 T_1, T_2 に対して, 根付き木同型写像 $id : V(T_1) \to V(T_2)$ が存在し, 任意の $u, v \in T_1$ に対して, $u \prec v \Longrightarrow id(u) \prec id(v)$ が成り立つとき, T_1 と T_2 は (順序木) 同型であるといい, $T_1 \equiv^o T_2$ で表す. また, 特に無順序木として同型であることを強調したい場合には, $T_1 \equiv T_2$ を $T_1 \equiv^u T_2$ と表す.

要素数 n の集合 $\{1, \ldots, n\}$ からその集合自身への全単射を n **置換** (n-*permutation*), あるいは単に**置換** (*permutation*) という. n 置換 σ を $(\sigma(1), \ldots, \sigma(n))$ で表す. また, n 置換すべてからなる集合を Σ_n で表す. さらに, $\pi(i) = i (1 \leq i \leq n)$ となる n 置換 π を**恒等** n **置換** (*identity n-permutation*) といい, I_n で表す.

定義 2.1 (巡回的順序木)**.** $1 \leq p \leq n$ に対して, n 置換 $\beta_n, \gamma_{p,n}, \gamma_{p,n}^{-1}$ を次のように定める.

$$\beta_n(i) = n - i + 1,$$

$$\gamma_{p,n}(i) = ((i + p - 1) \mod n) + 1,$$

$$\gamma_{p,n}^{-1}(i) = ((n - i - p + 1) \mod n) + 1.$$

すなわち,

$$\beta_n = (n, n-1, \ldots, 1),$$

$$\gamma_{p,n} = (p+1, p+2, \ldots, n, 1, 2, \ldots, p),$$

$$\gamma_{p,n}^{-1} = (n-p+1, n-p, \ldots, 2, 1, n, n-1, \ldots, n-p).$$

このとき, 明らかに $\gamma_{n,n} = I_n$ である.

すべての葉でないノード $v \in T \setminus lv(T)$ に対して, $\Pi_v \subseteq \Sigma_{d(v)}$ が与えられているとき, T を**順列木** (*permuted tree*) という. このとき, Π_v を v の**許容される** (*admissible*) 置換という. また, 直積 $\bigotimes_{v \in T} \Pi_v$ を T の許容される置換といい, Π_T で表す.

1. $\Pi_T = \bigotimes_{v \in T} \{I_{d(v)}, \beta_{d(v)}\}$ であるとき, T を**両順序木** (*biordered tree*) という.
2. $\Pi_T = \bigotimes_{v \in T} \{\gamma_{p,d(v)} \mid 1 \leq p \leq n\}$ であるとき, T を**巡回順序木** (*cyclic-ordered tree*) という.
3. $\Pi_T = \bigotimes_{v \in T} \{\gamma_{p,d(v)}, \gamma_{p,d(v)}^{-1} \mid 1 \leq p \leq n\}$ であるとき, T を**巡回両順序木** (*cyclic-biordered tree*) という.

特に, 両順序木, 巡回的順序木, 巡回両順序木を合わせて**巡回的順序木** (*cyclically ordered tree*) という. また, $\Pi_T = \bigotimes_{v \in T} \{I_{d(v)}\}$ のとき T は順序木であり, $\Pi_T = \bigotimes_{v \in T} \Sigma_{d(v)}$ のとき T は無順序木である.

順列木 T の各ノード v に対して, v の子ノードの兄弟関係 $\preceq_v$ が与えられているとする. (すなわち, T は順序木でもある.) また, $\pi \in \Pi_v$ とする. さらに, v の子ノード $v_1, \ldots, v_{d(v)} \in ch(v)$ を, $v_1 \prec_v v_2 \prec_v \cdots \prec_v v_{d(v)}$ を満たすようにとる. このとき, π の逆写像 π^{-1} を用いて $v_{\pi^{-1}(1)} \prec'_v v_{\pi^{-1}(2)} \prec'_v \cdots \prec'_v v_{\pi^{-1}(d(v))}$ となるような $\prec'_v$ を定めることにより得られる全順序関係 $\preceq'_v$ を, $\preceq_v$ を π で並べ替えた (v の子ノードの) 兄弟関係という.

T をノード数 n の順列木とし, すべての $v_i \in T$ の子ノードに対して, 兄弟関係 $\preceq_{v_i}$ が与えられているとする. また, $\pi = (\pi_1, \ldots, \pi_n) \in \Pi_T$(ただし $\pi_i \in \Pi_{v_i}$) とする. このとき,

$\preceq_{v_i}$ を π_i で並び替えた兄弟関係 $\preceq'_{v_i}$ を与えることにより得られる順序木を $\pi(T)$ で表し, $\{\pi(T) \mid \pi \in \Pi_T\}$ を $\Pi(T)$ で表す.

根付き木同型写像 $id : V(T_1) \to V(T_2)$ が存在する 2 つの順列木 T_1, T_2 を考える. $T'_1 \equiv^o T'_2$ を満たす順序木 $T'_1 \in \Pi(T_1), T'_2 \in \Pi(T_2)$ が存在するとき, T_1 と T_2 を順列木として同型という. 特に, T_1, T_2 が両順序木 (巡回順序木, 巡回両順序木) の場合, 順列木同型な T_1 と T_2 を両順序木同型 (巡回順序木同型, 巡回両順序木同型) といい $T_1 \equiv^b T_2$ ($T_1 \equiv^c T_2$, $T_1 \equiv^{cb} T_2$) と表す.

木の集まり $[T_1, ..., T_n]$ を**森** (*forest*) という. 特に断りのない限り, 順序木の森は順序木の列, 無順序木の森は無順序木の集合を指すものとする. 森 F の各木の根ノードを, 新たなノード v と接続することで得られる (v を根とする) 木を $v(F)$ で表す. 木 T とそのノード $v \in T$ に対して, $T[v]$ から v を取り除くことで得られる森を $T(v)$ で表す. 特に, 順序木の森 F の左側に順序木 T を並べたものを $T \bullet F$ で表す. また, ノード $v \in T$ の子ノードを左から順に $v_1, \ldots, v_{d(v)}$ としたとき, $T(v)$ の左から数えて, i 本目から j 本目の木を集めた森 $[T[v_i], \ldots, T[v_j]]$ を $F(v_i, v_j)$ で表す.

2.2　木編集距離と Tai マッピング

本節では, 編集距離と, それに密接に関係する Tai マッピングの導入を行う. まず, 編集距離を導入するために, 木の編集操作を定義する.

定義 2.2 (編集操作)**.** 木 T の**編集操作** (*edit operation*) は以下のように定義される (図 2.1 参照):

1. **置換** (*substitution*): ノード $v \in T$ のラベルを変更する.
2. **削除** (*deletion*): v' を親に持つノード $v \in T$ を削除し, v の子を v' の子にする. T が順序木の場合は, v があった位置に部分列として v の子を挿入し, 無順序木の場合は,

v' の子と v の子の和集合を取り新たに v' の子とする.

3. **挿入** (*insertion*): 削除の逆操作. ノード v を $v' \in T$ の子として挿入し, v' の子の一部を v の子にする. T が順序木の場合は, v' の子の部分列を左右の順序を変えないように v の子にする. T が無順序木の場合は, v' の子の部分集合を v の子にする.

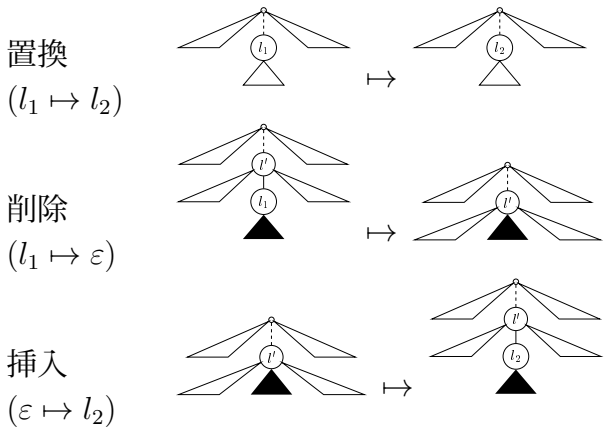


図 2.1: 木に対する編集操作.

ここで, ラベルの組 $(l_1, l_2) \in (\Sigma_\varepsilon \times \Sigma_\varepsilon \setminus \{(\varepsilon, \varepsilon)\})$ に対して, 編集操作を $(l_1 \mapsto l_2)$ と書き, $l_1 \neq \varepsilon$ かつ $l_2 \neq \varepsilon$ のときに置換を, $l_2 = \varepsilon$ のときに削除を, $l_1 = \varepsilon$ のときに挿入をそれぞれ表す. ノード u, v に対して, $(l(u) \mapsto l(v))$ を単に $(u \mapsto v)$ と表す. さらに, ラベルの組に, **コスト関数** (*cost function*)γ: $(\Sigma_\varepsilon \times \Sigma_\varepsilon \setminus \{(\varepsilon, \varepsilon)\}) \to \mathbb{R}^+$ を定義する. 特に断らない限り, コスト関数には $l_1 = l_2$ のとき $\gamma(l_1, l_2) = 0$, $l_1 \neq l_2$ のとき $\gamma(l_1, l_2) = 1$ となる**単一コスト関数** (*unit cost function*) を用いる. 関数 $f : A \times A \to \mathbb{R}^+$ が, 任意の $a, b, c \in A$ に対して, (1) $f(a, b) = 0 \Leftrightarrow a = b$ (2) $f(a, b) = f(b, a)$ (3) $f(a, c) \leq f(a, b) + f(b, c)$ を満たすとき, f は**メトリック** (*metric*) であるという. 単一コストはメトリックである.

定義 2.3 (編集距離)**.** γ をコスト関数とする. 編集操作 $e = (l_1 \mapsto l_2)$ に対して, $\gamma(e) = \gamma(l_1, l_2)$ で与えられる値を, 編集操作 e のコストという. また, 編集操作列 $E = e_1, \ldots, e_k$ に対して, $\gamma(E) = \sum_{i=i}^{k} \gamma(e_i)$ で与えられる値を, 編集操作列 E のコストという. そして,

木 T_1 と T_2 に対して, 以下のように定義される $\tau_{\mathrm{TAI}}(T_1, T_2)$ を, T_1 と T_2 間の**編集距離** (*edit distance*) という.

$$\tau_{\mathrm{TAI}}(T_1, T_2) = \min\{\gamma(E) \mid E \text{ は } T_1 \text{から } T_2 \text{を得るための編集操作列}\}.$$

定理 2.4 (木編集距離の計算時間)**.** $|T_1| = n$, $|T_2| = m$, $n \geq m$ とする. このとき, 順序木間の編集距離は $O(nm^2(1 + \log \frac{n}{m})))$ 時間で計算可能である [8]. 一方, 無順序木間の編集距離の計算問題は MAX SNP 困難である [61]. これは, 木の次数が 2 の場合でも成り立ち [18], 木の高さが 2 の場合でも成り立つ [2].

次に, 木編集距離の宣言的意味となる Tai マッピングを導入する.

定義 2.5 (Tai マッピング [45])**.** T_1 と T_2 を木とし, $M \subseteq V(T_1) \times V(T_2)$ とする. このとき, 任意の $(u_1, v_1), (u_2, v_2) \in M$ が

1. $u_1 = u_2 \Leftrightarrow v_1 = v_2$ (一対一対応),
2. $u_1 \leq u_2 \Leftrightarrow v_1 \leq v_2$ (先祖関係保存),
3. $u_1 \preceq u_2 \Leftrightarrow v_1 \preceq v_2$ (兄弟関係保存)

を満たすとき, 3つ組 (M, T_1, T_2) を**順序 Tai マッピング** (*ordered Tai mapping*) という. また, 一対一対応と先祖子孫関係を満たす 3 つ組 (M, T_1, T_2) を**無順序 Tai マッピング** (*unordered Tai mapping*) という. 誤解のない限り, (M, T_1, T_2) を単に M と書く. また, 順序 Tai マッピングや無順序 Tai マッピングを単に**マッピング**といい, $M \in \mathcal{M}_{\mathrm{TAI}}(T_1, T_2)$ と書く.

特に, 任意の $u \in T_1$ に対して, $(u, v) \in M$ となる $v \in T_2$ が存在するような M を, T_1 から T_2 への**包含マッピング** (*inclusion mapping*) という. このとき, v を $M(u)$ で表す. また, マッピング M に対して, $\{u \mid (u, v) \in M\}$ を $M|_1$, $\{v \mid (u, v) \in M\}$ を $M|_2$ でそれぞれ表す.

M を T_1 から T_2 へのマッピングとし, γ をコスト関数とする. I_M と J_M をそれぞれ T_1 と T_2 のノードのうち, M に含まれないノードの集合, すなわち $I_M = V(T_1) \setminus M|_1$, $J_M = V(T_2) \setminus M|_2$ とする. このとき, M のコスト $\gamma(M)$ を以下のように与える.

$$\gamma(M) = \sum_{(u,v)\in M} \gamma(u,v) + \sum_{u\in I_M} \gamma(u,\varepsilon) + \sum_{v\in J_M} \gamma(\varepsilon,v).$$

Tai マッピングが編集距離の宣言的意味を与えることは, 以下の定理に基づいている.

定理 2.6 (Tai マッピングと編集距離の関係 [45])**.** 以下が成り立つ.

$$\tau_{\text{TAI}}(T_1,T_2) = \min\{\gamma(M) \mid M \in \mathcal{M}_{\text{TAI}}(T_1,T_2)\}.$$

次に, Tai マッピングに制限を加えることでマッピングの変種を以下のように導入する.

定義 2.7 (マッピングの変種)**.** T_1 と T_2 を木とし, $M \in \mathcal{M}_{\text{TAI}}(T_1,T_2)$ とする. また, $M \setminus \{(r(T_1),r(T_2))\}$ を M^- で表す.

1. M が以下の条件を満たすとき, M を**劣制限マッピング** (*less-constrained mapping*) [32] といい, $M \in \mathcal{M}_{\text{LESS}}(T_1,T_2)$ で表す.

$$\forall (u_1,v_1),(u_2,v_2),(u_3,v_3) \in M \Big(u_1 \sqcup u_2 < u_1 \sqcup u_3 \Longrightarrow v_2 \sqcup v_3 = v_1 \sqcup v_3 \Big).$$

2. M が以下の条件を満たすとき, M を**孤立部分木マッピング** (*isolated-subtree mapping*) [49] または**制限マッピング** (*constrained mapping*) [59, 60] といい, $M \in \mathcal{M}_{\text{ILST}}(T_1,T_2)$ で表す.

$$\forall (u_1,v_1),(u_2,v_2),(u_3,v_3) \in M \Big(u_3 < u_1 \sqcup u_2 \iff v_3 < v_1 \sqcup v_2 \Big).$$

3. M が以下の条件を満たすとき, M を**調和的マッピング** (*accordant mapping*) [29] または **Lu マッピング** (*Lu's mapping*) [33] [1] といい, $M \in \mathcal{M}_{\mathrm{Acc}}(T_1, T_2)$ で表す.

$$\forall (u_1, v_1), (u_2, v_2), (u_3, v_3) \in M \Big(u_1 \sqcup u_2 = u_1 \sqcup u_3 \iff v_1 \sqcup v_2 = v_1 \sqcup v_3 \Big).$$

4. M が以下の条件を満たすとき, M を**LCA 保存マッピング** (*LCA-preserving mapping*) または**次数 2 マッピング** (*degree-2 mapping*) [64] といい, $M \in \mathcal{M}_{\mathrm{LCA}}(T_1, T_2)$ で表す.

$$\forall (u_1, v_1), (u_2, v_2) \in M \Big((u_1 \sqcup u_2, v_1 \sqcup v_2) \in M \Big).$$

5. M が以下の条件を満たすとき, M を**トップダウンマッピング** (*top-down mapping*) [7, 38] または**次数 1 マッピング** (*degree-1 mapping*) [4] といい, $M \in \mathcal{M}_{\mathrm{Top}}(T_1, T_2)$ で表す.

$$\forall (u, v) \in M^{-} \Big((par(u), par(v)) \in M \Big).$$

6. M が以下の条件を満たすとき, M を**ボトムアップマッピング** (*bottom-up mapping*) [29, 47, 51] [2] といい, $M \in \mathcal{M}_{\mathrm{Bot}}(T_1, T_2)$ で表す.

$$\forall (u, v) \in M \left(\begin{array}{l} \forall u' \in T_1[u] \exists v' \in T_2[v] \Big((u', v') \in M \Big) \\ \quad \wedge \forall v' \in T_2[v] \exists u' \in T_1[u] \Big((u', v') \in M \Big) \end{array} \right).$$

7. M が以下の条件を満たすとき, M を**断片マッピング** (*segmental mapping*) [22] とい

[1] Lu は [33] にて, 編集距離を求めるアルゴリズムを設計していたが, 実際にはより制約の強い距離を求めるアルゴリズムとなっていた. また, Lu はマッピングに相当するものを提示していない. [33] のアルゴリズムが求める距離に対応するマッピングとして, Kuboyama [29] が導入したマッピングが調和的マッピングである. ここでは [29] に従い, Lu のアルゴリズムが求める距離に対応するマッピングを Lu マッピングとも呼称する.

[2] Valiente [47] は孤立部分木マッピングにおいてボトムアップマッピングを導入したが, 彼のアルゴリズムは孤立部分木距離ではなくボトムアップ距離を計算するものであった. したがって, 本論文では孤立部分木マッピングではなく, Tai マッピングにおける定義を採用する. 詳しくは [29, 51] を参照のこと.

い, $M \in \mathcal{M}_{\mathrm{SG}}(T_1, T_2)$ で表す.

$$\forall (u,v) \in M^{-} \left(\begin{array}{c} \exists (u',v') \in M \Big(\Big(u' \in anc(u) \Big) \wedge \Big(v' \in anc(v) \Big) \Big) \\ \Longrightarrow \Big((par(u), par(v)) \in M \Big) \end{array} \right).$$

8. M が $(r(T_1), r(T_2)) \in M$ を満たす断片マッピングであるとき, M を**トップダウン断片マッピング** (*top-down segmental mapping*) [22] といい, $M \in \mathcal{M}_{\mathrm{TOPSG}}(T_1, T_2)$ で表す.

例 2.8. 図 2.2 にマッピングの例 M_i $(1 \leq i \leq 8)$ [22, 29] を示す. ここで, 各マッピングは $M_1 \in \mathcal{M}_{\mathrm{TOP}}(T_1, T_2)$ だが $M_1 \notin \mathcal{M}_{\mathrm{BOT}}(T_1, T_2)$; $M_2 \in \mathcal{M}_{\mathrm{LCA}}(T_1, T_2)$ だが $M_2 \notin \mathcal{M}_{\mathrm{TOP}}(T_1, T_2)$; $M_3 \in \mathcal{M}_{\mathrm{ACC}}(T_1, T_2)$ だが $M_3 \notin \mathcal{M}_{\mathrm{LCA}}(T_1, T_2)$; $M_4 \in \mathcal{M}_{\mathrm{ILST}}(T_1, T_2)$ だが $M_4 \notin \mathcal{M}_{\mathrm{ACC}}(T_1, T_2)$; $M_5 \in \mathcal{M}_{\mathrm{ALN}}(T_1, T_2)$ だが $M_5 \notin \mathcal{M}_{\mathrm{ILST}}(T_1, T_2)$; $M_6 \in \mathcal{M}_{\mathrm{TAI}}(T_1, T_2)$ だが $M_6 \notin \mathcal{M}_{\mathrm{ALN}}(T_1, T_2)$; $M_7 \in \mathcal{M}_{\mathrm{SG}}(T_1, T_2)$ だが $M_7 \notin \mathcal{M}_{\mathrm{TOP}}(T_1, T_2)$ かつ $M_7 \notin \mathcal{M}_{\mathrm{BOT}}(T_1, T_2)$; $M_8 \in \mathcal{M}_{\mathrm{BOT}}(T_1, T_2)$ だが $M_8 \notin \mathcal{M}_{\mathrm{ALN}}(T_1, T_2)$ である.

特に, 断片マッピング $M \in \mathcal{M}_{\mathrm{SG}}(T_1, T_2)$ について, 定義 2.7.8 の式は $u' \in anc(u)$, $v' \in anc(v)$ となる $(u,v), (u',v') \in M$ に対して, $u_1 = u$, $u_k = u'$, $v_1 = v$, $v_k = v'$, $u_{i+1} = par(u_i)$, $v_{i+1} = par(v_i)$ $(1 \leq i \leq k-1)$, $(u_i, v_i) \in M$ $(1 \leq i \leq k)$ が成り立つような $u_1, \ldots, u_k \in T_1$, $v_1, \ldots, v_k \in T_2$ が存在することを示している.

定義2.7のマッピングは, 図2.3の**Taiマッピング階層** (*Tai mapping hierarchy*) を与える. 図 2.3 の階層は, これまでに新たなマッピングを追加することによって, 図の左から右へと拡張されてきている. ここで, Iso は同型写像を表す.

また, マッピングの変種に対して, 以下のように編集距離の変種を定義する.

定義 2.9. すべてのTaiマッピング階層中の $\mathcal{M}_{\mathtt{A}}(T_1, T_2)$ ($\mathtt{A} \in \{\textsc{Less}, \textsc{Ilst}, \textsc{Acc}, \textsc{Lca}, \textsc{Top}, \textsc{TopSg}, \textsc{Sg}, \textsc{Bot}\}$) に対して, 木 T_1, T_2 間の距離 $\tau_{\mathtt{A}}(T_1, T_2)$ を以下のように定義する.

$$\tau_{\mathtt{A}}(T_1, T_2) = \min\{\gamma(M) \mid M \in \mathcal{M}_{\mathtt{A}}(T_1, T_2)\}.$$

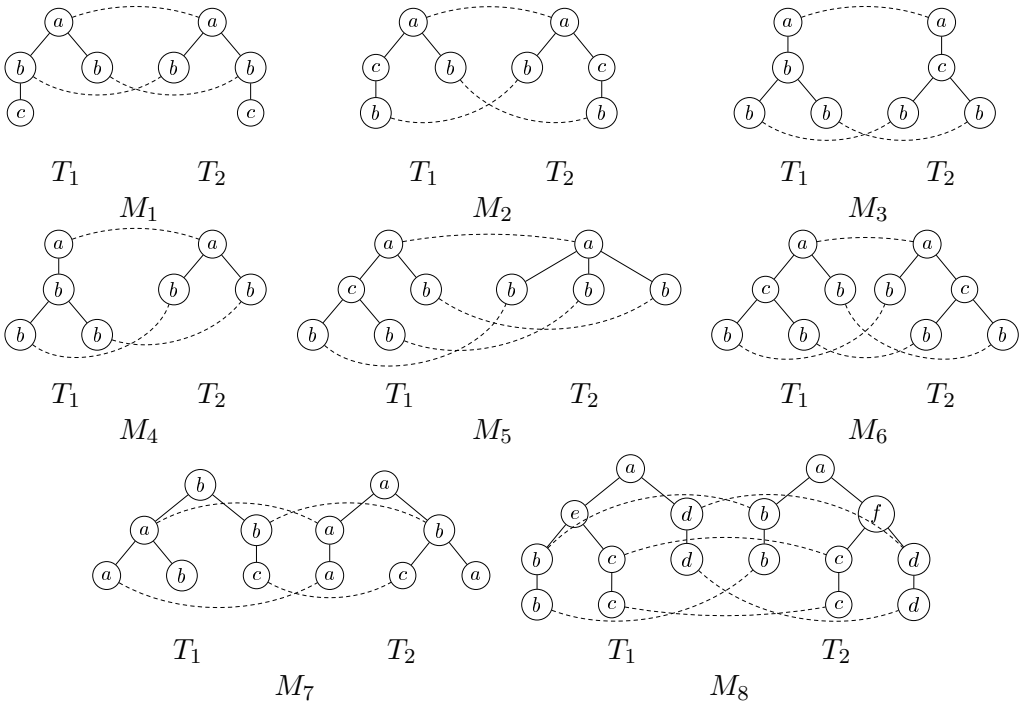


図 2.2: 例 2.8 のマッピング M_i $(1 \leq i \leq 8)$.

さらに, 木 T_1, T_2 とコスト関数 γ に対して, 最適なマッピングの集合 $\{M \in \mathcal{M}_{\mathtt{A}} \mid \gamma(M) = \tau_{\mathtt{A}}(T_1, T_2)\}$ を $\mathcal{M}^*_{\mathtt{A}}(T_1, T_2, \gamma)$ で表す.

木 T_1, T_2 に対して, T_1, T_2 と両順序 (巡回順序, 巡回両順序) 木として同型な木 T_1', T_2' を考える. また, 同型写像を $id_i : V(T_i) \to V(T_i')(i = 1, 2)$ とする. このとき T_1', T_2' 間の順序木マッピング M' に対して, $M = \{(u, v) \in V(T_1) \times V(T_2) \mid (id_1(u), id_2(v)) \in M'\}$ となるようなマッピング M を, T_1, T_2 間の**両順序マッピング** (*biordered mapping*) (**巡回順序マッピング** (*cyclic-ordered mapping*), **巡回両順序マッピング** (*cyclic-biordered mapping*)) という. また, $M' \in \mathcal{M}_{\mathtt{A}}(T_1, T_2)$ のとき, M が順序木マッピング, 両順序木マッピング, 巡回順序木マッピング, 巡回両順序木マッピング, 無順序木マッピングであることをそれぞれ, $M \in \mathcal{M}^o_{\mathtt{A}}(T_1, T_2)$, $M \in \mathcal{M}^b_{\mathtt{A}}(T_1, T_2)$, $M \in \mathcal{M}^c_{\mathtt{A}}(T_1, T_2)$, $M \in \mathcal{M}^{cb}_{\mathtt{A}}(T_1, T_2)$, $M \in \mathcal{M}^u_{\mathtt{A}}(T_1, T_2)$ で表す.

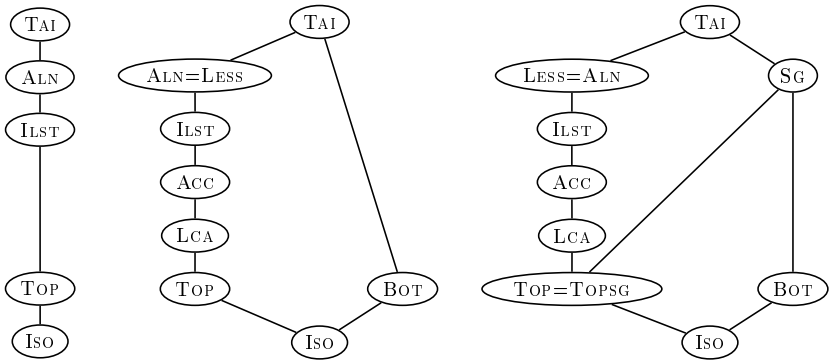


図 2.3: それぞれ Wang, Zhang [49] (左), Kuboyama [29] (中央) Kan ら [22] (右) によって導入された Tai マッピング階層 (再掲).

2.3　木アライメント距離

本節では, 2 つの根付き木間に定義されるアライメント木を導入し, アライメント木により定義されるアライメント距離を導入する.

定義 2.10 (アライメント木)**.** T_1, T_2, $\mathcal{T}$ を木とし, $I_i(i = 1, 2)$ を T_i から $\mathcal{T}$ への包含マッピングとする. さらに, $l_i : \mathcal{T} \to \Sigma_\varepsilon$ を, $I_i(v) = u \in \mathcal{T}$ なる $v \in V(T_i)$ が存在するとき $l_i(u) = l(v)$, そのような v が存在しないとき $l_i = \varepsilon$ とする. このとき, $\mathcal{T}$ のすべてのノード u を $(l_1(u), l_2(u))$ でラベル付けした木を T_1, T_2 間の**アライメント木** (*alignment tree*) という.

定義 2.11 (アライメント距離)**.** $\mathcal{T}$ を T_1, T_2 間のアライメント木とし, γ をコスト関数とする. このとき, $\mathcal{T}$ のコスト $\gamma(\mathcal{T})$ を以下のように定義する.

$$\gamma(\mathcal{T}) = \sum_{(l_1,l_2)\in\mathcal{T}} \gamma(l_1, l_2).$$

コストが最小となる T_1, T_2 間のアライメント木を T_1, T_2 間の**最適アライメント** (*optimal alignment*) といい, 最適アライメント $\mathcal{T}$ のコスト $\gamma(\mathcal{T})$ を T_1, T_2 間の**アライメント距離** (*alignment distance*) といい, $\alpha(T_1, T_2)$ と表す.

例 2.12. 図 2.4 の右の木 $\mathcal{T}$ は, 図 2.4 の左で示す木 T_1, T_2 に対するアライメント木である.

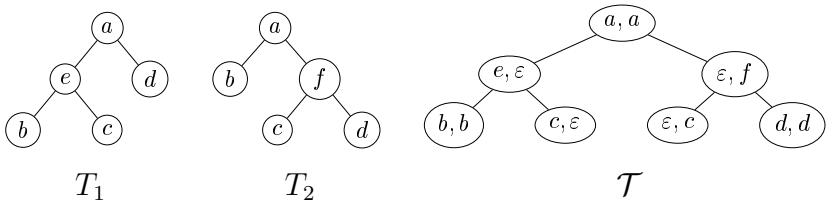


図 2.4: 例 2.12 の順序木 T_1, T_2 (左) とその最適アライメント $\mathcal{T}$(右).

アライメントをマッピングで特徴づけるために, Kuboyama [29] は以下のようなアライメント可能マッピングを導入した.

定義 2.13 (Kuboyama [29])**.** 木 $\mathcal{T}$ に対し, T_1 から $\mathcal{T}$ への包含マッピング I_1 と T_2 から $\mathcal{T}$ への包含マッピング I_2 が存在して, 任意の $(u, v) \in M$ に対し, $I_1(u) = I_2(v)$ が成り立つとき, M を**アライメント可能マッピング** (*alignable mapping*) といい, $M \in \mathcal{M}_{\text{ALN}}(T_1, T_2)$ で表す.

そして Kuboyama [29] は, アライメント可能マッピングについて, 以下の定理を示した.

定理 2.14 (Kuboyama [29])**.** 木 T_1 と T_2 に対して以下が成り立つ.

1. $\alpha(T_1, T_2) = \min\{\gamma(M) \mid M \in \mathcal{M}_{\text{ALN}}(T_1, T_2)\}$.
2. $\mathcal{M}_{\text{ALN}}(T_1, T_2) = \mathcal{M}_{\text{LESS}}(T_1, T_2)$.

定理 2.14.1 より, 以後, $\alpha(T_1, T_2)$ を $\tau_{\text{ALN}}(T_1, T_2)$ と表す. また, 定理 2.14.2 は, 4 章のアンカーアライメント問題において別証明を与える形で触れる.

定理 2.15 (木アライメント距離の計算時間 [21])**.** 木 T_1, T_2 に対して, $n = \max\{|T_1|, |T_2|\}$, $D = \max\{d(T_1), d(T_2)\}$ とする. このとき, 順序木間のアライメント距離は, $O(n^2D^2)$ 時間で計算可能である. 一方, 無順序木間のアライメント距離の計算は MAX SNP 困難であるが, 2 つの木の次数が定数 D 以下である場合は, $O(n^2D^2D!)$ 時間で計算可能である.

第3章　共通部分森に基づくTaiマッピング階層

本章では, Taiマッピング階層を共通部分森の観点から捉え直すことでマッピングを特徴付けるとともに, 新たなマッピングを追加することでTaiマッピング階層の補填を行う. また, 新たなマッピングの変種に対応する木編集距離の変種について, 順序木と無順序木の場合での計算時間を解析する. なお, 本章の内容は論文 [57] に基づいている.

3.1　Taiマッピング階層

2.2章の定義2.5によって導入したTaiマッピングの変種を, 以下のように2つのマッピングの共通部分を取ることで拡張する.

定義 3.1 (マッピングの変種)**.** T_1 と T_2 を木とし, $M \in \mathcal{M}_{\mathrm{TAI}}(T_1, T_2)$ とする. また, $M \setminus \{(r(T_1), r(T_2))\}$ を M^- で表す. マッピング $\mathtt{A}$, $\mathtt{B}$ に対して, $\mathcal{M}_{\mathtt{A}}(T_1, T_2) \cap \mathcal{M}_{\mathtt{B}}(T_1, T_2)$ を $\mathcal{M}_{\mathtt{AB}}(T_1, T_2)$ と表す. 本稿では以下の組み合わせを扱う.

$\mathcal{M}_{\mathrm{ILSTSG}}(T_1, T_2)$, $\mathcal{M}_{\mathrm{ACCSG}}(T_1, T_2)$, $\mathcal{M}_{\mathrm{LCASG}}(T_1, T_2)$, $\mathcal{M}_{\mathrm{SGALN}}(T_1, T_2)$, $\mathcal{M}_{\mathrm{BOTALN}}(T_1, T_2)$, $\mathcal{M}_{\mathrm{ILSTBOT}}(T_1, T_2)$, $\mathcal{M}_{\mathrm{ACCBOT}}(T_1, T_2)$, $\mathcal{M}_{\mathrm{LCABOT}}(T_1, T_2)$.

また, LCA保存マッピングに制限を加えたLCA保存根保存マッピングを以下の通りに定義する.

定義 3.2. T_1 と T_2 を木とし, $M \in \mathcal{M}_{\mathrm{LCA}}(T_1, T_2)$ とする. このとき, $(r(T_1), r(T_2)) \in M$ を満たす M を **LCA 保存根保存マッピング** (*LCA- and root-preserving mapping*) といい, M が LCA 保存根保存マッピングであることを $M \in \mathcal{M}_{\mathrm{LCART}}(T_1, T_2)$ で表す.

定義 2.7 と, [22, 29, 49] の結果より, 以下の補題が成り立つ.

補題 3.3. T_1, T_2 を木とする. このとき, 以下の関係が成り立つ.

1. $\mathcal{M}_{\mathrm{ISO}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{TOP}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{LCA}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{ACC}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{ILST}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{ALN}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{TAI}}(T_1, T_2)$ [29, 49].
2. $\mathcal{M}_{\mathrm{ISO}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{BOT}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{SG}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{TAI}}(T_1, T_2)$ [22].
3. マッピング A, B に対して, $\mathcal{M}_{\mathtt{AB}}(T_1, T_2) \subseteq \mathcal{M}_{\mathtt{A}}(T_1, T_2)$ となる.
4. マッピング A, B, C に対して, もし $\mathcal{M}_{\mathtt{B}}(T_1, T_2) \subseteq \mathcal{M}_{\mathtt{C}}(T_1, T_2)$ ならば, $\mathcal{M}_{\mathtt{AB}}(T_1, T_2) \subseteq \mathcal{M}_{\mathtt{AC}}(T_1, T_2)$ となる.

定理 3.4. Tai マッピングとその変種は図 3.1 に示す階層を与える. ここで, マッピング $\mathcal{M}_{\mathtt{A}}(T_1, T_2)$ をその添え字 A で表している. 特に, 以下の関係式が成り立つ. ただし, $S \# S'$ は, 2 つの集合 S, S' に対して, $S \subseteq S'$ でも $S' \subseteq S$ でもないことを表す.

1. $\mathcal{M}_{\mathrm{ACCSG}}(T_1, T_2) = \mathcal{M}_{\mathrm{ILSTSG}}(T_1, T_2)$ かつ $\mathcal{M}_{\mathrm{ACCBOT}}(T_1, T_2) = \mathcal{M}_{\mathrm{ILSTBOT}}(T_1, T_2)$.
2. $\mathcal{M}_{\mathrm{LCASG}}(T_1, T_2) \# \mathcal{M}_{\mathrm{LCART}}(T_1, T_2)$, $\mathcal{M}_{\mathrm{TOP}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{LCASG}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{LCA}}(T_1, T_2)$ かつ $\mathcal{M}_{\mathrm{TOP}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{LCART}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{LCA}}(T_1, T_2)$.
3. $\mathcal{M}_{\mathrm{LCASG}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{ACCSG}}(T_1, T_2)$, $\mathcal{M}_{\mathrm{ACCSG}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{ACC}}(T_1, T_2)$ かつ $\mathcal{M}_{\mathrm{ACCSG}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{SGALN}}(T_1, T_2)$.
4. $\mathcal{M}_{\mathrm{LCA}}(T_1, T_2) \# \mathcal{M}_{\mathrm{ACCSG}}(T_1, T_2)$.

5. $\mathcal{M}_{\text{SgAln}}(T_1, T_2) \subset \mathcal{M}_{\text{Sg}}(T_1, T_2)$ かつ $\mathcal{M}_{\text{BotAln}}(T_1, T_2) \subset \mathcal{M}_{\text{Bot}}(T_1, T_2)$.

6. $\mathcal{M}_{\text{BotAln}}(T_1, T_2) \subset \mathcal{M}_{\text{SgAln}}(T_1, T_2)$.

7. $\mathcal{M}_{\text{LcaBot}}(T_1, T_2) \subset \mathcal{M}_{\text{AccBot}}(T_1, T_2) \subset \mathcal{M}_{\text{BotAln}}(T_1, T_2) \subset \mathcal{M}_{\text{Bot}}(T_1, T_2)$.

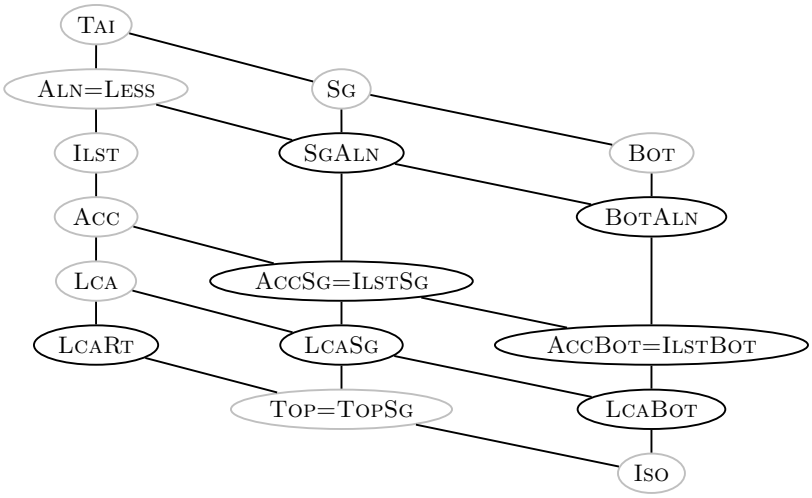


図 3.1: 新たな Tai マッピング階層.

[証明]. 1. $\mathcal{M}_{\text{AccSg}}(T_1, T_2) = \mathcal{M}_{\text{IlstSg}}(T_1, T_2)$ のみを示すが, $\mathcal{M}_{\text{AccBot}}(T_1, T_2) = \mathcal{M}_{\text{IlstBot}}(T_1, T_2)$ についても同様の手法で示すことができる. 補題 3.3 より, $\mathcal{M}_{\text{AccSg}}(T_1, T_2) \subseteq \mathcal{M}_{\text{IlstSg}}(T_1, T_2)$ が成り立つので, 逆方向である $\mathcal{M}_{\text{IlstSg}}(T_1, T_2) \subseteq \mathcal{M}_{\text{AccSg}}(T_1, T_2)$ を示せばよい.

$M \in \mathcal{M}_{\text{IlstSg}}(T_1, T_2)$ を仮定し, $(u_1, v_1), (u_2, v_2), (u_3, v_3) \in M$ とする. このとき, $u_3 < u_1 \sqcup u_2 \iff v_3 < v_1 \sqcup v_2$ が成り立つ. また, $M \notin \mathcal{M}_{\text{AccSg}}(T_1, T_2)$ と仮定する. ここで, $u_1 \sqcup u_2 = u_1 \sqcup u_3$ であるが $v_1 \sqcup v_2 \neq v_1 \sqcup v_3$ となる $(u_1, v_1), (u_2, v_2), (u_3, v_3) \in M$ が存在する場合を考える.

$u_1 \# u_2$, $u_2 \# u_3$, $u_3 \# u_1$ と仮定する. M はマッピングであるため, $v_1 \# v_2$, $v_2 \# v_3$, $v_3 \# v_1$ が成り立つ. $v_1 \sqcup v_2 \neq v_1 \sqcup v_3$ と T_2 が木であることから, T_2 上の関係として (1) $v_1 \sqcup v_2 < v_1 \sqcup v_3$ (2) $v_1 \sqcup v_3 < v_1 \sqcup v_2$ のどちらか一方が成り立つ (図 3.2).

(1) の場合, $v_3 \not< v_1 \sqcup v_2$ が成り立つため, $u_3 < u_1 \sqcup u_2 \iff v_3 < v_1 \sqcup v_2$ と矛盾する. (2)

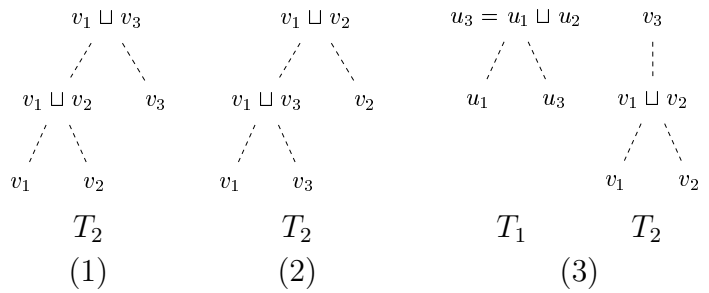


図 3.2: 定理 3.4 の証明に用いる木の模式図 (1), (2), (3).

の場合, $v_2 \not< v_1 \sqcup v_3$ が成り立つため, $u_2 < u_1 \sqcup u_3 \iff v_2 < v_1 \sqcup v_3$ と矛盾する. (この議論は断片マッピング以外でも成立する.)

$u_1 \# u_2$, $u_1 < u_3$, $u_2 < u_3$ と仮定する. $u_1 \sqcup u_2 = u_1 \sqcup u_3$ より, T_1 上で $u_1 \sqcup u_2 = u_3$ が成り立つ. 一方, $v_1 \sqcup v_2 \neq v_1 \sqcup v_3$ より, T_2 上で $v_1 \sqcup v_2 < v_1 \sqcup v_3 = v_3$ が成り立つ (図 3.2 の (3)).

M は断片マッピングなので, $(u', v_1 \sqcup v_2) \in M$ を満たすノード $u' \in T_1$ が存在する. $v_1 \sqcup v_2 < v_3$ より, $u' < u_3 = u_1 \sqcup u_2$ が成り立つ. 一方, $v_1 < v_1 \sqcup v_2$, $v_2 < v_1 \sqcup v_2$ より, $u_1 < u'$, $u_2 < u'$ が成り立つが, これはすなわち, $u_3 = u_1 \sqcup u_2 \leq u'$ ということであり, 矛盾する. 上記の議論もまた, $v_1 \sqcup v_2 = v_1 \sqcup v_3$ であるが $u_1 \sqcup u_2 \neq u_1 \sqcup u_3$ となるような $(u_1, v_1), (u_2, v_2), (u_3, v_3) \in M$ が存在する場合にも成り立つ.

ゆえに, 任意の $(u_1, v_1), (u_2, v_2), (u_3, v_3) \in M$ に対して, $u_1 \sqcup u_2 = u_1 \sqcup u_3 \iff v_1 \sqcup v_2 = v_1 \sqcup v_3$ を満たし, $M \in \mathcal{M}_{\mathrm{AccSg}}(T_1, T_2)$ が成り立つ.

他の関係式を示すためには, 真の包含関係を示す必要がある.

包含関係については, 補題 3.3 より, $\mathcal{M}_{\mathrm{Top}}(T_1, T_2) \subseteq \mathcal{M}_{\mathrm{LcaRt}}(T_1, T_2)$ と $\mathcal{M}_{\mathrm{Top}}(T_1, T_2) \subseteq \mathcal{M}_{\mathrm{LcaSg}}(T_1, T_2)$ を示せば充分である. 任意の $M \in \mathcal{M}_{\mathrm{Top}}(T_1, T_2)$ に対して, $(r(T_1), r(T_2)) \in M$ であること, および $\mathcal{M}_{\mathrm{Top}}(T_1, T_2) \subset \mathcal{M}_{\mathrm{Lca}}(T_1, T_2)$ であることから, $M \in \mathcal{M}_{\mathrm{LcaRt}}(T_1, T_2)$ が成り立つ. $\mathcal{M}_{\mathrm{Top}}(T_1, T_2) = \mathcal{M}_{\mathrm{TopSg}}(T_1, T_2)$ [22] と, $\mathcal{M}_{\mathrm{Top}}(T_1, T_2) \subseteq \mathcal{M}_{\mathrm{Lca}}(T_1, T_2)$ から,

補題 3.3.4 より, $\mathcal{M}_{\mathrm{Top}}(T_1, T_2) \subseteq \mathcal{M}_{\mathrm{LcaSg}}(T_1, T_2)$ が成り立つ.

一方, 真の包含関係については, 図 3.3 のマッピング M_i $(1 \leq i \leq 9)$ を用いて示すことができる.

2. $M_1 \in \mathcal{M}_{\mathrm{LcaSg}}(T_1, T_2)$ だが, $M_1 \notin \mathcal{M}_{\mathrm{Top}}(T_1, T_2)$ かつ $M_1 \notin \mathcal{M}_{\mathrm{LcaRt}}(T_1, T_2)$; $M_2 \in \mathcal{M}_{\mathrm{LcaRt}}(T_1, T_2)$ だが, $M_2 \notin \mathcal{M}_{\mathrm{Top}}(T_1, T_2)$ かつ $M_2 \notin \mathcal{M}_{\mathrm{LcaSg}}(T_1, T_2)$; $M_3 \in \mathcal{M}_{\mathrm{Lca}}(T_1, T_2)$ だが, $M_3 \notin \mathcal{M}_{\mathrm{LcaSg}}(T_1, T_2)$ かつ $M_3 \notin \mathcal{M}_{\mathrm{LcaRt}}(T_1, T_2)$ である.

3. $M_4 \in \mathcal{M}_{\mathrm{AccSg}}(T_1, T_2)$ だが $M_4 \notin \mathcal{M}_{\mathrm{LcaSg}}(T_1, T_2)$; $M_5 \in \mathcal{M}_{\mathrm{Acc}}(T_1, T_2)$ だが $M_5 \notin \mathcal{M}_{\mathrm{AccSg}}(T_1, T_2)$; $M_6 \in \mathcal{M}_{\mathrm{SgAln}}(T_1, T_2)$ だが $M_6 \notin \mathcal{M}_{\mathrm{AccSg}}(T_1, T_2)$ である.

4. $M_3 \in \mathcal{M}_{\mathrm{Lca}}(T_1, T_2)$ だが $M_3 \notin \mathcal{M}_{\mathrm{AccSg}}(T_1, T_2)$; $M_4 \in \mathcal{M}_{\mathrm{AccSg}}(T_1, T_2)$ だが $M_4 \notin \mathcal{M}_{\mathrm{Lca}}(T_1, T_2)$ である.

5. $M_7 \in \mathcal{M}_{\mathrm{Sg}}(T_1, T_2)$ だが $M_7 \notin \mathcal{M}_{\mathrm{SgAln}}(T_1, T_2)$ である. また, $M_7 \in \mathcal{M}_{\mathrm{Bot}}(T_1, T_2)$ だが $M_7 \notin \mathcal{M}_{\mathrm{BotAln}}(T_1, T_2)$ である.

6. $M_8 \in \mathcal{M}_{\mathrm{SgAln}}(T_1, T_2)$ だが $M_8 \notin \mathcal{M}_{\mathrm{BotAln}}(T_1, T_2)$.

7. $M_9 \in \mathcal{M}_{\mathrm{AccBot}}(T_1, T_2)$ だが $M_9 \notin \mathcal{M}_{\mathrm{LcaBot}}(T_1, T_2)$; $M_6 \in \mathcal{M}_{\mathrm{BotAln}}(T_1, T_2)$ だが $M_6 \notin \mathcal{M}_{\mathrm{AccBot}}(T_1, T_2)$; $M_7 \in \mathcal{M}_{\mathrm{Bot}}(T_1, T_2)$ だが $M_7 \notin \mathcal{M}_{\mathrm{BotAln}}(T_1, T_2)$ である. □

次に, 図 3.1 の Tai マッピング階層を特徴づけるために, 2 つの木の共通部分森を導入する.

定義 3.5 (部分森, 共通部分森)**.** $T = (V, E)$ を木とする. また, $I : V' \to V$ とする.

1. 任意の $u, v \in V'$ に対して, $I(u) < I(v)$ であって, $I(u) < I(w)$ かつ $I(w) < I(v)$ となる $w \in V'$ が存在しないときに $(u, v) \in E'$ となるような $F = (V', E')$ を, T の**埋め込み部分森** (*embedded subforest*) という.

2. T の埋め込み部分森 $F = (V', E')$ に対して, $(I(u), I(v)) \in E$ となる任意の F のノード $u, v \in V'$ について $(u, v) \in E'$ が成り立つとき, F を T の**誘導部分森** (*induced*

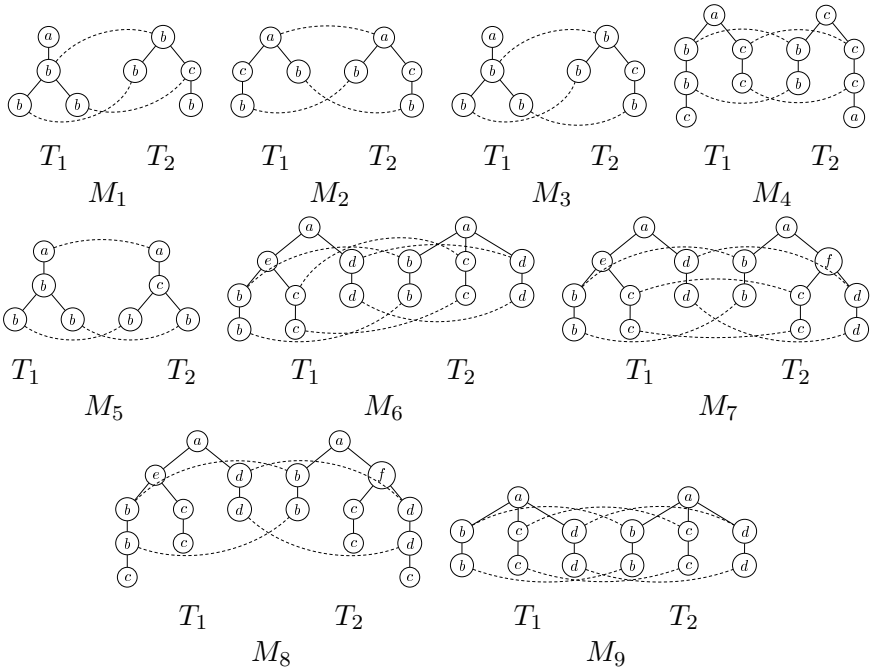


図 3.3: 定理 3.4 の証明に用いるマッピング M_i $(1 \leq i \leq 9)$.

subforest) という.

3. T の誘導部分森 $F = (V', E')$ が, 任意の F のノード $v \in V'$ について, その子孫がすべて V' の要素になっているとき, F を T の**完全部分森** (*complete subforest*) という.

さらに, F が木 T_1 と T_2 の埋め込み (誘導, 完全) 部分森であるとき, F を T_1 と T_2 の共通埋め込み (誘導, 完全) 部分森であるという. これら 3 つの区別が必要ないとき, これらを**共通部分森** (*common subforest*) という.

定義 3.6 (共通部分森中の部分木の並び)**.** F を T_1 と T_2 の共通部分森とする. $v \in F$ に対して, $v \in (T_1 \cap F)$ と $v \in (T_1 \cap F)$ を, それぞれ v^1, v^2 と表記する.

1. T_1 で $u^1 \sqcup v^1 < v^1 \sqcup w^1$ となり, かつ, T_2 で $v^2 \sqcup w^2 < u^2 \sqcup v^2$ となるノード $u, v, w \in F$ が存在するとき, F を**ねじれている** (*twisting*) といい, そうでないとき**ねじれていない** (*non-twisting*) という.

2. 任意のノード $u, v, w \in F$ について, T_1 で $u^1 \sqcup v_1 = u^1 \sqcup w^1$ となり, かつ, その場合

に限り, T_2 で $u^2 \sqcup v_2 = u^2 \sqcup w^2$ となるとき, F を**並列である** (*prallel*) という.

3. F が木であるとき, F を**部分木** (*subtree*) という.

4. F が木であり, $r(F) = r(T_1) = r(T_2)$ であるとき, F を**根保存部分木** (*root-preserving subtree*) という.

注意 3.7. T_1 と T_2 の共通部分森をマッピング M によって特徴づけることができる. I_M と J_M は, それぞれ T_1 と T_2 のノードのうち, マッピング M に含まれないノードの集合である. このとき, $T_1 = (V_1, E_1)$ と $T_2 = (V_2, E_2)$ に対して, M から, $F' = V_1 \setminus I_M = V_2 \setminus J_M$ となるような共通部分森 $F' = (V', E')$ が得られる. このような共通部分森を M **による共通部分森** (*common subforest through* M) という.

断片マッピングは親子関係を保存する必要があるため, 断片マッピングおよびその部分マッピング (Sg, SgAln, AccSg, LcaSg, Top) による共通部分森は誘導部分森となる. 操作的には, 削除と挿入を葉あるいは根のみに適用することに対応する. 特に Top では, 削除と挿入を葉にのみ適用する.

ボトムアップマッピングは完全部分木を保存する必要があるため, ボトムアップマッピングおよびその部分マッピング (Bot, BotAln, AccBot, LcaBot, Iso) による共通部分森は完全部分森となる. 操作的には, 削除と挿入を根のみに適用することに対応する.

注意 3.8. LCA 保存マッピングは含まれるノードすべてに対してその最近共通先祖がマッピングに含まれるため, LCA 保存マッピングおよびその部分マッピング (Lca, LcaSg, LcaBot, LcaRt, Top, Iso) による共通部分森は部分木となる. 特に, LCA 保存根保存マッピングは与えられた木の根の組を含むため, LCA 保存根保存マッピングおよびその部分マッピング (LcaRt, Top, Iso) による共通部分森は根保存部分木になる.

孤立部分木マッピングの定義は共通部分森が並列である条件と同じであるため, 孤立部分木マッピングおよびその部分マッピング (Acc, AccSg, AccBot) による共通部分森

は並列になる.

ねじれている共通部分森と同様に, 木T_1, T_2間のマッピングに関しても, ねじれを定義することができる. $(u_1 \sqcup u_2 < u_2 \sqcup u_3) \wedge (v_2 \sqcup v_3 < v_1 \sqcup v_2)$ を満たす組 $(u_1, v_1), (u_2, v_2), (u_3, v_3) \in M$ が存在するとき, M はねじれているといい, 組 $(u_1, v_1), (u_2, v_2), (u_3, v_3)$ を, M の**ねじれ** (*twist*) という.

ノード $v_1, v_2, v_3 \in T$ に対して, $\neg(v_2 \sqcup v_3 < v_1 \sqcup v_2) \equiv (v_1 \sqcup v_2 \leq v_2 \sqcup v_3)$ と $(v_1 \sqcup v_2 \leq v_2 \sqcup v_3) \equiv (v_1 \sqcup v_3 = v_2 \sqcup v_3)$ が常に成り立つので, 図 3.4 の通りにねじれの非存在の式を変形することができる. よって, マッピング M が劣制限マッピングである場合に限り M はねじれを持たないことを論理式の変形によって示すことができる. したがって, アライメント可能マッピングおよびその部分マッピング (Aln, SgAln, BotAln) による共通部分森はねじれを持たない.

$$
\begin{aligned}
&\neg\exists(u_1, v_1)(u_2, v_2)(u_3, v_3) \in M\big((u_1 \sqcup u_2 < u_2 \sqcup u_3) \wedge (v_2 \sqcup v_3 < v_1 \sqcup v_2)\big)\\
\equiv&\forall(u_1, v_1)(u_2, v_2)(u_3, v_3) \in M\neg\big((u_1 \sqcup u_2 < u_2 \sqcup u_3) \wedge (v_2 \sqcup v_3 < v_1 \sqcup v_2)\big)\\
\equiv&\forall(u_1, v_1)(u_2, v_2)(u_3, v_3) \in M\big(\neg(u_1 \sqcup u_2 < u_2 \sqcup u_3) \vee \neg(v_2 \sqcup v_3 < v_1 \sqcup v_2)\big)\\
\equiv&\forall(u_1, v_1)(u_2, v_2)(u_3, v_3) \in M\big((u_1 \sqcup u_2 < u_2 \sqcup u_3) \Rightarrow \neg(v_2 \sqcup v_3 < v_1 \sqcup v_2)\big)\\
\equiv&\forall(u_1, v_1)(u_2, v_2)(u_3, v_3) \in M\big((u_1 \sqcup u_2 < u_2 \sqcup u_3) \Rightarrow (v_1 \sqcup v_2 \leq v_2 \sqcup v_3)\big)\\
\equiv&\forall(u_1, v_1)(u_2, v_2)(u_3, v_3) \in M\big((u_1 \sqcup u_2 < u_2 \sqcup u_3) \Rightarrow (v_1 \sqcup v_3 = v_2 \sqcup v_3)\big).
\end{aligned}
$$

図 3.4: ねじれに関する式変形

このことをまとめて共通部分森の視点から Tai マッピング階層を整理すると図 3.5 のようになる.

定義 3.9 (編集距離の変種)**.** 図 3.5 で示した階層の各マッピング $\mathcal{M}_{\mathtt{A}}(T_1, T_2)$($\mathtt{A} \in \{$IlstSg, AccSg, LcaSg, SgAln, BotAln, IlstBot, AccBot, LcaBot, LcaRt$\}$) に対し, 編集距離 τ_{TAI} の変種 $\tau_{\mathtt{A}}(T_1, T_2)$ を, $\mathcal{M}_{\mathtt{A}}(T_1, T_2)$ に含まれるマッピングの最小コストとして定義する:

$$\tau_{\mathtt{A}}(T_1, T_2) = \min\{\gamma(M) \mid M \in \mathcal{M}_{\mathtt{A}}(T_1, T_2)\}.$$

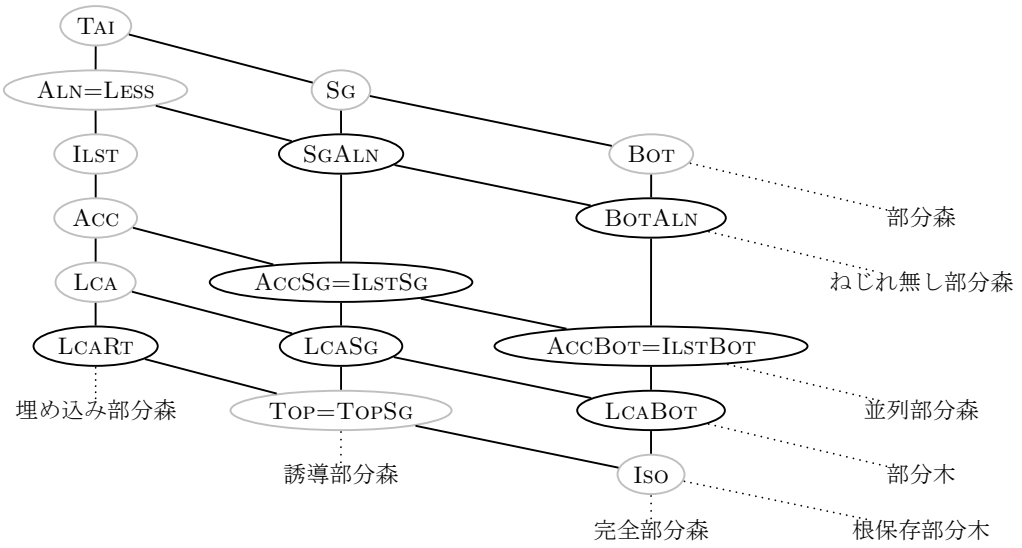


図 3.5: 共通部分森の観点から整理した Tai マッピング階層.

特に, 順序木, 無順序木を区別する場合は, 順序木間の $\tau_{\mathtt{A}}(T_1, T_2)$ を $\tau^{o}_{\mathtt{A}}(T_1, T_2)$, 無順序木間の $\tau_{\mathtt{A}}(T_1, T_2)$ を $\tau^{u}_{\mathtt{A}}(T_1, T_2)$ と表す.

注意 3.10. コスト関数がメトリックである場合, $\tau_{\mathrm{LcaRt}}(T_1, T_2) = \tau_{\mathrm{Lca}}(T_1, T_2) = \tau_{\mathrm{Acc}}(T_1, T_2)$ が成り立つ ([29] 参照). 一方, 図 3.6 の木 T_1, T_2 を考える. このとき, 単一コスト関数 γ の下で, 図 3.6 に示す最小コストマッピング $M_1 \in \mathcal{M}_{\mathrm{AccSg}}(T_1, T_2)$ と最小コストマッピング $M_2 \in \mathcal{M}_{\mathrm{LcaSg}}(T_1, T_2)$ を得る. ゆえに, $\tau_{\mathrm{AccSg}}(T_1, T_2) = \gamma(M_1) = 4 < 6 = \tau_{\mathrm{LcaSg}}(T_1, T_2) = \gamma(M_2)$ が成り立つ.

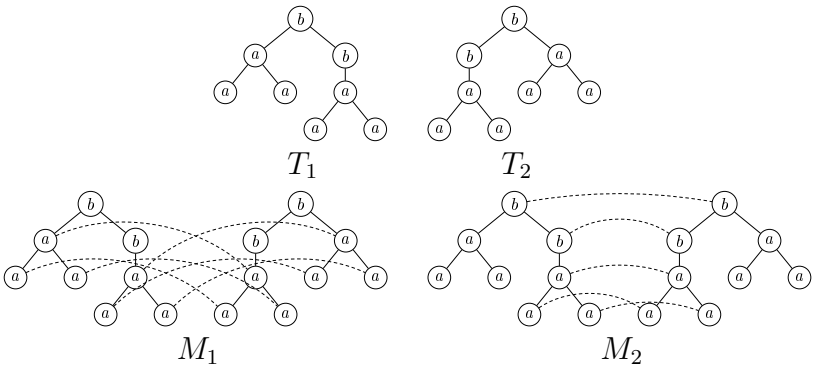


図 3.6: 例 3.10 の木 T_1, T_2 とマッピング M_1, M_2.

τ_{TAI}, τ_{ILST}, $\tau_{\mathrm{LCA}}(=\tau_{\mathrm{ACC}})$, τ_{TOP}, τ_{SG}, τ_{BOT} はメトリックであるが, τ_{ALN} はメトリックではないことが知られている [22, 29]. 本節の残りでは, 新たに導入したマッピング $\mathtt{A} \in \{\mathrm{SGALN}, \mathrm{ACCSG}, \mathrm{LCASG}, \mathrm{BOTALN}, \mathrm{ACCBOT}, \mathrm{LCABOT}, \mathrm{LCART}\}$ に対応する距離 $\tau_{\mathtt{A}}$ がメトリックであるか否かを確認する.

$M_i \in \mathcal{M}_{\mathtt{A}}(T_i, T_{i+1})(i = 1, 2)$ とする. このとき, マッピング M_1 と M_2 の**合成** (*composition*) $M_1 \circ M_2$ を以下のように定義する.

$$M_1 \circ M_2 = \left\{ \begin{array}{l|l} (u, w) & \exists v \in V(T_2) \\ \quad \in V(T_1) \times V(T_3) & \text{s.t. } (u, v) \in M_1 \text{かつ } (v, w) \in M_2 \end{array} \right\}.$$

補題 3.11 ([22, 29])**.** $\mathtt{A} \in \{\mathrm{TAI}, \mathrm{ILST}, \mathrm{ACC}, \mathrm{LCA}, \mathrm{TOP}, \mathrm{SG}, \mathrm{BOT}\}$ とし, $M_i \in \mathcal{M}_{\mathtt{A}}(T_i, T_{i+1})$ $(i = 1, 2)$ とする. このとき, 以下が成り立つ.

1. $M_1 \circ M_2 \in \mathcal{M}_{\mathtt{A}}(T_1, T_3)$.
2. メトリックなコスト関数 γ に対して, $\gamma(M_1 \circ M_2) \leq \gamma(M_1) + \gamma(M_2)$.

補題 3.12. $\mathcal{M}_{\mathtt{A}}$ と $\mathcal{M}_{\mathtt{B}}$ が補題 3.11 の関係式を満たすとき, $\mathcal{M}_{\mathtt{AB}}$ も同様に関係式を満たす.

[**証明**]. $M_i \in \mathcal{M}_{\mathtt{AB}}(T_i, T_{i+1})(i = 1, 2)$ とする. 定義より $\mathcal{M}_{\mathtt{AB}}(T_i, T_{i+1}) = \mathcal{M}_{\mathtt{A}}(T_i, T_{i+1}) \cap \mathcal{M}_{\mathtt{B}}(T_i, T_{i+1})$ であるため, $M_i \in \mathcal{M}_{\mathtt{A}}(T_i, T_{i+1})$ かつ $M_i \in \mathcal{M}_{\mathtt{B}}(T_i, T_{i+1})$ が成り立つ. 補題 3.11.1 より, $\mathcal{M}_{\mathtt{A}}$, $\mathcal{M}_{\mathtt{B}}$ に対して, $M_1 \circ M_2 \in \mathcal{M}_{\mathtt{A}}(T_1, T_3)$ と $M_1 \circ M_2 \in \mathcal{M}_{\mathtt{B}}(T_1, T_3)$ が成り立つ. ゆえに, $M_1 \circ M_2 \in \mathcal{M}_{\mathtt{AB}}(T_1, T_3)$ が成り立ち, $\mathcal{M}_{\mathtt{A}}$ は補題 3.11.1 を満たす. 補題 3.11.2 の証明は補題 3.11.1 にだけ依存するので, $\mathcal{M}_{\mathtt{AB}}$ に対しても補題 3.11.2 を満たす. □

定理 3.13 (メトリック性)**.** コスト関数がメトリックならば, τ_{ACCSG}, τ_{LCASG}, τ_{ACCBOT}, τ_{LCABOT} τ_{LCART} はメトリックである. 一方, コスト関数がメトリックであっても, τ_{SGALN} と τ_{BOTALN} はメトリックにならない.

[**証明**]. まず, τ_{AccSg} がメトリックであることを示す. これを示すには, $\tau_{\mathrm{AccSg}}(T_1, T_2) \geq 0$, $\tau_{\mathrm{AccSg}}(T_1, T_2) = 0 \Leftrightarrow T_1 \equiv T_2$, $\tau_{\mathrm{AccSg}}(T_1, T_2) = \tau_{\mathrm{AccSg}}(T_2, T_1)$, $\tau_{\mathrm{AccSg}}(T_1, T_3) \leq \tau_{\mathrm{AccSg}}(T_1, T_2) + \tau_{\mathrm{AccSg}}(T_2, T_3)$ を示せばよい. 最初の 3 つの関係式は $\mathcal{M}_{\mathrm{AccSg}}(T_1, T_2)$ の定義より明らかである. M_i を T_i, T_{i+1} 間の最小コスト調和的断片マッピングとする ($i = 1, 2$). 補題 3.11 および 3.12 により, $\tau_{\mathrm{AccSg}}(T_1, T_3) \leq \gamma(M_1 \circ M_2) \leq \gamma(M_1) + \gamma(M_2) = \tau_{\mathrm{AccSg}}(T_1, T_2) + \tau_{\mathrm{AccSg}}(T_2, T_3)$ が成り立つ. 同様の手法で, τ_{LcaSg}, τ_{AccBot}, τ_{LcaBot} がメトリックであることも示すことができる. τ_{Lca} がメトリックであるので, τ_{LcaRt} も明らかにメトリックとなる.

次に, τ_{SgAln} と τ_{BotAln} がメトリックでないことを示す. 図 3.7 に示す木 T_1, T_2, T_3 を考える. また, γ を単一コスト関数とする. このとき, γ 下での最小コストマッピング $M_{12} \in \mathcal{M}_{\mathrm{BotAln}}(T_1, T_2)$, $M_{13} \in \mathcal{M}_{\mathrm{BotAln}}(T_1, T_3)$, $M_{23} \in \mathcal{M}_{\mathrm{BotAln}}(T_2, T_3)$ を図 3.7 のように得る.

ゆえに, $8 = \gamma(M_{12}) = \tau_{\mathrm{BotAln}}(T_1, T_2) > \tau_{\mathrm{BotAln}}(T_1, T_3) + \tau_{\mathrm{BotAln}}(T_2, T_3) = \gamma(M_{13}) + \gamma(M_{23}) = 3 + 3 = 6$ が成り立つ. 同様のことが τ_{SgAln} にも言えるため, τ_{SgAln}, τ_{BotAln} は三角不等式を満たさず, これらはメトリックでないといえる. □

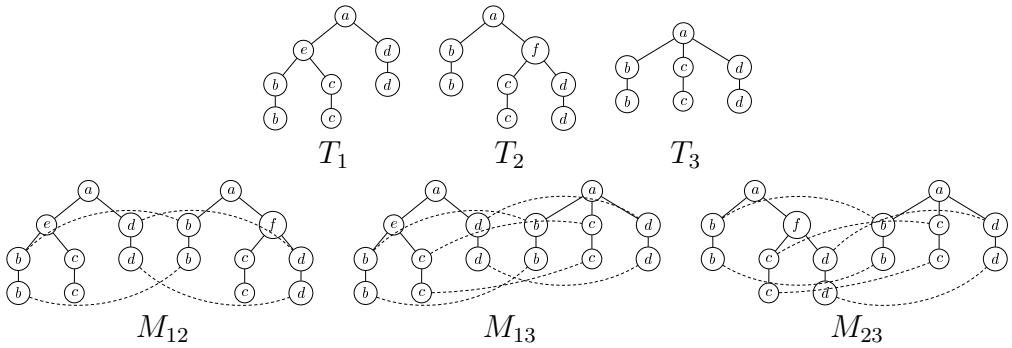


図 3.7: 定理 3.13 の証明に用いる木 T_1, T_2, T_3 とマッピング M_{12}, M_{13}, M_{23}.

3.2　多項式時間計算可能な木編集距離の変種

本節および次節では, 順序木, 無順序木それぞれに対して, 編集距離の変種の計算の時間計算量について議論する. 以降, ノードを先行走査順の数字で表す. 本節および次節では $n = |T_1|$, $m = |T_2|$, $D = \max\{d(T_1), d(T_2)\}$, $d = \min\{d(T_1), d(T_2)\}$ とする. また, マッピング $\mathtt{A}$ に対する順序木間の距離を $\tau^o_{\mathtt{A}}$ で表し, 順序森間の距離を $\delta^o_{\mathtt{A}}$ で表す. 同様に, マッピング $\mathtt{A}$ に対する無順序木間の距離を $\tau^u_{\mathtt{A}}$ で表し, 無順序森間の距離を $\delta^u_{\mathtt{A}}$ で表す.

本節では, 多項式時間計算可能な木編集距離の変種について取り扱う.

注意 3.14. 任意の組 $(i, j) \in T_1 \times T_2 (1 \leq i \leq n, 1 \leq j \leq m)$ に対して, $T_1[i]$, $T_2[j]$ 間のトップダウン距離 $\tau^o_{\mathrm{TOP}}(T_1[i], T_2[j])$ は $O(nm)$ 時間で計算が可能である [22].

一方, $\tau^u_{\mathrm{TOP}}(T_1[i], T_2[j])$ は, 図 3.8 に示す再帰式を用いることで計算ができる [51]. ここで, $\mathtt{BM}$ は, 完全二部グラフ $G_M = (X, Y, E)$ における, 重み ω に関する最大重み二部マッチングである. ただし, X は T_1 のノード i の子ノードの集合, Y は T_2 のノード j の子ノードの集合であり, 辺 $(i', j') \in E$ の重みは $\tau^u_{\mathrm{TOP}}(T_1[i'], \emptyset) + \tau^u_{\mathrm{TOP}}(\emptyset, T_2[j']) - \tau^u_{\mathrm{TOP}}(T_1[i'], T_2[j'])$ とする [51, 64]. ゆえに, すべての $(i, j) \in T_1 \times T_2$ に対して, $\tau^u_{\mathrm{TOP}}(T_1[i], T_2[j])$ を $O(nmd)$ 時間で計算することができる.

$$
\begin{aligned}
\tau^u_{\mathrm{TOP}}(T_1[i], T_2[j]) &= \delta^u_{\mathrm{TOP}}(T_1(i), T_2(j)) + \gamma(i, j), \\
\delta^u_{\mathrm{TOP}}(T_1(i), T_2(j)) &= \sum_{T \in T_1(i)} \sum_{i' \in T} \gamma(i', \varepsilon) + \sum_{T \in T_2(j)} \sum_{j' \in T} \gamma(\varepsilon, j') - \sum_{(i', j') \in \mathtt{BM}} \omega((i', j')).
\end{aligned}
$$

図 3.8: 根無し木のトップダウン距離 $\tau^u_{\mathrm{TOP}}(T_1, T_2)$ を計算する再帰式.

定理 3.15. $\tau^o_{\mathrm{LCARt}}(T_1, T_2)$ は $O(nm)$ 時間で, $\tau^u_{\mathrm{LCARt}}(T_1, T_2)$ は $O(nmd)$ 時間で計算可能である.

[**証明**]. τ_{LCARt} を計算する再帰式は, (根付き木, 根無し木ともに) τ_{LCA} を計算する再帰式に

$$
\tau_{\mathrm{LCARt}}(T_1[r_1], T_2[r_2]) = \delta_{\mathrm{LCA}}(T_1(r_1), T_2(r_2)) + \gamma(r_1, r_2)
$$

を加えることで設計できる [51, 64]. ここで, r_i は T_i の根ノード $(i = 1, 2)$ である. ゆえに, $\tau^o_{\mathrm{LCART}}(T_1, T_2)$ は $O(nm)$ 時間で, $\tau^u_{\mathrm{LCART}}(T_1, T_2)$ は $O(nmd)$ 時間で計算可能である. □

定理 3.16. $\tau^o_{\mathrm{LCASG}}(T_1, T_2)$ は $O(nm)$ 時間で, $\tau^u_{\mathrm{LCASG}}(T_1, T_2)$ は $O(nmd)$ 時間で計算可能である.

[**証明**]. すべての組 $(i, j) \in T_1 \times T_2$ に対して, $\tau^o_{\mathrm{TOP}}(T_1[i], T_2[j])$ に基づいた T_1, T_2 間の距離 $d(T_1, i, T_2, j)$ を以下のように定義する.

$$d(T_1, i, T_2, j) = \tau^o_{\mathrm{TOP}}(T_1[i], T_2[j]) + \sum_{i' \in T_1 - T_1[i]} \gamma(i', \varepsilon) + \sum_{j' \in T_2 - T_2[j]} \gamma(\varepsilon, j').$$

ここで,

$$\tau^o_{\mathrm{LCASG}}(T_1, T_2) = \min\{d(T_1, i, T_2, j) \mid 1 \le i \le n, 1 \le j \le m\}$$

が成り立つため, 注意 3.14 より, $\tau^o_{\mathrm{LCASG}}(T_1, T_2)$ は $O(nm)$ 時間で計算可能である. 同様に, 注意 3.14 より, $d(T_1, i, T_2, j)$ の $\tau^o_{\mathrm{TOP}}(T_1[i], T_2[j])$ を $\tau^u_{\mathrm{TOP}}(T_1[i], T_2[j])$ と置き換えることにより τ^u_{LCASG} を計算することができるため, $\tau^u_{\mathrm{LCASG}}(T_1, T_2)$ は $O(nmd)$ 時間で計算が可能である. □

定理 3.17. $\tau^o_{\mathrm{ACCSG}}(T_1, T_2)$ は $O(nm)$ 時間で, $\tau^o_{\mathrm{SGALN}}(T_1, T_2)$ は $O(nmD^2)$ 時間で計算可能である.

[**証明**]. 図 3.9 の τ^o_{ACCSG} と τ^o_{SGALN} を計算する再帰式を考える. τ^o_{ACCSG} を計算する再帰式と τ^o_{ACC} を計算する再帰式 [29] との違いは, τ^o_{ACCSG} の最初の 2 つの式である. また, τ^o_{SGALN} を計算する再帰式と τ^o_{ALN} を計算する再帰式 [21] との違いは, τ^o_{SGALN} の最初の式である. τ^o_{ACCSG} と τ^o_{SGALN} の最初の式は, $(i, j) \in T_1 \times T_2$ の断片に対する距離を計算している. τ^o_{ACCSG} の 2 つ目の式では, τ^o_{ACC} 中の $\gamma(i, j)$ を $\gamma(i, \varepsilon) + \gamma(\varepsilon, j)$ に置き換えている. これらは (注意 3.14 で指摘したように, 事前に $\tau^o_{\mathrm{TOP}}(T_1[i], T_2[j])$ を計算することによって) 定数時間で計算ができるため, [29], [21] での議論と同様にして, 定理の計算時間を得る. □

$$
\begin{aligned}
&\tau^o_{\mathrm{AccSG}}(T_1[i], T_2[j]) \\
&\quad = \min \left\{ \begin{array}{l}
\tau^o_{\mathrm{TOP}}(T_1[i], T_2[j]) \cdots \text{事前に計算した値を用いる}, \\
\delta^o_{\mathrm{AccSG}}(T_1(i), T_2(j)) + \gamma(i, \varepsilon) + \gamma(\varepsilon, j), \\
\tau^o_{\mathrm{AccSG}}(T_1[i], \emptyset) + \min\limits_{T \in T_1(i)} \{\tau^o_{\mathrm{AccSG}}(T, T_2[j]) - \tau^o_{\mathrm{AccSG}}(T, \emptyset)\}, \\
\tau^o_{\mathrm{AccSG}}(\emptyset, T_2[j]) + \min\limits_{T \in T_2(j)} \{\tau^o_{\mathrm{AccSG}}(T_1[i], T) - \tau^o_{\mathrm{AccSG}}(\emptyset, T)\}
\end{array} \right\}, \\
&\delta^o_{\mathrm{AccSG}}(F_1(i_1, i_s), F_2(j_1, j_t)) \\
&\quad = \min \left\{ \begin{array}{l}
\delta^o_{\mathrm{AccSG}}(F_1(i_1, i_{s-1}), F_2(j_1, j_t)) + \tau^o_{\mathrm{AccSG}}(T_1[i_s], \emptyset), \\
\delta^o_{\mathrm{AccSG}}(F_1(i_1, i_s), F_2(j_1, j_{t-1})) + \tau^o_{\mathrm{AccSG}}(\emptyset, T_2[j_t]), \\
\delta^o_{\mathrm{AccSG}}(F_1(i_1, i_{s-1}), F_2(j_1, j_{t-1})) + \tau^o_{\mathrm{AccSG}}(T_1[i_s], T_2[j_t])
\end{array} \right\}. \\
&\tau^o_{\mathrm{SGALN}}(T_1[i], T_2[j]) \\
&\quad = \min \left\{ \begin{array}{l}
\tau^o_{\mathrm{TOP}}(T_1[i], T_2[j]) \cdots \text{事前に計算した値を用いる}, \\
\delta^o_{\mathrm{SGALN}}(T_1(i), T_2(j)) + \gamma(i, j), \\
\tau^o_{\mathrm{SGALN}}(T_1[i], \emptyset) + \min\limits_{T \in T_1(i)} \{\tau^o_{\mathrm{SGALN}}(T, T_2[j]) - \tau^o_{\mathrm{SGALN}}(T, \emptyset)\}, \\
\tau^o_{\mathrm{SGALN}}(\emptyset, T_2[j]) + \min\limits_{T \in T_2(j)} \{\tau^o_{\mathrm{SGALN}}(T_1[i], T) - \tau^o_{\mathrm{SGALN}}(\emptyset, T)\}
\end{array} \right\}, \\
&\delta^o_{\mathrm{SGALN}}(F_1(i_1, i_s), F_2(j_1, j_t)) \\
&\quad = \min \left\{ \begin{array}{l}
\delta^o_{\mathrm{SGALN}}(F_1(i_1, i_{s-1}), F_2(j_1, j_t)) + \tau^o_{\mathrm{SGALN}}(T_1[i_s], \emptyset), \\
\delta^o_{\mathrm{SGALN}}(F_1(i_1, i_s), F_2(j_1, j_{t-1})) + \tau^o_{\mathrm{SGALN}}(\emptyset, T_2[j_t]), \\
\delta^o_{\mathrm{SGALN}}(F_1(i_1, i_{s-1}), F_2(j_1, j_{t-1})) + \tau^o_{\mathrm{SGALN}}(T_1[i_s], T_2[j_t]), \\
\gamma(i_s, \varepsilon) + \min\limits_{1 \le k < t} \left\{ \begin{array}{c} \delta^o_{\mathrm{SGALN}}(F_1(i_1, i_{s-1}), F_2(j_1, j_{k-1})) \\ + \delta^o_{\mathrm{SGALN}}(T_1(i_s), F_2(j_k, j_t)) \end{array} \right\}, \\
\gamma(\varepsilon, j_t) + \min\limits_{1 \le k < s} \left\{ \begin{array}{c} \delta^o_{\mathrm{SGALN}}(F_1(i_1, i_{k-1}), F_2(j_1, j_{t-1})) \\ + \delta^o_{\mathrm{SGALN}}(F_1(i_k, i_s), T_2(j_t)) \end{array} \right\}
\end{array} \right\}.
\end{aligned}
$$

図 3.9: 順序木の距離 $\tau^o_{\mathrm{AccSG}}(T_1, T_2)$, $\tau^o_{\mathrm{SGALN}}(T_1, T_2)$ を計算する再帰式.

定理 3.18. $\tau^u_{\mathrm{AccSG}}(T_1, T_2)$ は $O(nmd)$ 時間で計算可能である.

[**証明**]. 注意 3.14 の図 3.8 に示す再帰式に従って, すべての $\tau^u_{\mathrm{TOP}}(T_1(i), T_2(j))((i, j) \in T_1 \times T_2)$ を $O(nmd)$ 時間で格納した後, 図 3.10 の再帰式を使うことによって, $\tau^u_{\mathrm{AccSG}}(T_1, T_2)$ を $O(nmd)$ 時間で計算することができる. □

次に, ボトムアップ距離 τ_{BOT} の変種について議論する.

注意 3.19. 木 T_1, T_2 に対して, 木 T_1 と T_2 が順序木 (無順序木) 同型であるとき, 特にラベルを考慮しない同型であることを強調して, **ラベルなし順序木同型** (*label-free ordered tree isomorphic*) (**ラベルなし無順序木同型** (*label-free unordered tree isomorphic*)) であるといい, これを $T_1 \equiv^o_l T_2$ ($T_1 \equiv^u_l T_2$) で表す. さらに, $diff^r(T_1, T_2)(r \in \{o, u\})$ を以下のように

$$
\begin{aligned}
&\tau^u_{\mathrm{AccSg}}(T_1[i], T_2[j]) \\
&\quad = \min \left\{ \begin{array}{l}
\tau^u_{\mathrm{Top}}(T_1[i], T_2[j]), \\
\delta^u_{\mathrm{AccSg}}(T_1(i), T_2(j)) + \gamma(i, \varepsilon) + \gamma(\varepsilon, j), \\
\tau^u_{\mathrm{AccSg}}(T_1[i], \emptyset) + \min\limits_{T \in T_1(i)} \{\tau^u_{\mathrm{AccSg}}(T, T_2[j]) - \tau^u_{\mathrm{AccSg}}(T, \emptyset)\}, \\
\tau^u_{\mathrm{AccSg}}(\emptyset, T_2[j]) + \min\limits_{T \in T_2(j)} \{\tau^u_{\mathrm{AccSg}}(T_1[i], T) - \tau^u_{\mathrm{AccSg}}(\emptyset, T)\}
\end{array} \right\}, \\
&\delta^u_{\mathrm{AccSg}}(T_1(i), T_2(j)) \\
&\quad = \min \left\{ \begin{array}{l}
\delta^u_{\mathrm{AccSc}}(T_1(i), \emptyset) + \min\limits_{1 \le i' \le d(i)} \{\delta^u_{\mathrm{AccSc}}(T_1(i'), T_2(j)) - \delta^u_{\mathrm{AccSc}}(T_1(i'), \emptyset)\}, \\
\delta^u_{\mathrm{AccSc}}(\emptyset, T_2(j)) + \min\limits_{1 \le j' \le d(j)} \{\delta^u_{\mathrm{AccSc}}(T_1(i), T_2(j')) - \delta^u_{\mathrm{AccSc}}(\emptyset, T_2(j'))\}, \\
\delta^u_{\mathrm{Top}}(T_1(i), T_2(j))
\end{array} \right\}.
\end{aligned}
$$

図 3.10: 無順序木の距離 $\tau^u_{\mathrm{AccSg}}(T_1, T_2)$ を計算する再帰式.

定義する. ここで, id は T_1, T_2 間のラベルなし同型写像である.

$$
\mathit{diff}^r(T_1, T_2) \quad = \quad \left\{ \begin{array}{ll}
\displaystyle\sum_{(i,j) \in id} \gamma(i, j) & (T_1 \equiv^r_l T_2), \\
\displaystyle\sum_{i \in T_1} \gamma(i, \varepsilon) + \sum_{j \in T_2} \gamma(\varepsilon, j) & (T_1 \not\equiv^r_l T_2).
\end{array} \right.
$$

$T_1 \equiv^r_l T_2$ であるかどうかの確認と, $\mathit{diff}^r(T_1, T_2)$ の計算は, ともに $O(n + m)$ 時間で可能である [47].

定理 3.20. $\tau^o_{\mathrm{LcaBot}}(T_1, T_2)$ および $\tau^u_{\mathrm{LcaBot}}(T_1, T_2)$ は, $O(nm)$ 時間で計算可能である.

[**証明**]. $r \in \{o, u\}$ に対して, 定理 3.16 の $d(T_1, i, T_2, j)$ 中にある $\tau^r_{\mathrm{Top}}(T_1[i], T_2[j])$ を $\mathit{diff}^r(T_1[i], T_2[j])$ と置き換えることと注意 3.19 により, $\tau^r_{\mathrm{LcaBot}}(T_1, T_2)$ は $O(nm)$ 時間で計算可能である. □

定理 3.21. $\tau^o_{\mathrm{AccBot}}(T_1, T_2)$ は $O(nm)$ 時間で, $\tau^o_{\mathrm{BotAln}}(T_1, T_2)$ は $O(nmD^2)$ 時間で計算可能である.

[**証明**]. 注意 3.19 より, 図 3.9 に示す τ^o_{AccSg} と τ^o_{SgAln} を計算する再帰式の $\tau^o_{\mathrm{Top}}(T_1[i], T_2[j])$ を $\mathit{diff}^o(T_1[i], T_2[j])$ に置き換えることで, τ^o_{AccBot} と τ^o_{BotAln} をそれぞれ $O(nm)$ 時間と $O(nmD^2)$ 時間で計算可能である. □

定理 3.22. $\tau^u_{\mathrm{AccBot}}(T_1, T_2)$ は $O(nmd)$ 時間で計算可能である.

[**証明**]. 注意 3.19 無順序木に対して定義された関数 diff^u を, 森に対する関数 fdiff^u に拡張する. $F_1 = [S_1, \ldots, S_k]$, $F_2 = [T_1, \ldots, T_l]$ を森とする. まず, $X = \{1, \ldots, k\}$, $Y = \{1, \ldots, l\}$ に対して, $S_i \equiv_l^u T_j$ であるとき, かつそのときに限り $(i, j) \in E$ となり, 重みが $\omega((i, j)) = |S_i| + |T_j| - \mathit{diff}^u(S_i, T_j)$ となるような重み付き二部グラフ $G = (X, Y, E)$ を構築する. また $\mathtt{BM} \subseteq X \times Y$ を G の最大重み二部マッチングとし, $\mathtt{BM}_X^- = \{i \in X \mid (i, j) \notin \mathtt{BM}\}$, $\mathtt{BM}_Y^- = \{j \in Y \mid (i, j) \notin \mathtt{BM}\}$ とする. このとき, $\mathit{fdiff}^u(F_1, F_2)$ を以下の通りに定義する.

$$\mathit{fdiff}^u(F_1, F_2) = \sum_{(i,j)\in\mathtt{BM}} \mathit{diff}^u(S_i, T_j) + \sum_{i\in\mathtt{BM}_X^-} |S_i| + \sum_{j\in\mathtt{BM}_X^-} |T_j|.$$

$\mathit{fdiff}^u(F_1, F_2)$ は明らかに F_1 と F_2 ラベルなし同型の木の組を選ぶときに最小値を取り, これは $\delta^u_{\mathrm{AccBot}}(F_1, F_2)$ と等価である. したがって, 図 3.10 の $\tau^u_{\mathrm{Top}}(T_1[i], T_2[j])$, $\delta^u_{\mathrm{Top}}(T_1(i), T_2(j))$ を $\mathit{diff}^u(T_1[i], T_2[j])$, $\mathit{fdiff}^u(T_1(i), T_2(j))$ に置き換えることにより, $\tau^u_{\mathrm{AccBot}}(T_1, T_2)$ を $O(nmd)$ 時間で計算することができる. □

3.3　多項式時間近似困難な木編集距離の変種

本節では, 多項式時間近似困難な木編集距離の変種について議論する.

Π_1 と Π_2 を最適化問題とする. このとき, Π_1 の問題例 I に対して以下の式を満たすような多項式時間アルゴリズム f, g と定数 $\alpha, \beta > 0$ が存在するとき, Π_1 は Π_2 に L **還元可能** (L-*reducible*) という.

1. $\mathit{opt}(f(I)) \leq \alpha \cdot \mathit{opt}(I)$.
2. 重みが s_2 の $f(I)$ の解に対して, アルゴリズム g は, 多項式時間で $|s_1 - \mathit{opt}(I)| \leq \beta \cdot |s_2 - \mathit{opt}(f(I))|$ を満たす重みが s_1 の I の解を与える.

もし Π_1 が Π_2 に L 還元可能であり, Π_2 が多項式時間で最適解の $1+\varepsilon$ 倍で近似可能ならば,

Π_1 は多項式時間で最適解の $1+\alpha\beta\varepsilon$ 倍で近似可能である. また, もし Π_2 が**多項式時間近似スキーム** (*polynomial time approximation scheme*)(PTAS) を持つなら, Π_1 も PTAS を持つ [36].

最適化問題は, 任意の MAX SNP の問題がその問題に L 還元可能であるとき **MAX SNP 困難** (MAX SNP-*hard*) という. 2 つの L 還元の組み合わせもまた L 還元であるため, ある問題が MAX SNP 困難であるためには, MAX SNP 困難な問題から, その問題へ L 還元可能であればよい. 任意の MAX SNP 困難な問題が PTAS を持つなら, P = NP であることが知られている. ゆえに, MAX SNP 困難な問題が PTAS を持つことはほとんどない [36].

Zhang と Jiang [61] は, $\tau^u_{\mathrm{TAI}}(T_1,T_2)$ の計算問題が MAX SNP 困難であることを最初に示した. Akutsu ら [2] と Hirata ら [18] は, それぞれ, たとえ T_1 と T_2 の高さが高々2 であったり, 二分木であったとしても, $\tau^u_{\mathrm{TAI}}(T_1,T_2)$ の計算問題は MAX SNP 困難であることを示した. さらに, Yamamoto ら [51] は, たとえ T_1 と T_2 が二分木であっても $\tau^u_{\mathrm{BOT}}(T_1,T_2)$ の計算問題が MAX SNP 困難であることを示した. このことから, $\tau^u_{\mathrm{SG}}(T_1,T_2)$ の計算問題も MAX SNP 困難となる.

ここで, これらの証明 [2, 18, 51, 61] で用いられているマッピングはアライメント可能マッピングではないため, $\tau^u_{\mathrm{SGALN}}(T_1,T_2)$ および $\tau^u_{\mathrm{BOTALN}}(T_1,T_2)$ の計算問題が MAX SNP 困難であることを示すために直接利用することはできないことに注意する. したがって本論文では, 以下の問題 MAX 3SC-3 からの L 還元により, $\tau^u_{\mathrm{SGALN}}(T_1,T_2)$ および $\tau^u_{\mathrm{BOTALN}}(T_1,T_2)$ の計算問題が MAX SNP 困難であることを, 文献 [51] と類似した方法で証明する:

> Maximum Bounded Covering by 3-Sets (MAX 3SC-3) [24]:
> 問題例: 有限集合 S, および, 3 つの要素からなる S の部分集合の集合族 $\mathcal{C}$. ただし, S の任意の要素は, $\mathcal{C}$ 内の高々3 つの部分集合にしか出現しない.
> 解: S の最大被覆. ここで**被覆** (*covering*) とは, $\mathcal{C}$ の互いに素な集合からなる集合族である.

$S = \{s_1, \ldots, s_m\}$, $\mathcal{C} = \{C_1, \ldots, C_n\}$ とし, $C_i = \{s_{i1}, s_{i2}, s_{i3}\}$ を MAX 3SC-3 の問題例とする. ここで, $s_{i1}, s_{i2}, s_{i3} \in S$ である. また, $\mathcal{C}^*$ を S の最大被覆とする. コスト関数 γ を単一コスト関数と仮定する.

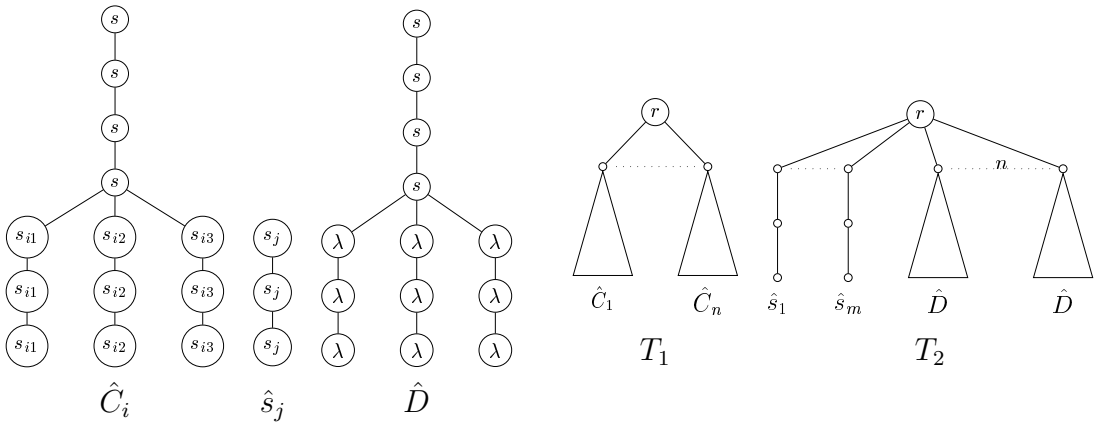


図 3.11: 木 $\hat{C}_i$ $(1 \leq i \leq n)$, $\hat{s}_j$ $(1 \leq j \leq m)$, $\hat{D}$(左) と木 T_1, T_2(右).

まず, C_i $(1 \leq i \leq n)$ に対する $\hat{C}_i$, s_j $(1 \leq j \leq m)$ に対する $\hat{s}_j$, $\hat{D}$ という 3 種類の木を図 3.11(左) のように構成する. ここで, λ は新しいラベルである. そして, T_1 と T_2 を図 3.11(右) のように構成する. MAX 3SC-3 の問題例から, 木 T_1 と T_2 への変換を f という. このとき, 文献 [51] と同様に次の補題が成り立つ.

補題 3.23. $\mathtt{A} \in \{\text{SgAln,BotAln}\}$ に対して, M を $\mathcal{M}^u_{\mathtt{A}}(T_1, T_2)$ の最小コストマッピングとする. このとき, 任意の T_1 中の $\hat{C}_i$ に対して, M は (a) $\hat{C}_i$ のすべての 3 つの部分木 $s_{\hat{i}1}$, $s_{\hat{i}2}$, $s_{\hat{i}3}$ を T_2 の同一の 3 つの部分木に対応付けるか, (b) $\hat{C}_i$ を T_2 中のダミーの部分木 $\hat{D}$ に対応付ける.

注意 3.24. $\mathtt{A} \in \{\text{Ilst,Acc,AccSg,AccBot}\}$ に対して, 補題 3.23 は, $\mathcal{M}^u_{\mathtt{A}}(T_1, T_2)$ のマッピングでは成り立たない. i, i' に対して, $\hat{C}_i$ が補題 3.23(a) を, $\hat{C}_{i'}$ が補題 3.23(b) を満たすと仮定する. ここで, $s_{\hat{ij}}$ でラベル付された T_1 の葉を v_{ij}, T_2 の葉を w_{ij} とし, λ でラベル付された T_2 の葉を w_λ とする.

このとき, $M = \{(v_{i1}, w_{i1}), (v_{i2}, w_{i2}), (v_{i'1}, w_\lambda)\}$ は補題 3.23 の T_1 と T_2 のマッピングの一

部に対応する. しかし, $w_\lambda < w_{i1} \sqcup w_{i2}$ だが $v_{i'1} \not< v_{i1} \sqcup v_{i2}$ が成り立つ. ゆえに, M は孤立部分木マッピングではない.

補題 3.25. $\mathtt{A} \in \{\text{SgAln,BotAln}\}$ に対して, $\tau^u_{\mathtt{A}}(T_1, T_2) = 9n + 3m - |\mathcal{C}^*| + 2$ となる.

[**証明**]. M を $\mathcal{M}_{\mathtt{A}}(T_1, T_2)$ の最小コストのマッピングとし, $k = |\mathcal{C}^*|$ とする.

補題 3.23 より, 任意の $C_i \in \mathcal{C}^*$ に対して, M は $\hat{C}_i$ の 3 つの部分木 $\hat{s}_{i1}$, $\hat{s}_{i2}$, $\hat{s}_{i3}$ を T_2 の同一の 3 つの部分木に対応付ける. T_1 の $\hat{C}_i$ と T_2 の $\hat{s}_{i1}$, $\hat{s}_{i2}$, $\hat{s}_{i3}$ の間のマッピングのコストは 4 (これは, T_1 の $\hat{C}_i$ において M で対応付けられていないノードの数) であり, $|\mathcal{C}^*| = k$ なので, $\mathcal{C}^*$ に関する M のコストは $4k$ である. また, 補題 3.23 より, 任意の $C_i \in \mathcal{C} - \mathcal{C}^*$ に対して, M は T_1 の $\hat{C}_i$ を T_2 のいくつかのダミー木 $\hat{D}$ に対応付ける. T_1 の $\hat{C}_i$ と T_2 の $\hat{D}$ の間のマッピングのコストは 9 (これは, T_1 で s_{ij} $(j = 1, 2, 3)$ とラベル付けられたノードと T_2 で λ とラベル付けられたノードの組の数) であり, $|\mathcal{C} - \mathcal{C}^*| = n - k$ なので, $\mathcal{C} - \mathcal{C}^*$ に関する M のコストは $9(n - k)$ である. さらに, M は, T_2 の s_j に対する $m - 3k$ 個の部分木 $\hat{s}_j$, T_2 の $n - (n - k) = k$ 個のダミー木 $\hat{D}$, r とラベル付けられた 2 つの根, を対応付けないので, これに関連する M のコストは $3(m - 3k) + 13k + 2$ となる.

したがって, $\tau^u_{\mathtt{A}}(T_1, T_2) = \gamma(M) = 4k + 9(n - k) + 3(m - 3k) + 13k + 2 = 9n + 3m - k + 2$ となる. □

定理 3.26. $\mathtt{A} \in \{\text{SgAln,BotAln}\}$ に対して, $\tau^u_{\mathtt{A}}(T_1, T_2)$ の計算問題は MAX SNP 困難である.

[**証明**]. T_1 と T_2 のマッピング $M \in \mathcal{M}_{\mathtt{A}}(T_1, T_2)$ から $\mathcal{C}$ の被覆 $\mathcal{C}'$ へ変換する以下のようなアルゴリズム g を考える:

> もし M が, T_1 の $\hat{C}_i$ における $\hat{s}_{i1}$, $\hat{s}_{i2}$, $\hat{s}_{i3}$ を, T_2 の同一の 3 つの部分木に対応付けるならば, C_i を $\mathcal{C}'$ に追加する.

文献 [61] の定理 7 より, $n/7 \leq opt(I)$ となる. よって, 補題 3.25, および, $m \leq 3n$ と $opt(I) \geq 1$ から以下の不等式が成り立つ.

$$\begin{aligned} opt(f(I)) &= 9n + 3m - opt(I) + 2 \leq 18n - opt(I) + 2 \leq 127 \cdot opt(I), \\ s_2 - opt(f(I)) &= \gamma(M) - \tau^u_{\mathtt{A}}(T_1, T_2) \\ &\geq 9n + 3m - |\mathcal{C}'| + 2 - (9n + 3m - opt(I) + 2) \\ &= opt(I) - |\mathcal{C}'| = opt(I) - s_1. \end{aligned}$$

したがって, (f, g) は MAX 3SC-3 からこの問題への L 還元となる. □

ここで, 編集距離 $\tau^u_{\mathrm{TAI}}(T_1, T_2)$ と同様に $\tau^u_{\mathrm{ALN}}(T_1, T_2)$ の計算問題は一般には MAX SNP 困難であるが, $\tau^u_{\mathrm{TAI}}(T_1, T_2)$ とは異なり T_1 と T_2 の次数がある定数以下である (これを次数限定という) 場合には $\tau^u_{\mathrm{ALN}}(T_1, T_2)$ は多項式時間で計算可能である [21]. 同様に, $\tau^u_{\mathrm{SGALN}}(T_1, T_2)$ と $\tau^u_{\mathrm{BOTALN}}(T_1, T_2)$ に対しても以下の定理が成り立つ.

定理 3.27. *T_1 と T_2 が次数限定であれば, $\tau^u_{\mathrm{SGALN}}(T_1, T_2)$ と $\tau^u_{\mathrm{BOTALN}}(T_1, T_2)$ は多項式時間で計算可能である.*

[**証明**]. 文献 [21] による δ^o_{ALN} の計算の再帰式から次数限定 δ^u_{ALN} の計算の再帰式への改良と同様に, 図 3.9 の τ^o_{SGALN} における τ^o_{TOP} を図 3.8 の τ^u_{TOP} と置き換え, 図 3.9 の $\delta^o_{\mathrm{SGALN}}$ の再帰式を図 3.12 の再帰式に改良することで, τ^u_{SGALN} を計算するアルゴリズムを設計することができる. ここで, $A \subseteq T_1(i)$ であり, $B \subseteq T_2(j)$ である. T_1 と T_2 は次数限定なので, A と B の組合せの数も定数以下となる. したがって, $\delta^u_{\mathrm{SGALN}}(A, B)$ は多項式時間で計算可能である.

さらに, τ^o_{BOTALN} を計算する再帰式 (定理 3.21 参照) における $diff^o$ を $diff^u$ と置き換え, 上と同様の改良を加えることで, 次数限定 $\delta^u_{\mathrm{BOTALN}}$ を多項式時間で計算できるアルゴリズムを設計することができる. □

$$\delta^u_{\mathrm{SGALN}}(A,B) = \min\left\{\begin{array}{l} \min\limits_{T_1[i']\in A, T_2[j']\in B}\left\{\begin{array}{l}\delta^u_{\mathrm{SGALN}}(A-\{T_1[i']\}, B-\{T_2[j']\}) \\ \quad +\tau^u_{\mathrm{SGALN}}(T_1[i'], T_2[j'])\end{array}\right\}, \\ \min\limits_{T_1[i']\in A, B'\subseteq B}\left\{\begin{array}{l}\delta^u_{\mathrm{SGALN}}(A-\{T_1[i']\}, B-B') \\ \quad +\delta^u_{\mathrm{SGALN}}(T_1(i'), B')+\gamma(i',\varepsilon)\end{array}\right\}, \\ \min\limits_{A'\subseteq A, T_2[j_t]\in B}\left\{\begin{array}{l}\delta^u_{\mathrm{SGALN}}(A-A', B-\{T_2[j']\}) \\ \quad +\delta^u_{\mathrm{SGALN}}(A', T_2(j'))+\gamma(\varepsilon,j')\end{array}\right\} \end{array}\right\}.$$

図 3.12: 次数限定 $\delta^u_{\mathrm{SGALN}}$ を計算する再帰式.

第4章　木アライメント距離の計算

本章では, 木アライメント距離に着目し, アンカーアライメントを用いたアンカーアライメント距離を導入, 実験を行っている. また, 巡回的順序木におけるアライメント距離を求めるアルゴリズムの設計も行っている. なお, 本章の内容は論文 [20, 55, 58] に基づいている.

4.1　アンカーアライメント問題と劣制限マッピング

Schiermer と Giegerich [37] が導入したアンカーアライメント問題は, 2 つの木とその間の Tai マッピングを入力として, 入力の Tai マッピングの関係を保持するような, 入力された 2 つの木のアライメント木である**アンカーアライメント木** (*anchored alignment tree*) を構築する問題である. 一方で, この問題では, アライメント木を構築できない場合が考慮されていない. そのため, 本論文ではアンカーアライメント問題を以下の通りに再定義する.

> 問題例: 2 つの木 T_1, T_2 と, マッピング $M \subseteq V(T_1) \times V(T_2)$. ここで, M を**アンカー** (*anchoring*) と呼ぶ.
> 解: すべての $(v, w) \in M$ に対する, $(l(v), l(w))$ でラベル付されたノードを含む T_1, T_2 のアンカーアライメント木 $\mathcal{T}$. そのような $\mathcal{T}$ が存在しない場合は “no” を返す.

定理 2.14 でも触れた通り, 劣制限マッピングとアライメント可能マッピングは等価であることが Kuboyama [29] によって示されており, アライメント可能マッピングの最小値で

あるアライメント距離を求めるために, 劣制限マッピングの最小値を用いることができる.

本節では, 劣制限マッピングを特徴づける被覆列を導入し, アライメント可能マッピングが劣制限マッピングであることを, ([29] の方法とは異なる) 被覆列を用いた証明にて示す. また, 劣制限マッピングがアライメント可能マッピングであることを, 構成的証明で示し, その証明の手順を用いて, アンカーからアライメント木を構築するアルゴリズムを設計する.

定義 4.1 (被覆集合と被覆列)**.** T を r を根とする木とし, v を T のノードとする. また, $U \subseteq V(T)$ とする. さらに, $[r, v]$ のノードを $v = v_1, \ldots, v_n = r$ とする. このとき, 集合 $\{w \in T[v] \mid w \in U\}$ ($T[v] \cap U$ と同義) を $v(\in T)$ の U に関する**被覆集合** (*cover set*) といい, $C_T(v, U)$ で表す. また, 列 $C_1, \ldots, C_n$ の各 $C_i (1 \leq i \leq n)$ が $C_i = C_T(v_i, U)$ を満たすとき, この列 $C_1, \ldots, C_n$ を $v(\in T)$ の U に関する**被覆列** (*cover sequence*) といい, $S_T(v, U)$ で表す.

特に, T_1, T_2 間のマッピング M に関係する被覆列として, $(u, v) \in M$ に対して, $S_{T_1}(u, M|_1)$, $S_{T_2}(v, M|_2)$ を用いる. $r_1 = r(T_1), r_2 = r(T_2)$ に対して, $[r_1, u]$, $[r_2, v]$ をそれぞれ, $S_{T_1}(u, M|_1)$, $S_{T_2}(v, M|_2)$ の**パス** (*path*) といい, $P_{T_1}(u)$, $P_{T_2}(v)$ で表す. さらに, $C_i \in S_{T_1}(u, M|_1)$, $C_j \in S_{T_2}(v_j, M|_2)$ に対して, $C'_i = \{u' \in T_2 \mid v' \in C_i, (u', v') \in M\}$ と C'_j が集合の包含関係の意味で比較可能 (不能) なとき, C_i と C_j を比較可能 (不能) という. 加えて, 比較不能となるような $C_i \in S_{T_1}(u, M|_1)$, $C_j \in S_{T_2}(v_j, M|_2)$ が存在するとき, $S_{T_1}(u, M|_1)$ と $S_{T_2}(v_j, M|_2)$ を比較不能といい, そうでないとき, $S_{T_1}(u, M|_1)$ と $S_{T_2}(v_j, M|_2)$ を比較可能という.

例 4.2. 図 4.1 の木 T_0, T_1, T_2, T_3 について考える. 図 4.1 の $M_i (i = 1, 2, 3)$ は, T_0 と T_i とのマッピングである. ここで, M_1, M_2 は劣制限であるが, M_3 は劣制限ではない. また, $M_4 = M_3 \setminus \{(u_1, v_1)\}$, $M_5 = M_3 \setminus \{(u_2, v_2)\}$, $M_6 = M_3 \setminus \{(u_3, v_3)\}$ は, それぞれ劣制限である.

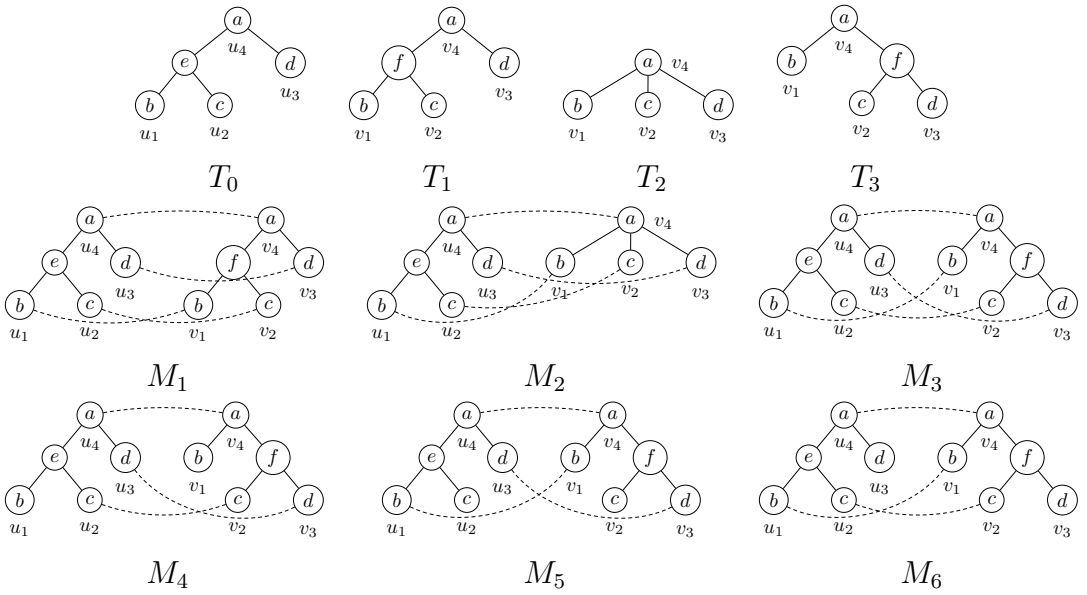


図 4.1: 例 4.2 の木 T_0, T_1, T_2, T_3 (上) とマッピング M_1, M_2, M_3 (中央), マッピング M_4, M_5, M_6 (下).

このとき, マッピング $M_1,\ M_2,\ M_3$ に対する被覆列 $S_{T_0}(j, M_i|_1),\ S_{T_i}(j, M_i|_2)$ は図 4.2 のようになる. ただし, マッピング $M_i\ (i = 1, 2, 3)$ に対して, $u_j \in M_i|_1$ と $v_j \in M_i|_2$ $(j = 1, 2, 3, 4)$ を, 添え字 j で表している.

$\{1, 2\}, \{2, 3\}$ は (集合の包含関係の意味で) 比較不能であるため, $S_{T_0}(2, M_3|_1), S_{T_3}(2, M_3|_2)$ は比較不能である. 一方, $(i, j) \in \{1, 2, 3\} \times \{1, 2, 3, 4\} \setminus \{(3, 2)\}$ に対して, $S_{T_0}(j, M_i|_1)$, $S_{T_i}(j, M_i|_2)$ は比較可能である.

$S_{T_0}(1, M_i|_1) = \{1\}, \{1, 2\}, \{1, 2, 3, 4\}$.　　$S_{T_2}(1, M_2|_2) = \{1\}, \{1, 2, 3, 4\}$.
$S_{T_0}(2, M_i|_1) = \{2\}, \{1, 2\}, \{1, 2, 3, 4\}$.　　$S_{T_2}(2, M_2|_2) = \{2\}, \{1, 2, 3, 4\}$.
$S_{T_0}(3, M_i|_1) = \{3\}, \{1, 2, 3, 4\}$.　　$S_{T_2}(3, M_2|_2) = \{3\}, \{1, 2, 3, 4\}$.
$S_{T_0}(4, M_i|_1) = \{1, 2, 3, 4\}$.　　$S_{T_2}(4, M_2|_2) = \{1, 2, 3, 4\}$.
$S_{T_1}(1, M_1|_2) = \{1\}, \{1, 2\}, \{1, 2, 3, 4\}$.　　$S_{T_3}(1, M_3|_2) = \{1\}, \{1, 2, 3, 4\}$.
$S_{T_1}(2, M_1|_2) = \{2\}, \{1, 2\}, \{1, 2, 3, 4\}$.　　$S_{T_3}(2, M_3|_2) = \{2\}, \{2, 3\}, \{1, 2, 3, 4\}$.
$S_{T_1}(3, M_1|_2) = \{3\}, \{1, 2, 3, 4\}$.　　$S_{T_3}(3, M_3|_2) = \{3\}, \{2, 3\}, \{1, 2, 3, 4\}$.
$S_{T_1}(4, M_1|_2) = \{1, 2, 3, 4\}$.　　$S_{T_3}(4, M_3|_2) = \{1, 2, 3, 4\}$.

図 4.2: 図 4.1 のマッピング M_1, M_2, M_3 に対する被覆列 $S_{T_0}(j, M_i|_1),\ S_{T_i}(j, M_i|_2)(i = 1, 2, 3)$.

$S_{T_0}(2, M_4|_1) = \{2\}, \{2\}, \{2, 3, 4\}$.　$S_{T_3}(2, M_4|_2) = \{2\}, \{2, 3\}, \{2, 3, 4\}$.
$S_{T_0}(3, M_4|_1) = \{3\}, \{2, 3, 4\}$.　$S_{T_3}(3, M_4|_2) = \{3\}, \{2, 3\}, \{2, 3, 4\}$.
$S_{T_0}(4, M_4|_1) = \{2, 3, 4\}$.　$S_{T_3}(4, M_4|_2) = \{2, 3, 4\}$.
$S_{T_0}(1, M_5|_1) = \{1\}, \{1\}, \{1, 3, 4\}$.　$S_{T_3}(1, M_5|_2) = \{1\}, \{1, 3, 4\}$.
$S_{T_0}(3, M_5|_1) = \{3\}, \{2, 3, 4\}$.　$S_{T_3}(3, M_5|_2) = \{3\}, \{3\}, \{1, 3, 4\}$.
$S_{T_0}(4, M_5|_1) = \{1, 3, 4\}$.　$S_{T_3}(4, M_5|_2) = \{1, 3, 4\}$.
$S_{T_0}(1, M_6|_1) = \{1\}, \{1, 2\}, \{1, 2, 4\}$.　$S_{T_3}(1, M_6|_2) = \{1\}, \{1, 2, 4\}$.
$S_{T_0}(2, M_6|_1) = \{2\}, \{1, 2\}, \{1, 2, 4\}$.　$S_{T_3}(2, M_6|_2) = \{2\}, \{2\}, \{1, 2, 4\}$.
$S_{T_0}(4, M_6|_1) = \{1, 2, 4\}$.　$S_{T_3}(4, M_6|_2) = \{1, 2, 4\}$.

図 4.3: 図 4.1 のマッピング $M_4,\ M_5,\ M_6$ に対する被覆列 $S_{T_0}(j, M_i|_1),\ S_{T_2}(j, M_i|_2)(i = 1, 2, 3)$.

さらに, マッピング M_4, M_5, M_6 に対する被覆列 $S_{T_0}(j, M_i|_1)$, $S_{T_2}(j, M_i|_2)$ は図 4.3 のようになる. ここで, $j \in I_i$, $I_4 = \{2, 3, 4\}$, $I_5 = \{1, 3, 4\}$, $I_6 = \{1, 2, 4\}$ である. これら被覆列はすべて比較可能である.

定理 4.3. *T_1, T_2 を木とする. また, M を T_1, T_2 間のマッピングとする. このとき, M が T_1, T_2 間の劣制限マッピングではないとき, かつそのときに限り, $S_{T_1}(u, M|_1)$, $S_{T_2}(v, M|_2)$ が比較不能となるような $(u, v) \in M$ が存在する.*

[**証明**]. 被覆集合 $C_1 \in S_{T_1}(u_1, M|_1)$, $C_2 \in S_{T_2}(v_1, M|_2)$ が比較不能となるような $(u_1, v_1) \in M$ が存在すると仮定する. このとき, $u_2 \in C_1 \setminus C_2$ かつ $v_3 \in C_2 \setminus C_1$ を満たす $u_2 \in M|_1$, $v_3 \in M|_2$ が存在する. $u^*,\ v^*$ をそれぞれ, ノード $u_1 \sqcup u_2,\ v_1 \sqcup v_3$ とする. このとき, $C_1 = C_{T_1}(u^*, M|_1)$, $C_2 = C_{T_2}(v^*, M|_2)$ を仮定することができる. また, u_3, v_2 を考える.

$u_3 \notin C_1$ より, $u^* < u_1 \sqcup u_3$ が, $v_2 \notin C_2$ より, $v^* < v_2 \sqcup v_3$ がそれぞれ成り立つ. ゆえに, たとえ $u^* = u_1 \sqcup u_2 < u_2 \sqcup u_3$ であろうとも, $v^* = v_1 \sqcup v_3 < v_2 \sqcup v_3$ が成り立ち, これは M が劣制限マッピングでないことを意味する.

逆に, M が劣制限ではないと仮定する. このとき, 以下の (1) を満たし, (2) と (3) のうち一方を満たす $u_1, u_2, u_3 \in M|_1$, $v_1, v_2, v_3 \in M|_2$ が存在する. (1) $u_1 \sqcup u_2 < u_1 \sqcup u_3$, (2) $v_2 \sqcup v_3 < v_1 \sqcup v_3$, (3) $v_2 \sqcup v_3 > v_1 \sqcup v_3$.

条件 (1) より, 被覆列 $S_{T_1}(u_1, M|_1)$, $S_{T_1}(u_2, M|_2)$ は, $\{u_1, u_2\} \subseteq C_1$, $u_3 \notin C_1$ を満たす被覆集合 C_1 を含む. 一方, 条件 (2) より, 被覆列 $S_{T_2}(v_2, M|_2)$ は, $\{v_2, v_3\} \subseteq C_2$, $v_1 \notin C_2$ を満たす被覆集合 C_2 を含み, C_1 と C_2 は比較不能である. また, 条件 (3) より, 被覆列 $S_{T_2}(v_1, M|_2)$ は, $\{v_1, v_3\} \subseteq C_3$, $v_2 \notin C_3$ を満たす被覆集合 C_3 を含み, C_1, C_3 は比較不能である. よって (2), (3) のどちらを満たす場合でも比較不能な被覆列が存在する. □

系 4.4. T_1, T_2 を木とする. また, M を T_1, T_2 間のマッピングとする. このとき, M が T_1, T_2 間の劣制限マッピングのとき, かつそのときに限り, すべての組 $(u, v) \in M$ に対して, $S_{T_1}(u, M|_1)$ と $S_{T_2}(v, M|_2)$ が比較可能となる.

次に, 定理 4.3 と系 4.4 を用いて, 定理 2.14 の別証明を与える.

補題 4.5. T_1, T_2 を木とし, M を T_1, T_2 間のアライメント可能マッピングとする. このとき, M は劣制限マッピングでもある.

[**証明**]. すべての $(u, v) \in M$ に対して, $u \in M|_1$ と $v \in M|_2$ を同一視する. アライメント可能マッピング M に対して, $M = M_{\mathcal{T}}$ となるようなアライメント木 $\mathcal{T}$ が存在する. また, M を劣制限マッピングではないと仮定する. 定理 4.3 より, 被覆集合 $C_1 \in S_{T_1}(u, M|_1)$, $C_2 \in S_{T_2}(v, M|_2)$ が比較不能となるような組 $(u, v) \in M$ が存在する. このとき, $u_1 \in C_1 \backslash C_2$, $v_2 \in C_2 \backslash C_1$ となるような, $u_1 \in M|_1$, $v_2 \in M|_2$ が存在する. ここで, u', v' を $u \sqcup u_1$, $v \sqcup v_2$ とする. このとき, $C_1 = C_{T_1}(u', M|_1)$, $C_2 = C_{T_2}(v', M|_2)$ と想定できる. また, $(u_1, v_1) \in M$, $(u_2, v_2) \in M$ となるような v_1, u_2 を考える.

$M = M_{\mathcal{T}}$ なので, $(l(u_1), l(v_1))$, $(l(u_2), l(v_2))$ の両方が $\mathcal{T}$ に出現する. また, $u_1 \in C_1$ より, $\mathcal{T}$ で $(l(u), l(v)) < (l(u_1), l(v_1))$ が, $v_2 \in C_2$ より, $\mathcal{T}$ で $(l(u), l(v)) < (l(u_2), l(v_2))$ が成り立つ. さらに, $u_1 \in C_1 \setminus C_2$ と $v_2 \in C_2 \setminus C_1$ より, 以下の通りに $(l(u_1), l(v_1))$ と $(l(u_2), l(v_2))$ が $\mathcal{T}$ 内で先祖子孫関係を持たないことを示すことができる.

もし $\mathcal{T}$ 中で $(l(u_1), l(v_1)) < (l(u_2), l(v_2))$ ならば, T_1 で $u < u_1 < u_2$ が, T_2 で $v < v_1 < v_2$ が成り立つ. このとき, $u' = u_1$, $v' = v_2$ より, $C_1 = C_{T_1}(u', M|_1) \subseteq C_{T_1}(u_2, M|_1) =$

$C_{T_2}(v', M|_2) = C_2$ が成り立つ．もし $\mathcal{T}$ 中で $(l(u_2), l(v_2)) < (l(u_1), l(v_1))$ ならば，T_1 で $u < u_2 < u_1$ が，T_2 で $v < v_2 < v_1$ が成り立つ．このとき，$u' = u_1$，$v' = v_2$ より，$C_2 = C_{T_2}(v', M|_2) \subseteq C_{T_2}(v_1, M|_2) = C_{T_1}(u', M|_1) = C_1$ が成り立つ．これらは，C_1 と C_2 が比較不能であることに反する．

結果として，$(l(u), l(v)), (l(u_1), l(v_1)), (l(u_2), l(v_2)) \in \mathcal{T}$ に対して，$(l(u), l(v)) < (l(u_1), l(v_1))$，$(l(u), l(v)) < (l(u_2), l(v_2))$，$(l(u_1), l(v_1)) \# (l(u_2), l(v_2))$ が成り立ち，これは $\mathcal{T}$ が木であることに矛盾する．　□

補題 4.5 の逆を示すために，次の補題から始める．

補題 4.6. T_1, T_2 を木とし，M を T_1, T_2 間の劣制限マッピングとする．このとき，$(v, w) \in M$ に対して，$S_{T_1}(v, M|_1)$ と $S_{T_2}(w, M|_2)$ は，$S_{T_1}(v, M|_1)(S_{T_2}(w, M|_2))$ の最後の要素が $M|_1$ $(M|_2)$ となるような比較可能な単調非減少列である．

M を T_1，T_2 間の Tai マッピングとする．このとき，すべての $(v_j, w_j) \in M$ に対して，$v_j \in M|_1$ と $w_j \in M|_2$ を同一視し，その添え字 j で表す．そのような同一視の元では，$M|_1 = M|_2$ とみなすことができる．次に，集合の比較可能な単調非減少列に対して，以下の**アライン列** (*aligned sequence*) と**アラインパス** (*aligned path*) を導入する．

定義 4.7 (アライン列, アラインパス)**.** $S_1 = A_1, \ldots, A_n$，$S_2 = B_1, \ldots, B_m$ を $A_n = B_m$ となるような比較可能な単調非減少列とする．このとき，アルゴリズム 1 の手続き AlnSq により，S_1 と S_2 から構築される列 $S_1' = A_1', \ldots, A_k'$ と $S_2' = B_1', \ldots, B_k'$ を，S_1 と S_2 の**アライン列** (*aligned sequences*) という．

さらに，S_1 と S_2 のアライン列 $S_1' = A_1', \ldots, A_k'$，$S_2' = B_1', \ldots, B_k'$ に対して，$V = \{p_1, \ldots, p_k\}$，$E = \{(p_i, p_{i+1}) \mid 1 \leq i \leq k-1\}$ であり，p_1 を根に持ち $p_i (1 \leq i \leq k)$ が (A'_{k-i+1}, B'_{k-i+1}) でラベル付されている根付きラベル付きパス $P = (V, E)$ を，S_1 と S_2 の**アラインパス** (*aligned path*) と定義する．そのようなパスを単に $[p_1, \ldots, p_k]$ で表すことも

```
   procedure AlnSq(S_1, S_2)
   |  /* S_1 = A_1, ..., A_n, S_2 = B_1, ..., B_m */
1  |  i ← 1; j ← 1; k ← 1;
2  |  while i ≤ n + 1 and j ≤ m + 1 do
3  |  |  if i = n + 1 then A'_k ← λ; B'_k ← B_j; j++;
4  |  |  else if j = m + 1 then A'_k ← A_i; B'_k ← λ; i++;
5  |  |  else if A_i = B_j then A'_k ← A_i; B'_k ← B_j; i++; j++;
6  |  |  else if A_i ⊂ B_j then A'_k ← A_i; B'_k ← λ; i++;
7  |  |  else if A_i ⊃ B_j then A'_k ← λ; B'_k ← B_j; j++;
8  |  |  k++;
9  |  return S'_1 = A'_1, ..., A'_k, S'_2 = B'_1, ... B'_k;
```

アルゴリズム 1: AlnSq.

ある.

例 4.8. 例 4.2 のマッピング M_2 を考える. 補題 4.6 により, 例 4.2 の $S_{T_0}(j, M_2|_1)$, $S_{T_2}(j, M_2|_2)$ は比較可能な単調非減少列である. このとき, 図 4.4 (左) に示すアライン列 $S'_{T_0}(j, M_2|_1)$, $S'_{T_2}(j, M_2|_2)$ を構築できる. また, $S_{T_0}(j, M_2|_1)$ と $S_{T_2}(j, M_2|_2)$ のアラインパス $P_{M_2}(j) = [p_1, p_2, p_3]$ に対して, $P_{M_2}(j)$ のすべての頂点 p_i $(i = 1, 2, 3)$ のラベル $l(p_i)$ は図 4.4 (右) に示す通りである.

$S'_{T_0}(1, M_2|_1) = \{1\}, \{1, 2\}, \{1, 2, 3, 4\}$.
$S'_{T_0}(2, M_2|_1) = \{2\}, \{1, 2\}, \{1, 2, 3, 4\}$.
$S'_{T_0}(3, M_2|_1) = \{3\}, \{1, 2, 3, 4\}$.
$S'_{T_0}(4, M_2|_1) = \{1, 2, 3, 4\}$.
$S'_{T_2}(1, M_2|_2) = \{1\}, \lambda, \{1, 2, 3, 4\}$.
$S'_{T_2}(2, M_2|_2) = \{2\}, \lambda, \{1, 2, 3, 4\}$.
$S'_{T_2}(3, M_2|_2) = \{3\}, \{1, 2, 3, 4\}$.
$S'_{T_2}(4, M_2|_2) = \{1, 2, 3, 4\}$.

$P_{M_2}(j)$	$l(p_1)$	$l(p_2)$	$l(p_3)$
$P_{M_2}(1)$	$(\{1,2,3,4\}, \{1,2,3,4\})$	$(\{1,2\}, \lambda)$	$(\{1\}, \{1\})$
$P_{M_2}(2)$	$(\{1,2,3,4\}, \{1,2,3,4\})$	$(\{1,2\}, \lambda)$	$(\{2\}, \{2\})$
$P_{M_2}(3)$	$(\{1,2,3,4\}, \{1,2,3,4\})$	$(\{3\}, \{3\})$	
$P_{M_2}(4)$	$(\{1,2,3,4\}, \{1,2,3,4\})$		

図 4.4: 例 4.8 の $S_{T_0}(j, M_2|_1)$ と $S_{T_2}(j, M_2|_2)$ のアライン列 $S'_{T_0}(j, M_2|_1)$, $S'_{T_2}(j, M_2|_2)$ (左) と, $S_{T_0}(j, M_2|_1)$ と $S_{T_i}(j, M_2|_2)$ のアラインパス $P_{M_2}(j)$ のラベル (右).

また, 例 4.2 のマッピング M_4, M_5, M_6 を考える. 補題 4.6 より, 例 4.2 の $S_{T_0}(j, M_i|_1)$ と $S_{T_2}(j, M_i|_2)$ $(i = 4, 5, 6, j \in I_i)$ は, 比較可能単調非減少列である. このとき, 図 4.5 (左) に示すアライン列 $S'_{T_0}(j, M_i|_1)$, $S'_{T_2}(j, M_i|_2)$ を構築できる. また, $S_{T_0}(j, M_i|_1)$ と $S_{T_2}(j, M_i|_2)$ のアラインパス $P_{M_i}(j) = [p_1, p_2, p_3, p_4]$ に対して, $P_{M_i}(j)$ のすべての頂点 p_i のラベル $l(p_i)$

は図 4.5 (右) に示すとおりである.

$S'_{T_0}(2, M_4|_1) = \{2\}, \{2\}, \lambda, \{2,3,4\}$.
$S'_{T_0}(3, M_4|_1) = \{3\}, \lambda, \{2,3,4\}$.
$S'_{T_0}(4, M_4|_1) = \{2,3,4\}$.
$S'_{T_0}(1, M_5|_1) = \{1\}, \{1\}, \{1,3,4\}$.
$S'_{T_0}(3, M_5|_1) = \{3\}, \lambda, \{1,3,4\}$.
$S'_{T_0}(4, M_5|_1) = \{1,3,4\}$.
$S'_{T_0}(1, M_6|_1) = \{1\}, \{1,2\}, \{1,2,4\}$.
$S'_{T_0}(2, M_6|_1) = \{2\}, \lambda, \{1,2\}, \{1,2,4\}$.
$S'_{T_0}(4, M_6|_1) = \{1,2,4\}$.

$S'_{T_3}(2, M_4|_2) = \{2\}, \lambda, \{2,3\}, \{2,3,4\}$.
$S'_{T_3}(3, M_4|_2) = \{3\}, \{2,3\}, \{2,3,4\}$.
$S'_{T_3}(4, M_4|_2) = \{2,3,4\}$.
$S'_{T_3}(1, M_5|_2) = \{1\}, \lambda, \{1,3,4\}$.
$S'_{T_3}(3, M_5|_2) = \{3\}, \{3\}, \{1,3,4\}$.
$S'_{T_3}(4, M_5|_2) = \{1,3,4\}$.
$S'_{T_3}(1, M_6|_2) = \{1\}, \lambda, \{1,2,4\}$.
$S'_{T_3}(2, M_6|_2) = \{2\}, \{2\}, \lambda, \{1,2,4\}$.
$S'_{T_3}(4, M_6|_2) = \{1,2,4\}$.

$P_{M_i}(j)$	$l(p_1)$	$l(p_2)$	$l(p_3)$	$l(p_4)$
$P_{M_4}(2)$	$(\{2,3,4\}, \{2,3,4\})$	$(\lambda, \{2,3\})$	$(\{2\}, \lambda)$	$(\{2\}, \{2\})$
$P_{M_4}(3)$	$(\{2,3,4\}, \{2,3,4\})$	$(\lambda, \{2,3\})$	$(\{3\}, \{3\})$	
$P_{M_4}(4)$	$(\{2,3,4\}, \{2,3,4\})$			
$P_{M_5}(1)$	$(\{1,3,4\}, \{1,3,4\})$	$(\{1\}, \lambda)$	$(\{1\}, \{1\})$	
$P_{M_5}(3)$	$(\{1,3,4\}, \{1,3,4\})$	$(\lambda, \{3\})$	$(\{3\}, \{3\})$	
$P_{M_5}(4)$	$(\{1,3,4\}, \{1,3,4\})$			
$P_{M_6}(1)$	$(\{1,2,4\}, \{1,2,4\})$	$(\{1,2\}, \lambda)$	$(\{1\}, \{1\})$	
$P_{M_6}(2)$	$(\{1,2,4\}, \{1,2,4\})$	$(\{1,2\}, \lambda)$	$(\{1,2\}, \{2\})$	$(\{2\}, \{2\})$
$P_{M_6}(4)$	$(\{1,2,4\}, \{1,2,4\})$			

図 4.5: 例 4.8 の $S_{T_0}(j, M_i|_1)$ と $S_{T_2}(j, M_i|_2)$ のアライン列 $S'_{T_0}(j, M_i|_1)$, $S'_{T_2}(j, M_i|_2)$ (上) と, $S_{T_0}(j, M_i|_1)$ と $S_{T_2}(j, M_i|_2)$ のアラインパス $P_{M_i}(j)$ のラベル (下).

M を T_1, T_2 間の劣制限マッピングとし, $r_1 = r(T_1)$, $r_2 = r(T_2)$ とする. 補題 4.6 より, すべての $(v, w) \in M$ に対する $S_{T_1}(v, M|_1) = A_1, \ldots, A_n$ と $S_{T_2}(w, M|_2) = B_1, \ldots, B_m$ に対して, $v_1 = v$, $w_1 = w$, $v_n = r_1$, $w_m = r_2$ となる組 $P_{T_1}(v) = v_1, \ldots, v_n$, $P_{T_2}(w) = w_1, \ldots, w_m$ が存在する. また, $(v', w') \in M$ に対して $v' \in M|_1$ と $w' \in M|_2$ を同一視することで, $A_1 = B_1 = \{v\} = \{w\}$, $A_n = B_m = M|_1 = M|_2$ が成り立つ.

さらに, $S_{T_1}(v, M|_1)$, $S_{T_2}(w, M|_2)$ のアライン列 $S'_{T_1}(v, M|_1)$, $S'_{T_2}(w, M|_2)$ がそれぞれ $A'_1, \ldots, A'_k, B'_1, \ldots, B'_k$, といった形をしているとする. このとき, λ を含む, T_1 の $S'_{T_1}(v, M|_1)$ に対応するパスを $P'_{T_1}(v) = v'_1, \ldots, v'_k$ で表す. ただし, $A'_i = \lambda$ のとき $v'_i = \lambda$ で, そうでないとき $v'_i = v_{i'}$ とし, $i' = |\{l \mid 1 \leq l \leq i, v'_l \neq \lambda\}|$ である. また, λ を含む, T_2 の $S_{T_2}(v, M|_1)$

に対応するパスを $P'_{T_2}(w) = w'_1, \ldots, w'_k$ で表す. ただし, $B'_j = \lambda$ のとき $w'_j = \lambda$ で, そうでないとき $w'_j = w_{j'}$ とし, $j' = |\{l \mid 1 \leq l \leq j, w'_l \neq \lambda\}|$ である.

補題 4.9. *M を T_1, T_2 間の劣制限マッピングとする. また, $(v_1, w_1), (v_2, w_2) \in M$ に対して,*

$$S'_{T_1}(v_1, M|_1) = A'_1, \ldots, A'_k, \quad S'_{T_2}(w_1, M|_2) = B'_1, \ldots, B'_k,$$
$$S'_{T_1}(v_2, M|_1) = C'_1, \ldots, C'_h, \quad S'_{T_2}(w_2, M|_2) = D'_1, \ldots, D'_h$$

とする. $(A'_i, B'_i) = (C'_j, B'_j)$ かつ $(A'_{i-1}, B'_{i-1}) \neq (C'_{j-1}, B'_{j-1})$ となる最大の添え字を i, j $(2 \leq i \leq k, 2 \leq j \leq h)$ としたとき, すべての a $(1 \leq a \leq i-1)$, b $(1 \leq b \leq j-1)$ に対して $(A'_a, B'_a) \neq (C'_b, B'_b)$ が成り立つ.

[**証明**]. $A = A'_1 \cup \cdots \cup A'_{i-1} \setminus \{\lambda\}$, $B = B'_1 \cup \cdots \cup B'_{i-1} \setminus \{\lambda\}$, $C = C'_1 \cup \cdots \cup C'_{j-1} \setminus \{\lambda\}$, $D = D'_1 \cup \cdots \cup D'_{j-1} \setminus \{\lambda\}$ とする. このとき, $A \cap C = \emptyset$, $B \cap D = \emptyset$ を示す.

$A \cap C \neq \emptyset$ と仮定する. このとき, $v \in A \cap C$ となるノード $v \in T_1$ が存在し, $v \in P'_{T_1}(v_1)$, $v \in P'_{T_1}(v_2)$ が成り立つ. T_1 が根付き木であることから, $v_1 \sqcup v$, $v_2 \sqcup v$ は, $v_1 \sqcup v < v_2 \sqcup v$, $v_2 \sqcup v < v_1 \sqcup v$, $v_1 \sqcup v = v_2 \sqcup v$ のいずれかを満たす. $p'_1 = v_1$, $q'_1 = v_2$, $p'_k = q'_h = r(T_1)$ を満たす $P'_{T_1}(v_1) = p'_1, \ldots, p'_k$ と $P'_{T_1}(v_2) = q'_1, \ldots, q'_h$ に対して, $v^* = p'_{i'+1} = q'_{j'+1} \in T_1$ とする. このとき, $v_1 \sqcup v_2 = v^*$ が成り立つ.

$v_1 \sqcup v < v_2 \sqcup v$ が成り立つならば, $v_1 \sqcup v < v^*$ が成り立つ. これは, $v \notin C$ が成り立つことを意味する. $v_2 \sqcup v < v_1 \sqcup v$ が成り立つならば, $v_2 \sqcup v < v^*$ が成り立つ. これは, $v \notin A$ が成り立つことを意味する. $v_1 \sqcup v = v_2 \sqcup v$ が成り立つならば, $v^* \leq v_1 \sqcup v = v_2 \sqcup v$ が成り立つ. これは $v \notin A \cap C$ が成り立つことを意味する. 以上より, $A \cap C \neq \emptyset$ が成り立つ.

同様にして, $B \cap D = \emptyset$ を示すことができる. □

定義 4.10 (併合グラフ)**.** *T_1, T_2 間の劣制限マッピング M に対して, $\mathcal{P}_M$ を M に関係するすべてのアラインパスからなる集合とする. このとき, $\mathcal{P}_M$ のラベルが同じ頂点を同一視*

することで構築される根付きグラフを M の**併合グラフ** $(merged\ graph)\mathcal{G}_M$ と定める. この $\mathcal{G}_M$ の根ノードのラベルは $(M|_1, M|_2)$ である.

補題 4.11. *T_1, T_2 を木とし, M を T_1, T_2 間の劣制限マッピングとする. このとき, M 併合グラフ $\mathcal{G}_M$ は根付きラベル付き木となる.*

例 4.12. 例 4.2 のマッピング M_2, M_4, M_5, M_6 を考える. 例 4.8 より,
$\mathcal{P}_{M_2} = \{P_{M_2}(1), P_{M_2}(2), P_{M_2}(3), P_{M_2}(4)\}$ と $\mathcal{P}_{M_i} = \{P_{M_i}(j) \mid j \in I_i\}$ $(i = 4, 5, 6)$ が成り立つ. このとき, $\mathcal{P}_{M_i}$ の併合グラフ $\mathcal{G}_{M_i}$ は図 4.6 に示すとおりになる.

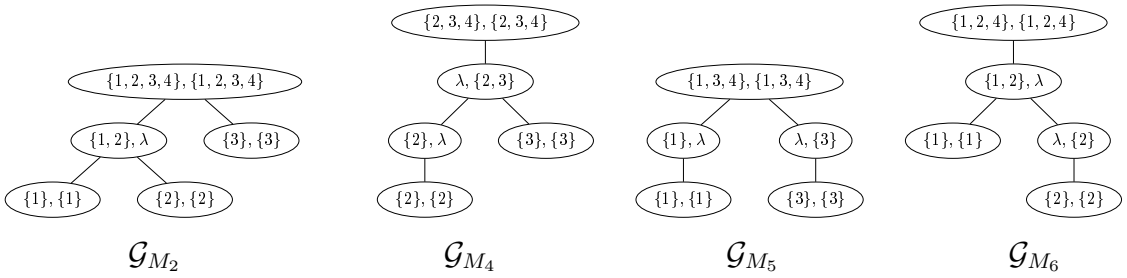


図 4.6: 例 4.12 の併合グラフ $\mathcal{G}_{M_i}$ $(i = 2, 4, 5, 6)$.

T_1, T_2 を木とし, M を T_1, T_2 間の劣制限マッピング, $\mathcal{G}_M$ を M の併合グラフとする. このとき, 頂点 $u \in \mathcal{G}_M$ に対して, u のラベルは (A, B) の形であり, A については $A \subseteq M_1$ または $A = \lambda$, B については $B \subseteq M_2$ または $B = \lambda$ が成り立つ. $A \neq \lambda$ のとき, A に対応する T_1 の唯一のノードが存在する. 同様に, $B \neq \lambda$) のとき, B に対応する T_2 の唯一のノードが存在する. そのようなノードを, それぞれ $v_{T_1}(A)$, $v_{T_2}(B)$ で表す.

すべての頂点 $u \in \mathcal{G}_M$ に対して, u のラベル (A, B) を, $A \neq \lambda$ かつ $B \neq \lambda$ のとき $(l(v_{T_1}(A)), l(v_{T_2}(B)))$ に; $A = \lambda$ かつ $B \neq \lambda$ のとき $(\varepsilon, l(v_{T_2}(B)))$ に; $A \neq \lambda$ かつ $B = \lambda$ のとき $(l(v_{T_1}(A)), \varepsilon)$ に置き換えることを考える. このすべての $u \in \mathcal{G}_M$ のラベルの置き換えによって $\mathcal{G}_M$ から得られる木を, $\mathcal{G}_M^*$ で表す.

補題 4.13. *T_1, T_2 を木とし, M を T_1, T_2 間の劣制限マッピング, $\mathcal{G}_M$ を M の併合グラフとする. このとき, $\mathcal{G}_M^*$ は T_1, T_2 間のアライメント木の部分木である.*

[**証明**]. $\mathcal{G}_M$ の頂点のラベルは (A, B) という形である. このとき, $\mathcal{G}_M$ の定義および補題 4.11 より, すべての A (B) に対して, $v_{T_1}(A)$ $(v_{T_2}(B))$ を接続して, λ を削除し, ラベルが λ である頂点 v に対して, v の子を v の親に接続することで T_1 (T_2) の部分木を構築する. □

$\mathcal{P}_1$ を $T_1 \setminus \{P_{T_1}(v) \mid v \in M|_1\}$ の根付き極大パスすべてからなる集合とし, $\mathcal{P}_2$ を $T_2 \setminus \{P_{T_2}(w) \mid w \in M|_2\}$ の根付き極大パスすべてからなる集合とする. $r(P) = p_1$ なるすべての $P = [p_1, \ldots, p_k] \in \mathcal{P}_1$ に対して, T_1 の p_1 の親である $v \in T_1$ が存在する. このような v を $par_{T_1}(P)$ で表す. 同様に, $r(Q) = q_1$ なるすべての $Q = [q_1, \ldots, q_k] \in \mathcal{P}_2$ に対して, T_2 の q_1 の親である $v \in T_2$ が存在する. このような v を $par_{T_2}(Q)$ で表す.

さらに, すべての $P = [p_1, \ldots, p_k] \in \mathcal{P}_1$ に対して, $l(p_i)$ を $(l(p_i), \varepsilon)$ と置き換えることで得られるラベル付きパスを $\langle P, \varepsilon \rangle$ で表し, すべての $Q = [q_1, \ldots, q_k] \in \mathcal{P}_2$ に対して, $l(q_i)$ を $(\varepsilon, l(q_i))$ と置き換えることで得られるラベル付きパスを $\langle \varepsilon, Q \rangle$ で表す.

補題 4.14. *T_1, T_2 を木とし, M を T_1, T_2 間の劣制限マッピングとする. このとき, M はアライメント可能マッピングでもある.*

[**証明**]. M から T_1, T_2 間のアライメント木を構築すれば十分である. 補題 4.13 より, $\mathcal{G}_M^*$ は T_1, T_2 間のアライメント木の部分木である. 完全なアライメント木を構築するためには, M でカバーされていないパスを挿入する必要がある. これは, 上で $\mathcal{P}_1$, $\mathcal{P}_2$ と定義したものである. ゆえに, すべての $P \in \mathcal{P}_1$ に対してパス $\langle P, \varepsilon \rangle$ を $\mathcal{G}_M^*$ の適切な $par_{T_1}(P)$ の子に, すべての $Q \in \mathcal{P}_2$ に対してパス $\langle \varepsilon, Q \rangle$ を $\mathcal{G}_M^*$ の適切な $par_{T_1}(P)$ の子に挿入することで, T_1, T_2 間のアライメント木を構築することができる. □

次の定理 4.15 では, 順序木, 無順序木の区別は必要ない.

定理 4.15. *木 T_1, T_2 とその間のマッピング M に対して, $n = |T_1|$, $m = |T_2|$, $h = \max\{h(T_1), h(T_2)\}$, $a = |M|$ とする. このとき, アンカーアライメント問題は $O(ha^2+n+m)$ 時間で計算でき, そのときの領域計算量は $O(ha)$ となる.*

[**証明**]. 集合操作に対して, a ビット $\{0,1\}$ ベクトルを合計 $O(ha)$ 領域分用いる. これは $O(ha)$ 時間で準備可能である.

アンカー M に対して, まず, 劣制限かどうかを系 4.4 を用いて, $O(ha)$ 時間で確認する. もし M が劣制限でないならば, "no" を返して終了する. そうでなければ, アルゴリズム 1 に示す AlnSq を用いて T_1, T_2 間のアライメント木の部分木 $\mathcal{T}$ を構築する. アルゴリズム 1 では, $A_i = B_j$, $A_i \subset B_j$, $A_i \supset B_j$ のいずれかであることを $O(a)$ 時間で確認するため, アルゴリズム 1 の計算時間は $O(ha)$ であり, この手続きすべての時間計算量は $O(ha^2)$ となる. 次に, 併合グラフ $\mathcal{G}_M$ を構築し, $\mathcal{G}_M$ を $\mathcal{G}_M^*$ に置き換えることで $\mathcal{T}$ を得る. 併合グラフを得るには, 順序木に対しては (後行走査順で隣接するノードの確認で十分であるため) $O(ha)$ 時間, 無順序木に対しては $O(ha^2)$ 時間を要する. 最後に, $\langle P, \varepsilon\rangle$ と $\langle \varepsilon, Q\rangle$ を $\mathcal{T}$ に追加することで完全ならアライメント木を得る. これは, 補題 4.14 より $O(n+m)$ 時間で可能である.

ゆえに, 上記手順によりアンカーアライメント問題を計算することができ, その時間計算量は, 順序木に対して $O(ha) + O(ha^2) + O(ha) + O(n+m) = O(ha^2 + n + m)$, 無順序木に対して $O(ha) + O(ha^2) + O(ha^2) + O(n+m) = O(ha^2 + n + m)$ となる. □

4.2　アンカーアライメント距離

本節では, アンカーアライメント距離を定式化し, そのアンカーアライメント距離を求めるアルゴリズムを設計する.

定義 4.16 (マッピングによるアンカーアライメント距離)**.** T_1, T_2 を木とし, $M \in \mathcal{M}_{\mathrm{TAI}}(T_1, T_2)$ とする. また, γ をコスト関数とする. このとき, γ 下での, T_1, T_2 間の M **によるアンカーアライメント距離** (*anchored alignment distance through* M) $\tau^{\gamma}_{\mathrm{ACH}}(T_1, T_2, M)$ を以下の通り定義する. ここで, $ach(T_1, T_2, M)$ は, アンカーアライメント問題で得られるアライメン

ト木を表す. ただし, アンカーアライメント問題が “no” を返す場合, $ach(T_1, T_2, M) = \emptyset$ となる.

$$\tau^{\gamma}_{\mathrm{ACH}}(T_1, T_2, M) = \begin{cases} \gamma(ach(T_1, T_2, M)), & ach(T_1, T_2, M) \neq \emptyset, \\ |T_1| + |T_2|, & \text{それ以外.} \end{cases}$$

定理 2.14.2 より, $M \in \mathcal{M}_{\mathrm{LESS}}(T_1, T_2) \Leftrightarrow ach(T_1, T_2, M) \neq \emptyset$ が成り立つ. AchAln の手順 1, 2 により, $M \in \mathcal{M}_{\mathrm{LESS}}(T_1, T_2)$ かどうかを $O(H|M|)$ 時間で判別できる. 加えて, 補題 3.3 の 1 より, $M \in \mathcal{M}_{\mathrm{ILST}}(T_1, T_2)$ ならば $M \in \mathcal{M}_{\mathrm{LESS}}(T_1, T_2)$ が成り立つ.

定理 4.17. 木 T_1, T_2 とコスト関数 γ に対して, $M_1 \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \gamma)$, $M_2 \in \mathcal{M}^*_{\mathrm{LESS}}(T_1, T_2, \gamma)$ とする. もし, $M_1 \subset M_2$ ならば, すべての $(v, w) \in M_2 \backslash M_1$ に対して, $(v_1, w_1), (v_2, w_2) \in M_1$ が存在し, 以下のいずれかが成り立つ.

1. $v < v_1 \sqcup v_2$ かつ $w \# w_1 \sqcup w_2$.
2. $v \# v_1 \sqcup v_2$ かつ $w < w_1 \sqcup w_2$.

[**証明**]. 関係式 1, 2 のどちらも成り立たないと仮定する. このとき, すべての $(v_1, w_1), (v_2, w_2) \in M_1$ に対して, (1) $v < v_1 \sqcup v_2$ かつ $w \leq w_1 \sqcup w_2$, $w = w_1 \sqcup w_2$, $w_1 \sqcup w_2 \leq w$ のうちどれか 1 つ (2) $w < w_1 \sqcup w_2$ かつ $v \leq v_1 \sqcup v_2$, $v = v_1 \sqcup v_2$, $v_1 \sqcup v_2 \leq v$ のうちどれか 1 つが成り立つ. 先祖子孫関係により, $v < v_1 \sqcup v_2 \iff w < w_1 \sqcup w_2$, すなわち $M_2 \in \mathcal{M}_{\mathrm{ILST}}(T_1, T_2)$ が成り立つ. しかし, $M_1 \subset M_2$ かつ $M_1 \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \gamma)$ なので, これは矛盾する. □

定理 4.18. $M_1 \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \gamma)$ と $M_2 \in \mathcal{M}^*_{\mathrm{LESS}}(T_1, T_2, \gamma)$ に対して, $M_1 \subset M_2$, $M_2 \subset M_1$ のどちらも満たさないような木 T_1, T_2 とコスト関数 γ が存在する. これは T_1, T_2 がラベルのない (あるいは単一のラベルからなる) 木であっても成立する.

[**証明**]. μ を単一コスト関数とする. まず, 図 4.7 (上) の木 T_1, T_2 を考える. 図 4.7 (下) は $M_1 \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \mu)$, $M_2 \in \mathcal{M}^*_{\mathrm{LESS}}(T_1, T_2, \mu)$ を示している. このとき, $\mu(M_1) = 3$, $\mu(M_2) = 2$ であるが, $M_1 \subset M_2$ でも $M_2 \subset M_1$ でもない.

同様に, 図 4.8 (上) の単一ラベル木 T_3, T_4 を考える. 図 4.8 (下) は $M_3 \in \mathcal{M}^*_{\mathrm{ILST}}(T_3, T_4, \mu)$, $M_4 \in \mathcal{M}^*_{\mathrm{LESS}}(T_3, T_4, \mu)$ を示している. このとき, $\mu(M_3) = 4, \mu(M_4) = 2$ であるが, $M_3 \subset M_4$ でも $M_4 \subset M_3$ でもない. □

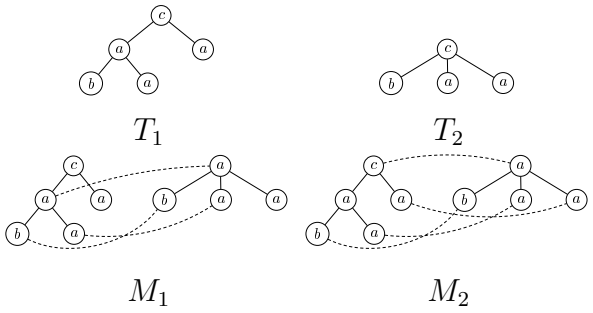


図 4.7: 定理 4.18 の証明に用いる木 T_1, T_2 (上) と, $M_1 \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \mu)$, $M_2 \in \mathcal{M}^*_{\mathrm{LESS}}(T_1, T_2, \mu)$ (下).

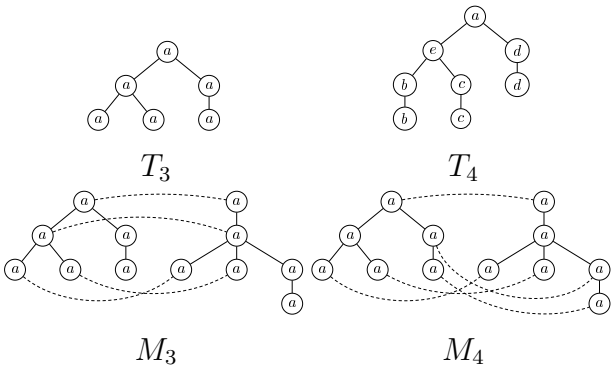


図 4.8: 定理 4.18 の証明に用いる単一ラベル木 T_3, T_4 (上) と, $M_3 \in \mathcal{M}^*_{\mathrm{ILST}}(T_3, T_4, \mu)$, $M_4 \in \mathcal{M}^*_{\mathrm{LESS}}(T_3, T_4, \mu)$ (下).

定理 4.18 は, 最小コスト劣制限マッピングと最小コスト孤立部分木マッピングは, (集合の包含関係の意味で) 比較可能とは限らないということを主張している. 一方, $\mathcal{M}_{\mathrm{ILST}}$ は Tai マッピング階層 [29, 57] の中で, 最も $\mathcal{M}_{\mathrm{LESS}}$ に近いマッピングであり, τ_{ILST} は多項式時間計算可能な τ_{TAI} の変種の中で最も一般的な距離である [51]. そこで本節では, 定理 4.17 によって, マップしていない葉ノードの組を $M \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \gamma)$ に追加することでアンカーの候補を構築する.

$M \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \gamma)$ に対して, $\overline{M}$ を M の補集合, すなわち, $\{(v, w) \in T_1 \times T_2 \mid (v, w) \notin M\}$ とする. $\overline{M} \cap (lv(T_1) \times lv(T_2))$ を M の**総葉マッピング** (*total leaf mapping*) と定義し, $lm(M)$ で表す. また, $M' \subseteq lm(M)$ ($M' = \emptyset$ も許容する) を M の**葉マッピング** (*leaf mapping*) とする. 葉マッピングを加えたマッピング $M \cup M'$ をアンカーにすることを考えるが, $M \cup M'$ が劣制限マッピングになるとは限らない. そこで, $M \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \gamma)$ と $M' \subseteq lm(M)$ に対して, アンカー $M \cup M'$ を入力としてアルゴリズム AchAln を適用することで, アンカーアライメント $ach(T_1, T_2, M \cup M')$ を構築する.

定義 4.19 (アンカーアライメント距離)**.** T_1, T_2 を木とし, γ をコスト関数とする. このとき, γ 下における T_1, T_2 間の**アンカーアライメント距離** (*anchored alignment distance*) $\tau^{\gamma}_{\mathrm{ACH}}(T_1, T_2)$ を以下の通りに定義する.

$$\tau^{\gamma}_{\mathrm{ACH}}(T_1, T_2) = \min \left\{ \gamma(ach(T_1, T_2, N)) \;\middle|\; \begin{array}{l} M \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \gamma), M' \subseteq lm(M), \\ N = M \cup M', N \in \mathcal{M}_{\mathrm{LESS}}(T_1, T_2) \end{array} \right\}.$$

$M' \subseteq lm(M)$ かつ $M \cup M' \in \mathcal{M}_{\mathrm{LESS}}(T_1, T_2)$ となる $M' \neq \emptyset$ が存在しなかった場合, $M \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \gamma)$ であることから, M' に $\emptyset$ と取ることで $\tau^{\gamma}_{\mathrm{ACH}}(T_1, T_2) = \tau^{\gamma}_{\mathrm{ILST}}(T_1, T_2) = \gamma(ach(T_1, T_2, M))$ が成り立つ. ゆえに, 定義 4.16 の $|T_1| + |T_2|$ のケースを避けることができ, $\tau^{\gamma}_{\mathrm{ALN}}(T_1, T_2) \leq \tau^{\gamma}_{\mathrm{ACH}}(T_1, T_2) \leq \tau^{\gamma}_{\mathrm{ILST}}(T_1, T_2)$ が成り立つ.

すべてのアライメント木 $\mathcal{T}$ に対して, $(l(v), l(w)) \in \mathcal{T}$ に対応したペア (v, w) を含むマッピングであるアライメント可能マッピングを構築することができる [29]. $ach(T_1, T_2, N)$ から構築したアライメント可能マッピングのうち, $M \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \gamma)$, $M' \subseteq lm(M)$, $N = M \cup M'$, $N \in \mathcal{M}_{\mathrm{LESS}}(T_1, T_2)$ を満たす γ 下でのコストが最小のマッピングすべてからなる集合を $\mathcal{M}^*_{\mathrm{ACH}}(T_1, T_2, \gamma)$ で表す.

定理 4.20. $\tau^{\gamma}_{\mathrm{ACH}}(T_1, T_2)$ は $O(nm(d + H2^{\eta}))$ 時間で計算可能である. ここで, $n = |T_1|$, $m = |T_2|$, $d = \min\{d(T_1), d(T_2)\}$, $H = \max\{h(T_1), h(T_2)\}$, $\eta = \min\{|lv(T_1)|, |lv(T_2)|\}$ で

ある.

[**証明**]. アルゴリズム 2 のアルゴリズム AchAlnDist について考える. ここで, アルゴリズム Ilst(T_1, T_2, γ) は 1 行目で孤立部分木距離 $d = \tau^{\gamma}_{\text{ILST}}(T_1, T_2)$ と最小コスト孤立部分木マッピング $M \in \mathcal{M}^{*}_{\text{ILST}}(T_1, T_2, \gamma)$ [51] の組を, $O(nmd)$ 時間で得る. 4 行目で, $M \cup M'$ が劣制限マッピングでない場合を無視している. 補題 3.3 の 1 より, $M \cup M' \in \mathcal{M}_{\text{LESS}}(T_1, T_2)$ なる $M' (\neq \emptyset)$ が存在しないならば, $\tau^{\gamma}_{\text{ACH}}(T_1, T_2) = \tau^{\gamma}_{\text{ILST}}(T_1, T_2)$ となるが, これは, 1 行目と 5 行目で対応している. 3 行目の時間計算量は, 定理 4.15 より $O(H|M|^2 + n + m) = O(mnH)$ で, 2 行目の M' の数は高々2^{η} であるため, 時間計算量は $O(nm(d + H2^{\eta}))$ となる. □

```
   procedure AchAlnDist(T1, T2, γ)
   |  /* T1, T2 : 木, γ: コスト関数 */
1  |  (d, M) ← Ilst(T1, T2, γ);
   |  /* d = τ^γ_ILST(T1, T2), M ∈ M*_ILST(T1, T2, γ) */
2  |  foreach M' ⊆ lm(M) s.t. M ∪ M' ∈ M_TAI(T1, T2) do
3  |  |  T ← AchAln(T1, T2, M ∪ M');
   |  |  /* M ∪ M' が劣制限でないなら T = ∅ */
4  |  |  if γ(T) > 0 then
   |  |  |  /* γ(T) = 0 ⟺ T = ∅ */
5  |  |  |  d ← min{d, γ(T)};
6  |  output d;
```

アルゴリズム 2: AchAlnDist.

4.3　アンカーアライメント距離の計算機実験

定理 4.20 より, アルゴリズム AchAlnDist の実行には葉の数の指数時間がかかるが, その部分はアルゴリズム AchAlnDist の 2 行目の M' によるものであり, 孤立部分木マッピングは多くの葉を含みうるため葉マッピングの数は非常に小さくなると考えられる. 本節では, アンカーアライメント距離と, 他の距離を比較するため, N 型糖鎖データとランダ

ム生成木を用いた実験を行う．以降, コスト関数を単一コスト μ と仮定し, コスト関数の添え字を省略する．また, 実験機環境は, CPU が Intel Xeon E51650 v3 (3.50GHz), RAM が 1GB, OS が Ubuntsu Linux (64bit) である．

まず, KEGG [26] の提供する N 型糖鎖データを用いた実験を行う．木の総数は $2,142$ であり, 総当たりの組数は $2,293,011$ となる．ノードと葉の数の平均はそれぞれ 11.0696, 3.2876 であり, 高さと次数の平均はそれぞれ 5.3838, 2.0724 である．また, $d \leq 5$, $D \leq 5$ である．

表 4.9 は N 型糖鎖データのすべての組の τ_{ALN}, τ_{ACH}, τ_{ILST} の計算時間である．

表 4.9: τ_{ALN}, τ_{ACH}, τ_{ILST} の計算時間.

距離	時間 (ms)
τ_{ALN}	50,503,659
τ_{ACH}	595,188
τ_{ILST}	274,425

表 4.9 は, N 型糖鎖データに対して, 時間計算量が $O(nm(d + H2^{\eta}))$ である τ_{ACH} の総計算時間はそれほど大きくなく, 時間計算量が $O(nmD^2D!)$ である τ_{ALN} の計算時間よりも, 時間計算量が $O(mnd)$ である τ_{ILST} の計算時間に近くなった．また, τ_{ACH} の総計算時間は, τ_{ILST} の総計算時間の 3 倍以下であった．

表 4.10 では, τ_{ALN}, τ_{ACH}, τ_{ILST} の不等式すべてに対しての組数を示す．ここで, 一般に $\tau_{\mathrm{ALN}} \leq \tau_{\mathrm{ACH}} \leq \tau_{\mathrm{ILST}}$ である．

表 4.10: τ_{ALN}, τ_{ACH}, τ_{ILST} の不等式すべてに対する糖鎖の組数.

不等式	組数	%
$\tau_{\mathrm{ALN}} = \tau_{\mathrm{ACH}} = \tau_{\mathrm{ILST}}$	2,166,005	94.4612
$\tau_{\mathrm{ALN}} < \tau_{\mathrm{ACH}} = \tau_{\mathrm{ILST}}$	109,255	4.7647
$\tau_{\mathrm{ALN}} = \tau_{\mathrm{ACH}} < \tau_{\mathrm{ILST}}$	17,144	0.7477
$\tau_{\mathrm{ALN}} < \tau_{\mathrm{ACH}} < \tau_{\mathrm{ILST}}$	607	0.0265

不等式	組数	%
$\tau_{\mathrm{ALN}} < \tau_{\mathrm{ILST}}$	127,006	5.5388
$\tau_{\mathrm{ALN}} < \tau_{\mathrm{ACH}}$	109,862	4.7912
$\tau_{\mathrm{ACH}} < \tau_{\mathrm{ILST}}$	17,751	0.7741

表 4.10 より, 全体の 94.4612% にあたる $2,116,005$ 組が $\tau_{\mathrm{ALN}} = \tau_{\mathrm{ACH}} = \tau_{\mathrm{ILST}}$ であり, 全

体の 99.2259% にあたる 2,275,260 組が $\tau_{\text{ACH}} = \tau_{\text{ILST}}$ であった. また, $\tau_{\text{ALN}} = \tau_{\text{ACH}} < \tau_{\text{ILST}}$ となる (すなわち, アンカーアライメントが効果的に機能した) 組の数は 17,144 で, 全体の 0.7477% であった. 一方, 定理 4.18 に対応する $\tau_{\text{ALN}} < \tau_{\text{ACH}}$ となる組の数は 109,862 で, 全体の 4.7912% であった.

表 4.11 は, AchAlnDist の 2 行目の条件を満たす M' の数ごとの組数を示している. 表 4.11 より, 全体の 99.2233% にあたる 2,275,201 組がアルゴリズム AchAlnDist2 行目の M' を選択していないということになる.

表 4.11: AchAlnDist の 2 行目の条件を満たす M' の数ごとの組数.

$\#M'$	#pairs	$\#M'$	#pairs	$\#M'$	#pairs
0	2,275,201	5	110	11	1
1	11,107	6	88	12	15
2	3,699	7	2	20	3
3	2,408	8	2	42	3
4	372				

M' の数は理論上の最悪量である $O(2^{\eta})$ より小さく, これは表 4.9 で示した τ_{ACH} の計算時間にも表れている. また, これは $\tau_{\text{ACH}} = \tau_{\text{ILST}}$ となる組数が表 4.10 で示した全組数に非常に近いことの理由と考えられる. さらに, $\#M' \geq 8$ となる 24 組に対して, $\tau_{\text{ALN}} = \tau_{\text{ACH}} < \tau_{\text{ISLT}}$ が成り立つ.

表 4.12 は, 表 4.10 で示す (1) $\tau_{\text{ALN}} < \tau_{\text{ACH}} = \tau_{\text{ISLT}}$, (2) $\tau_{\text{ALN}} = \tau_{\text{ACH}} < \tau_{\text{ISLT}}$, (3) $\tau_{\text{ALN}} < \tau_{\text{ACH}} < \tau_{\text{ISLT}}$ の場合に対して, 差の平均と最大値と, 差が最大の組についてまとめたものである. ここで, 糖鎖番号についた添字は, その木のノード数を示している.

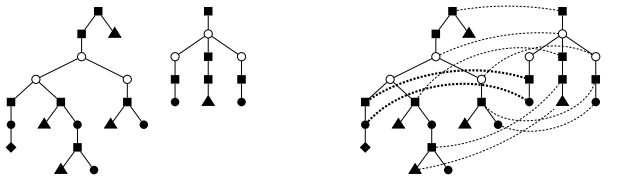
表 4.12 での出現頻度の高い糖鎖 G04570_{11} に着目する. 糖鎖 G04191_{18}, G04206_{37}, G11846_{38}, G11847_{37} と, G04570_{11} の組は $\tau_{\text{ALN}} < \tau_{\text{ACH}} < \tau_{\text{ILST}}$ を満たす. この 4 組は, 表 4.12 の (3) の $\tau_{\text{ACH}} < \tau_{\text{ILST}}$ に該当する.

図 4.13 に糖鎖 $T_1 = \text{G04191}_{18}$, $T_2 = \text{G04570}_{11}$ と, その最小コストマッピング $M_1 \in \mathcal{M}^*_{\text{LESS}}(T_1, T_2, \mu)$, $M_2 \in \mathcal{M}^*_{\text{ACH}}(T_1, T_2, \mu)$, $M_3 \in \mathcal{M}^*_{\text{ILST}}(T_1, T_2, \mu)$ を示す. ここで, 異なる

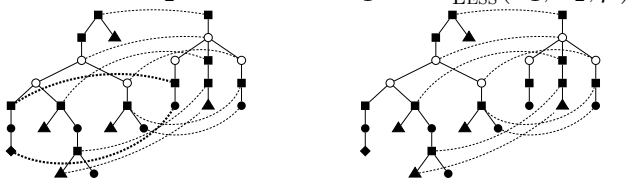
表 4.12: 各不等式を満たす組の距離の差の平均及び最大値と, 差が最大の組の数と例.

case	不等式	平均距離差	最大距離差	組数	組の例
(1)	$\tau_{\mathrm{ALN}} < \tau_{\mathrm{ACH}}$	1.0644	7	2	(G06867$_{28}$,G11335$_{19}$),(G06867$_{28}$,G11339$_{20}$)
(2)	$\tau_{\mathrm{ACH}} < \tau_{\mathrm{ILST}}$	1.0319	4	4	(G03669$_{17}$,G04570$_{11}$),(G04186$_{20}$,G04570$_{11}$), (G04570$_{11}$,G04972$_{19}$),(G04570$_{11}$,G06997$_{18}$)
(3)	$\tau_{\mathrm{ALN}} < \tau_{\mathrm{ACH}}$	1.0115	3	1	(G04045$_{36}$,G05896$_{19}$)
	$\tau_{\mathrm{ACH}} < \tau_{\mathrm{ILST}}$	1.0537	3	4	(G04191$_{18}$,G04570$_{11}$),(G04206$_{37}$,G04570$_{11}$), (G04570$_{11}$,G11846$_{38}$),(G04570$_{11}$,G11847$_{37}$)
	$\tau_{\mathrm{ALN}} < \tau_{\mathrm{ILST}}$	2.0659	5	3	(G04206$_{37}$,G04570$_{11}$),(G04570$_{11}$,G11846$_{38}$), (G04570$_{11}$,G11847$_{37}$)

形状ないしは色をしたノードは, 糖鎖構造において異なり, 異なるラベルとして扱っている. また, $\tau_{\mathrm{ALN}}(T_1, T_2) = 9$, $\tau_{\mathrm{ACH}}(T_1, T_2) = 10$, $\tau_{\mathrm{ILST}}(T_1, T_2) = 13$ である. 図 4.13 では, M_1, M_2, M_3 の差を太線で表現している. このとき, アルゴリズム AchAlnDist の入力を $M_3 \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \mu)$ とすると, 下の太線で示した葉の組だけでなく, 上の太線で示したそれらの先祖の組も M_3 に加えて M_2 を出力する. 一方, M_1 の下側で太線で表された組の T_1 側のノードは葉ではないため, アルゴリズム AchAlnDist では $M_1 \in \mathcal{M}^*_{\mathrm{LESS}}(T_1, T_2, \mu)$ を見つけることができない.



$T_1 = \mathrm{G041914}$　$T_2 = \mathrm{G04570}$　$M_1 \in \mathcal{M}^*_{\mathrm{LESS}}(T_1, T_2, \mu)$



$M_2 \in \mathcal{M}^*_{\mathrm{ACH}}(T_1, T_2, \mu)$　　$M_3 \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \mu)$

図 4.13: 糖鎖 $T_1 = \mathrm{G04191}$, $T_2 = \mathrm{G04570}$ と, 最小コストマッピング $M_1 \in \mathcal{M}^*_{\mathrm{LESS}}(T_1, T_2, \mu)$, $M_2 \in \mathcal{M}^*_{\mathrm{ACH}}(T_1, T_2, \mu)$, $M_3 \in \mathcal{M}^*_{\mathrm{ILST}}(T_1, T_2, \mu)$.

最後に, 成功した $\tau_{\mathrm{ALN}} = \tau_{\mathrm{ACH}} < \tau_{\mathrm{ILST}}$ の場合を考える. これは, 表 4.12 の (2) に該当す

る．図 4.14 は, $T_1 =$ G03669, $T_2 =$ G04186, $T_3 =$ G04972, $T_4 =$ G06997 と $T =$ G04570 に対するマッピング $M_i \in \mathcal{M}^*_{\mathrm{ACH}}(T_i, T, \mu)$ $(1 \leq i \leq 4)$ である．このとき, すべての M_i は $\mathcal{M}^*_{\mathrm{ILST}}(T_i, T, \mu)$ に太線で示した組を追加することで構築される．図 4.13 とは対照的に, アンカーアライメントの計算によって, 図 4.14 で示す最小コスト劣制限マッピングを見つけることができている．これは, アンカーで与えられた葉の組が最小コスト劣制限マッピングにも含まれており, アンカーで与えられた葉の組をその先祖と取り換える必要がなかったからあると考えられる．

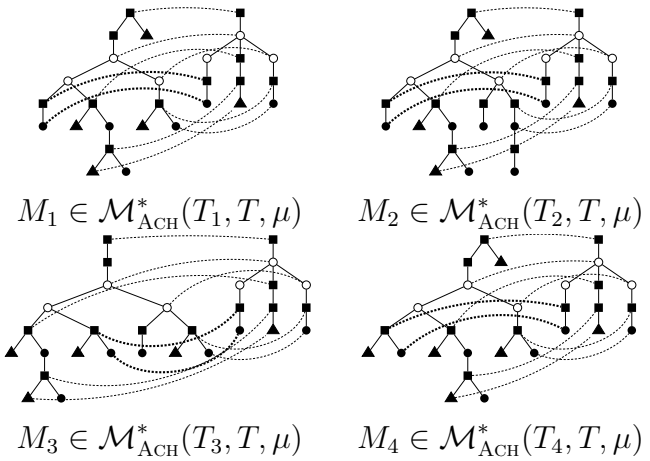


図 4.14: $T_1 =$ G03669, $T_2 =$ G04186, $T_3 =$ G04972, $T_4 =$ G06997 と $T =$ G04570 に対するマッピング $M_i \in \mathcal{M}^*_{\mathrm{ACH}}(T_i, T, \mu)$ $(1 \leq i \leq 4)$.

続いて, ランダム生成木を用いた実験を行う．アルゴリズム AchAlnDist の計算時間は, 定理 4.15 に示すとおり $O(nm(d + H2^{\eta}))$ 時間であり, 計算時間は葉の数に応じて指数的に増加する．ここで, η はアルゴリズム AchAlnDist の 2 行目の M' の数に依存しており, $|lv(T_1)| - |(M|_1) \cap lv(T_1)|$ と $|lv(T_2)| - |(M|_2) \cap lv(T_2)|$ の最小値である．そのため, η が小さい場合, τ_{ACH} は効率よく計算することができる．ランダム木での実験では, そのような状況の調査を行う．この実験では, アルゴリズム PTC [34] を用いて, 最大次数 2, 3, 4, 5, 10 のそれぞれに対してノードの数が 100 から 200 までの範囲で, 10 本の根付きラベル付き木を生成し, τ_{ACH}, τ_{ILST} を 10 本の木のすべての組み合わせ 45 組で計算する．表 4.15 は, τ_{ACH}, τ_{ILST} を計算するのに要した時間と, τ_{ACH}, τ_{ILST} の平均値, τ_{ACH}, τ_{ILST} が異なった組

の数をまとめたものである. また, この設定では最大次数が 2 であっても, τ_{ALN} を 1 日で計算し終えることができなかった.

表 4.15: τ_{ACH}, τ_{ILST} を計算するのに要した時間, τ_{ACH}, τ_{ILST} の平均値と, τ_{ACH}, τ_{ILST} が異なった組の数.

最大次数	2	3	4	5	10
τ_{ACH} 時間 (ms)	926	1,720	14,221	14,892	71,399
τ_{ILST} 時間 (ms)	719	635	609	545	552
τ_{ACH} 平均	121.93	130.87	133.30	126.51	133.89
τ_{ILST} 平均	121.93	131.22	133.20	127.60	136.00
$\tau_{\mathrm{ACH}} < \tau_{\mathrm{ILST}}$	0	10	21	25	34
	0%	22.22%	46.67%	55.56%	75.56%

表 4.15 より, 最大次数が増加しても距離の平均は変化していないが, 最大次数が増加するにつれ, 実行時間は指数的に増えており, また $\tau_{\mathrm{ACH}} < \tau_{\mathrm{ILST}}$ となる組も増加している.

4.4　巡回的順序木の木アライメント距離

本章では, 順序木, 巡回的順序木におけるアライメント距離を求めるアルゴリズムを説明する.

まず, 順序木アライメント距離を求めるアルゴリズム [21] を示す. 図 4.16 は, 空の木もしくは森を含んだ場合のアライメント距離 τ^{o}_{ALN} と森アライメント距離 $\delta^{o}_{\mathrm{ALN}}$ の再帰式である.

次に, 順序木アライメント距離を計算する再帰式 τ^{o}_{ALN}, $\delta^{o}_{\mathrm{ALN}}$ を図 4.17 に示す.

定理 4.21 ([21])**.** $\tau^{o}_{\mathrm{ALN}}(T_1, T_2)$ の時間計算量は $O(nmD^2)$ である.

この順序木アライメント距離を求めるアルゴリズムを用いて, 巡回的順序木のアライメント距離を求めるアルゴリズムを構築する. 以降, $\mathtt{B} \in \{o, b, c, cb\}$ の表記を用いて巡回的順序木を表す.

$$
\begin{aligned}
\delta^o_{\mathrm{ALN}}(\emptyset,\emptyset) &= 0,\\
\tau^o_{\mathrm{ALN}}(T_1[i],\emptyset) &= \delta^o_{\mathrm{ALN}}(T_1(i),\emptyset)+\gamma(i,\varepsilon), \delta^o_{\mathrm{ALN}}(T_1(i),\emptyset)=\sum_{k=1}^{d(i)}\tau^o_{\mathrm{ALN}}(T_1[i_k],\emptyset),\\
\tau^o_{\mathrm{ALN}}(\emptyset,T_2[j]) &= \delta^o_{\mathrm{ALN}}(\emptyset,T_2(j))+\gamma(\varepsilon,j), \delta^o_{\mathrm{ALN}}(\emptyset,T_2(j))=\sum_{k=1}^{d(j)}\tau^o_{\mathrm{ALN}}(\emptyset,T_2[j_k]).
\end{aligned}
$$

図 4.16: $\tau^o_{\mathrm{ALN}}(T_1,T_2)$ を計算する基本再帰式.

$$
\tau^o_{\mathrm{ALN}}(T_1[i],T_2[j]) = \min\left\{
\begin{array}{l}
\delta^o_{\mathrm{ALN}}(T_1(i),T_2(j))+\gamma(i,j),\\
\tau^o_{\mathrm{ALN}}(T_1[i],\emptyset)+\min\limits_{T\in T_1(i)}\{\tau^o_{\mathrm{ALN}}(T,T_2[j])-\tau^o_{\mathrm{ALN}}(T,\emptyset)\},\\
\tau^o_{\mathrm{ALN}}(\emptyset,T_2[j])+\min\limits_{T\in T_2(j)}\{\tau^o_{\mathrm{ALN}}(T_1[i],T)-\tau^o_{\mathrm{ALN}}(\emptyset,T)\}
\end{array}
\right\},
$$

$$
\delta^o_{\mathrm{ALN}}(F_1(i_1,i_s),F_2(j_1,j_t)) = \min\left\{
\begin{array}{l}
\delta^o_{\mathrm{ALN}}(F_1(i_1,i_{s-1}),F_2(j_1,j_t))+\tau^o_{\mathrm{ALN}}(T_1[i_s],\emptyset),\\
\delta^o_{\mathrm{ALN}}(F_1(i_1,i_s),F_2(j_1,j_{t-1}))+\tau^o_{\mathrm{ALN}}(\emptyset,T_2[j_t]),\\
\delta^o_{\mathrm{ALN}}(F_1(i_1,i_{s-1}),F_2(j_1,j_{t-1}))+\tau^o_{\mathrm{ALN}}(T_1[i_s],T_2[j_t]),\\
\gamma(i_s,\varepsilon)+\min\limits_{1\le k<t}\left\{
\begin{array}{l}
\delta^o_{\mathrm{ALN}}(F_1(i_1,i_{s-1}),F_2(j_1,j_{k-1}))\\
+\delta^o_{\mathrm{ALN}}(T_1(i_s),F_2(j_k,j_t))
\end{array}
\right\},\\
\gamma(\varepsilon,j_t)+\min\limits_{1\le k<s}\left\{
\begin{array}{l}
\delta^o_{\mathrm{ALN}}(F_1(i_1,i_{k-1}),F_2(j_1,j_{t-1}))\\
+\delta^o_{\mathrm{ALN}}(F_1(i_k,i_s),T_2(j_t))
\end{array}
\right\}
\end{array}
\right\}.
$$

図 4.17: 巡回的順序木間の $\tau^o_{\mathrm{ALN}}(T_1,T_2)$ を計算する再帰式.

$\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1[i],T_2[j])$ と $\delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_1,i_s),F_2(j_1,j_t))$ を計算する再帰式の中で, $T_1^p(i)$, $T_1^p(i_s)$, $T_2^q(j)$, $T_2^q(j_t)$ の表記を用いる. ここで, (1) $\mathtt{B}=o$ のとき $p=q=1$, (2) $\mathtt{B}=b$ のとき $p=\pm1$, $q=\pm1$, (3) $\mathtt{B}=c$ のとき $1\le p\le s$, $1\le q\le t$, (4) $\mathtt{B}=cb$ のとき $1\le p\le s$, $-s\le p\le -1$, $1\le q\le t$, $-t\le q\le -1$ である. よって, 次のような集合を用意する: (1) $o(s)=o(t)=\{1\}$, (2) $b(s)=b(t)=\{-1,1\}$, (3) $c(s)=\{1,\ldots,s\}$, $c(t)=\{1,\ldots,t\}$, (4) $cb(s)=\{-s,\ldots,-1,1,\ldots,s\}$, $cb(t)=\{-t,\ldots,-1,1,\ldots,t\}$. これらをまとめて, $\mathtt{B}(s)$, $\mathtt{B}(t)$ で表すこととする.

これらの集合 $\mathtt{B}(s)$, $\mathtt{B}(t)$ を用いて [21] にて与えられた再帰式を拡張し, $\mathtt{B}\in\{o,b,c,cb\}$ における, 巡回的順序木 T_1, T_2 間の $\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1,T_2)$ を計算する再帰式を設計したものを図 4.18 に示す.

$$\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1[i],T_2[j]) = \min\left\{\begin{array}{l}\min_{p\in\mathtt{B}(s),q\in\mathtt{B}(t)}\left\{\delta^{\mathtt{B}}_{\mathrm{ALN}}(T_1^p(i),T_2^q(j))+\gamma(i,j)\right\},\\ \tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1[i],\emptyset)+\min\limits_{T\in T_1(i)}\{\tau^{\mathtt{B}}_{\mathrm{ALN}}(T,T_2[j])-\tau^{\mathtt{B}}_{\mathrm{ALN}}(T,\emptyset)\},\\ \tau^{\mathtt{B}}_{\mathrm{ALN}}(\emptyset,T_2[j])+\min\limits_{T\in T_2(j)}\{\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1[i],T)-\tau^{\mathtt{B}}_{\mathrm{ALN}}(\emptyset,T)\}\end{array}\right\},$$

$$\delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_1,i_s),F_2(j_1,j_t)) = \min\left\{\begin{array}{l}\delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_1,i_{s-1}),F_2(j_1,j_t))+\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1[i_s],\emptyset),\\ \delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_1,i_s),F_2(j_1,j_{t-1}))+\tau^{\mathtt{B}}_{\mathrm{ALN}}(\emptyset,T_2[j_t]),\\ \delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_1,i_{s-1}),F_2(j_1,j_{t-1}))+\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1[i_s],T_2[j_t]),\\ \gamma(i_s,\varepsilon)+\min\limits_{1\le k<t,p\in\mathtt{B}(s)}\left\{\begin{array}{l}\delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_1,i_{s-1}),F_2(j_1,j_{k-1}))\\ +\delta^{\mathtt{B}}_{\mathrm{ALN}}(T_1^p(i_s),F_2(j_k,j_t))\end{array}\right\},\\ \gamma(\varepsilon,j_t)+\min\limits_{1\le k<s,q\in\mathtt{B}(t)}\left\{\begin{array}{l}\delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_1,i_{k-1}),F_2(j_1,j_{t-1}))\\ +\delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_k,i_s),T_2^q(j_t))\end{array}\right\}\end{array}\right\}.$$

図 4.18: 巡回的順序木間の $\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1,T_2)$ を計算する再帰式.

定理 4.22. 図 4.18 の再帰式は, $\mathtt{B}\in\{b,c,cb\}$ に対する巡回的順序木 T_1, T_2 間のアライメント距離 $\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1,T_2)$ を計算する.

[**証明**]. $\tau^{o}_{\mathrm{ALN}}(T_1,T_2)$ を計算する再帰式の正当性を示した Jiang ら [21] の証明において, 再帰式と最適アライメント木 (もしくは森)$\mathcal{T}$ との対応は以下の通りである.

1. 式 $\delta^{o}_{\mathrm{ALN}}(T_1(i),T_2(j))+\gamma(i,j)$ は, ラベル (i,j) が $T_1[i]$, $T_2[j]$ の最適アライメント木 $\mathcal{T}$ の中に含まれており, $\mathcal{T}$ が $T_1(i)$ と $T_2(j)$ のアライメントを含む場合に対応する.

2. 式 $\gamma(i_s,\varepsilon)+\min\limits_{1\le k<t}\{\delta^{o}_{\mathrm{ALN}}(F_1(i_1,i_{s-1}),F_2(j_1,j_{k-1}))+\delta^{o}_{\mathrm{ALN}}(T_1(i_s),F_2(j_k,j_t)))\}$ は, ラベル (i_s,ε) が $F_1(i_1,i_s)$, $F_2(j_1,j_t)$ の最適アライメント森 $\mathcal{T}$ に含まれており, $\mathcal{T}$ が $1\le k<t$ に対し, $T_1(i_s)$ と $F_2(j_k,j_t)$ のアライメントを含む場合に対応する.

3. 式 $\gamma(\varepsilon,j_t)+\min\limits_{1\le k<s}\{\delta^{o}_{\mathrm{ALN}}(F_1(i_1,i_{k-1}),F_2(j_1,j_{t-1}))+\delta^{o}_{\mathrm{ALN}}(F_1(i_k,i_s),T_2(j_t)))\}$ は, ラベル (ε,j_t) が $F_1(i_1,i_s)$, $F_2(j_1,j_t)$ の最適アライメント森 $\mathcal{T}$ に含まれており, $\mathcal{T}$ が $1\le k<s$ に対し, $F_1(i_k,i_s)$ と $T_2(j_t)$ のアライメントを含む場合に対応する.

上記 3 つの式だけが, 最適アライメント $\mathcal{T}$ が, T_1 か T_2(もしくはそのどちらも) のあるノードの兄弟を含み, それを展開していることに注意する. $\tau^{o}_{\mathrm{ALN}}(T_1,T_2)$ から $\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1,T_2)$ へ拡張するためには, 上記 3 つの式に現れる左から右の順を 2 つ以上の並びに分割すれば

よい. すなわち, $T_1(i)$, $T_2(j)$, $T_1(i_s)$, $T_2(j_t)$ を, それぞれ $T_1^p(i)$, $T_2^q(j)$, $T_1^p(i_s)$, $T_2^q(j_t)$ に置き換える. ここで, $p \in \mathtt{B}(s)$, $q \in \mathtt{B}(t)$ である. ゆえに, 上記 3 つの場合における式を以下の通りに置き換えることにより, $\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1, T_2)$ を計算することが可能になる.

1. $\min_{p\in\mathtt{B}(s),q\in\mathtt{B}(t)}\{\delta^{\mathtt{B}}_{\mathrm{ALN}}(T_1^p(i), T_2^q(j)) + \gamma(i,j)\}$.
2. $\gamma(i_s, \varepsilon) + \min_{1\le k<t, p\in\mathtt{B}(s)} \big\{\delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_1, i_{s-1}), F_2(j_1, j_{k-1})) + \delta^{\mathtt{B}}_{\mathrm{ALN}}(T_1^p(i_s), F_2(j_k, j_t)))\big\}$.
3. $\gamma(\varepsilon, j_t) +$
   $\min_{1\le k<s, q\in\mathtt{B}(t)} \big\{\delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_1, i_{k-1}), F_2(j_1, j_{t-1})) + \delta^{\mathtt{B}}_{\mathrm{ALN}}(F_1(i_k, i_s), T_2^q(j_t)))\big\}$.

□

定理 4.23. $\tau^b_{\mathrm{ALN}}(T_1, T_2)$ の時間計算量は $O(nmD^2)$ である. また, $\tau^c_{\mathrm{ALN}}(T_1, T_2)$ と $\tau^{cb}_{\mathrm{ALN}}(T_1, T_2)$ の時間計算量は $O(nmdD^3)$ である.

[**証明**]. 図 4.18 より, τ^o_{ALN} で呼び出される式の数は 3, δ^o_{ALN} では 5 である. また, τ^b_{ALN} では 6, δ^b_{ALN} では 7, τ^c_{ALN} では $d(i)d(j)+2$, δ^c_{ALN} では $d(i)+d(j)+3$, τ^{cb}_{ALN} では, $4d(i)d(j)+2$, $\delta^{cb}_{\mathrm{ALN}}$ では $2d(i)+2d(j)+3$ である.

Jiang ら [21] の証明より, $s = d(i)$, $t = d(j)$ とすると, $\delta^o_{\mathrm{ALN}}(F_1(i_{s'}, i_s), F_2(j_{t'}, j_t))$ は, $O((s-s') \times (t-t') \times ((s-s') + (t-t'))) = O(d(i)d(j)(d(i)+d(j)))$ 時間で計算可能である. また, $\delta^b_{\mathrm{ALN}}(F_1(i_{s'}, i_s), F_2(j_{t'}, j_t))$ は, $O(d(i)d(j)(d(i)+d(j)))$ 時間で計算可能である. 一方, 図 4.18 の再帰呼び出しの回数に着目すると, $\delta^c_{\mathrm{ALN}}(F_1(i_{s'}, i_s), F_2(j_{t'}, j_t))$ と $\delta^{cb}_{\mathrm{ALN}}(F_1(i_{s'}, i_s), F_2(j_{t'}, j_t))$ は, $O((s-s')d(j) \times (t-t')d(i) \times ((s-s') + (t-t'))) = O(d(i)^2 d(j)^2 (d(i)+d(j)))$ 時間で計算可能である.

したがって, [21] と同様に, 各 $(i,j) \in T_1 \times T_2$ に対する $\tau^b_{\mathrm{ALN}}(T_1[i], T_2[j])$ の計算時間も $O(d(i)d(j)(d(i)+d(j))d(i) + d(i)d(j)(d(i)+d(j))d(j)) = O(d(i)d(j)(d(i)+d(j))^2)$ となる. ゆえに, $\tau^b_{\mathrm{ALN}}(T_1, T_2)$ の時間計算量は以下のようになる.

$$\begin{aligned}\sum_{i=1}^{|T_1|}\sum_{j=1}^{|T_2|} O\Big(d(i)d(j)(d(i)+d(j))^2\Big) &\leq \sum_{i=1}^{|T_1|}\sum_{j=1}^{|T_2|} O\Big(d(i)d(j)(d(T_1)+d(T_2))^2\Big)\\ &\leq O\Big((d(T_1)+d(T_2))^2 \times \sum_{i=1}^{|T_1|} d(i) \times \sum_{j=1}^{|T_2|} d(j)\Big)\\ &\leq O(|T_1|\times|T_2|\times(d(T_1)+d(T_2))^2) = O(nmD^2).\end{aligned}$$

一方, 各 $(i,j) \in T_1 \times T_2$ に対する $\tau^{c}_{\mathrm{ALN}}(T_1[i],T_2[j])$ と $\tau^{cb}_{\mathrm{ALN}}(T_1[i],T_2[j])$ の計算時間は $O(d(i)^2d(j)^2(d(i)+d(j))d(i)+d(i)^2d(j)^2(d(i)+d(j))d(j)+d(i)d(j)) = O(d(i)^2d(j)^2(d(i)+d(j))^2)$ となる. 最後の式 $d(i)d(j)$ は, 図 4.18 の $\tau^{\mathtt{B}}_{\mathrm{ALN}}(T_1[i],T_2[j])$ における, 最初の式の時間計算量に対応する. ゆえに, $\tau^{c}_{\mathrm{ALN}}(T_1,T_2)$ と $\tau^{cb}_{\mathrm{ALN}}(T_1,T_2)$ の時間計算量は以下のようになる.

$$\begin{aligned}&\sum_{i=1}^{|T_1|}\sum_{j=1}^{|T_2|} O\left(d(i)^2d(j)^2(d(i)+d(j))^2\right)\\ \leq &\sum_{i=1}^{|T_1|}\sum_{j=1}^{|T_2|} O\left(d(i)d(j)d(T_1)d(T_2)\times(d(T_1)+d(T_2)^2)\right)\\ \leq &\, O\left(d(T_1)d(T_2)(d(T_1)+d(T_2))^2 \times \sum_{i=1}^{|T_1|} d(i) \times \sum_{j=1}^{|T_2|} d(j)\right)\\ \leq &\, O(|T_1|\times|T_2|\times d(T_1)d(T_2)(d(T_1)+d(T_2))^2) = O(nmdD^3).\end{aligned}$$

□

最後に, 両順序木アライメント距離と順序木編集距離の比較実験を行う. 4.3 章で利用した KEGG の N 型糖鎖に対し, 順序木編集距離 τ^{o}_{TAI} と両順序木アライメント距離 τ^{b}_{ALN} を総当たりで計算し, 同じ木の組に対する距離を比較する. 結果を図 4.19 に示す. 横軸が τ^{o}_{TAI}, 縦軸が τ^{b}_{ALN} を表し, 点の大きさ, 濃淡は, 該当する N 型糖鎖の組の頻度の対数を表している.

図 4.19 より, 多くの組で, $\tau^{b}_{\mathrm{ALN}} \leq \tau^{o}_{\mathrm{TAI}}$ となっていることがわかる. すべての N 型糖鎖の組について τ^{b}_{ALN} と τ^{o}_{TAI} を比較した組数を表 4.20 に示す.

表 4.20 より, N 型糖鎖の多くの組 (全体のおよそ 99.971%) で $\tau^{b}_{\mathrm{ALN}} \leq \tau^{o}_{\mathrm{TAI}}$ であることがわかる. $\tau^{b}_{\mathrm{ALN}} > \tau^{o}_{\mathrm{TAI}}$ となったのは 675 組 (全体のおよそ 0.029%) だけである. この順序木

図 4.19: 順序木編集距離と両順序木アライメント距離の比較.

編集距離と両順序アライメント距離の関係性は, 順序木編集距離と無順序木編集距離の関係性に似ており, N 型糖鎖において, τ^b_{ALN} は, τ^u_{TAI} の代替として使える可能性があるといえる.

表 4.20: τ^b_{ALN} と τ^o_{TAI} を比較した組数.

大小関係	組数
$\tau^b_{\mathrm{ALN}} > \tau^o_{\mathrm{TAI}}$	675
$\tau^b_{\mathrm{ALN}} = \tau^o_{\mathrm{TAI}}$	1193559
$\tau^b_{\mathrm{ALN}} < \tau^o_{\mathrm{TAI}}$	298777

第5章　さまざまな拡張

本章では, 木編集距離と Tai マッピングの拡張についての研究である. まず, 木編集距離を計算するための動的 A*アルゴリズムを設計を行う. 次に, Tai マッピングを根無し木へ拡張する. 最後に, 巡回的順序木と次数限定無順序木に対して, マッピングカーネルを構築する. なお, 本章の内容は論文 [17, 53, 56] に基づいている.

5.1　無順序木編集距離計算の動的 A*アルゴリズム

Horesh ら [19] は, 2 つの進化系統樹を比較するために (計算が困難である) 根付き**ラベルなし**無順序木間の編集距離を計算する **A*** **アルゴリズム** (A* *algorithm*) を開発した. この A*アルゴリズムは, 無順序木編集距離の定数倍以下となる 3 つの**下限関数** (*lower bounding functions*) を用いている. Higuchi ら [16] は, この A* アルゴリズムを根付き**ラベル付き**無順序木間の編集距離を計算するアルゴリズムに拡張している. 彼らは, 上記の 3 つの下限関数に加えて, さらに 2 つの下限関数を採用した. また, 標準的な A*アルゴリズムの連結グラフにおける最短経路を求める問題を無順序木編集距離を計算するものに変換する**編集距離探索木** (*edit distance search tree*) を導入した. そして, 上記 5 つの下限関数の最大値をヒューリスティック関数に設定し, 最良サーチにより編集距離探索木に対して必要に応じて辺を構築する A*アルゴリズムを設計した. しかし, この A*アルゴリズムは, 何度も部分的な Tai マッピングを複数回計算しているという欠点が残っている.

本節では, Higuchi ら [16] の導入した, 根付きラベル付き無順序木間の編集距離を計算

する A*アルゴリズムで用いられている編集距離探索木と 5 つの下限関数, および A*アルゴリズムを紹介し, A*アルゴリズムを, 部分計算を保持する動的 A*アルゴリズムに拡張する. また, 糖鎖データを用いて, 動的 A* アルゴリズムによる無順序木編集距離の計算時間を A*アルゴリズム [16], Shasha らの網羅的アルゴリズム [40], Fukagawa らのクリークアルゴリズム [12] による計算時間と比較する. さらに, 5 つの下限関数のどれが有効であるかを評価する.

定義 5.1 (編集距離探索木)**.** 木 T_1 と T_2 に対して, T_1 と T_2 の**編集距離探索木** (*edit distance search tree*)$ET(T_1, T_2)$ は, 深さが $|T_1| - 1$ であり, 根のラベルが 0 であり, 任意の内部ノードが $\varepsilon, 1, \ldots, |T_2| - 1$ というラベルを持つ $|T_2|$ 個の子を持つ木である.

さらに, 以下のような T_1 と T_2 のノードの組の集合 M_v が T_1 と T_2 のマッピングとなるとき, $ET(T_1, T_2)$ のノード v は**妥当である** (*valid*) という [1].

$$M_v = \bigcup_{w \in [ET(T_1,T_2),v]} \left\{ (t_1, t_2) \in T_1 \times T_2 \;\middle|\; \begin{array}{l} depth(w) = pre_{T_1}(t_1), \\ l(w) = pre_{T_2}(t_2), l(w) \neq \varepsilon \end{array} \right\}.$$

本論文では, 編集距離探索木 $ET(T_1, T_2)$ を妥当ノードのみによって構成されるものとする. よって, $ET(T_1, T_2)$ の深さは $|T_1| - 1$ となり, $ET(T_1, T_2)$ の次数は高々$|T_2|$ となる. $ET(T_1, T_2)$ の任意のノード v は, $depth(v) = pre_{T_1}(t_1)$, $l(v) = pre_{T_2}(t_2)(\neq \varepsilon)$ となる組 $(t_1, t_2) \in T_1 \times T_2$ を表しており, それはマッピングの構成要素となる.

例 5.2. 図 5.1(左) の木 T_1 と T_2 を考える. ここで, T_1 と T_2 のノードに割り振られた数は T_1 と T_2 の先行走査順である.

このとき, 図 5.1(右) は, T_1 と T_2 の編集距離探索木 $ET(T_1, T_2)$ である. 例えば, 下線が引かれたノードからなる $ET(T_1, T_2)$ のパス $\langle 0, 2, \varepsilon, \varepsilon \rangle$ は, T_1 と T_2 のマッピング $\{(0, 0), (1, 2)\}$ を表す. このパスでは, 深さ 1 のラベル 2 のノードはラベル ε の子孫しか持たない. なぜな

[1] 妥当ノードの定義は, 文献 [16] の定義には間違いが含まれていたので, 修正している. 本論文の定義ではノードの先行走査順による順番を利用している.

らば, マッピング $\{(0,0),(1,2)\}$ に $(2,1)$ も $(2,2)$ もマッピングとして加えることができないからである.

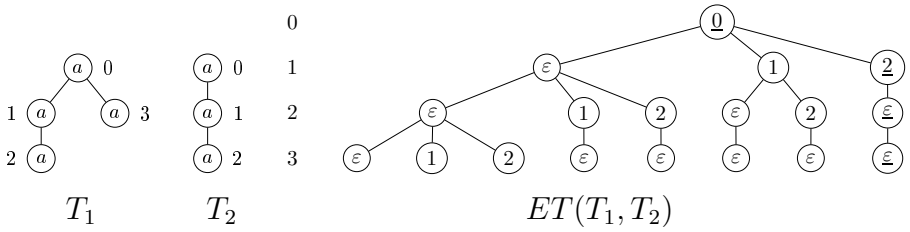


図 5.1: 例 5.2 の木 T_1 と T_2 (左) および編集距離探索木 $ET(T_1, T_2)$ (右).

次に, Higuchi ら [16] が用いた編集距離の下限関数を導入する. ここで, 木 T に対し, $d(T,k)$ で $d(v) = k$ となるノード v の頻度を表し, $l(T,a)$ で $l(v) = a$ となるノード v の頻度を表す.

定義 5.3 (編集距離の下限関数)**.** T_1 と T_2 を木とし, $D = \max\{d(T_1), d(T_2)\}$ とする. このとき, 以下の 5 つの関数を**下限関数** (*lower bounding function*) という.

1. $n(T_1, T_2) = \big||T_1| - |T_2|\big|$ [19].

2. **次数ヒストグラム L_1 距離** (*degree histogram L_1-distance*) $d_1(T_1, T_2)$ [19, 23]:

$$d_1(T_1, T_2) = \sum_{k=0}^{D} |d(T_1, k) - d(T_2, k)|.$$

3. **次数ヒストグラム L_∞ 距離** (*degree histogram L_∞-distance*) $d_\infty(T_1, T_2)$ [19]:

$$d_\infty(T_1, T_2) = \max_{0 \le k \le D} |d(T_1, k) - d(T_2, k)|.$$

4. **ラベルヒストグラム L_1 距離** (*label histogram L_1-distance*) $l_1(T_1, T_2)$ [23]:

$$l_1(T_1, T_2) = \sum_{a \in \Sigma} |l(T_1, a) - l(T_2, a)|.$$

5. **ラベルヒストグラム L_∞ 距離** (*label histogram L_∞-distance*) $l_\infty(T_1, T_2)$ [16]:

$$l_\infty(T_1, T_2) = \max_{a \in \Sigma} |l(T_1, a) - l(T_2, a)|.$$

例 5.4. 図 5.2 の木 T_1 と T_2 に対して, 次数ヒストグラムとラベルヒストグラムは表 5.3 のようになる.

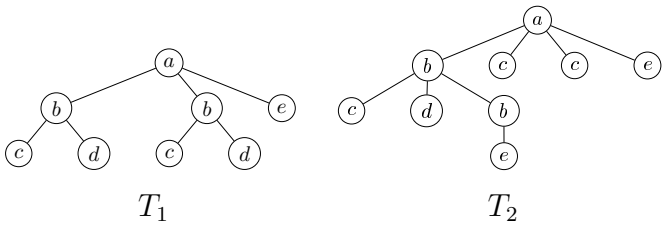


図 5.2: 例 5.4 の木 T_1 と T_2.

表 5.3: 図 5.2 の木 T_1 と T_2 の次数ヒストグラムとラベルヒストグラム.

次数	T_1	T_2
0	5	6
1	0	1
2	2	0
3	1	1
4	0	1

ラベル	T_1	T_2
a	1	1
b	2	2
c	2	3
d	2	1
e	1	2

したがって, $n(T, T_2) = 1$, $d_1(T, T_2) = 5$, $d_\infty(T, T_2) = 2$, $l_1(T, T_2) = 3$, $l_\infty(T, T_2) = 1$ となる.

補題 5.5. 木 T_1 と T_2 に対して以下が成り立つ.

1. $\tau^u_{\mathrm{TAI}}(T_1, T_2) \geq n(T_1, T_2)$ [19].

2. $\tau^u_{\mathrm{TAI}}(T_1, T_2) \geq d_1(T_1, T_2)/3$ [19, 23].

3. $\tau^u_{\mathrm{TAI}}(T_1, T_2) \geq d_\infty(T_1, T_2)$ [19].

4. $\tau^u_{\mathrm{TAI}}(T_1, T_2) \geq l_1(T_1, T_2)/2$ [23].

5. $\tau^u_{\mathrm{TAI}}(T_1, T_2) \geq l_\infty(T_1, T_2)$ [16].

これは T_1 と T_2 が森のときも成り立つ.

[証明]. 文献 [16] より, T_1 と T_2 が森のときに成り立つことを示せば十分である. T_1' と T_2' を, t_1 の子が T_1, t_2 の子が T_2, $l(t_1) = l(t_2)$ となる根 t_1 と t_2 を持つ木とする. このとき, $\tau^u_{\mathrm{TAI}}(T_1, T_2) = \tau^u_{\mathrm{TAI}}(T_1', T_2')$ となり, また, 任意の下限関数 f に対して $f(T_1, T_2) \leq f(T_1', T_2')$ となることを示すことができる. □

補題 5.5 より, 森 T_1 と T_2 に対して A* アルゴリズムにおける関数 $lb(T_1, T_2)$ を, 以下のように 5 つの下限関数の最大値と定義する.

$$lb(T_1, T_2) = \max\{n(T_1, T_2), d_1(T_1, T_2)/3, d_\infty(T_1, T_2), l_1(T_1, T_2)/2, l_\infty(T_1, T_2)\}.$$

例 5.4 の木 T_1 と T_2 に対して, $lb(T_1, T_2) = \max\{1, 5/3, 2, 3/2, 1\} = 2$ となる.

次に, 文献 [16] の A* アルゴリズム, および, その改良としての動的 A* アルゴリズムを設計する,

A* アルゴリズムの設計には, $g^*(T_1, T_2)$ と $h^*(T_1, T_2)$ という 2 つの関数を導入する必要がある. v を $depth(v) = pre_{T_1}(t_1)$, $l(v) = pre_{T_2}(t_2)$ となる $ET(T_1, T_2)$ のノードとし, v^p を $depth(v^p) = pre_{T_1}(t_1^p)$, $l(v^p) = pre_{T_2}(t_2^p)$ となる v の親ノードとする. また, T_1^m と T_2^m を, 既にマッピングの探索が終了した T_1 と T_2 のノードの集合とする. さらに, $N_v(t_1, t_2) \subseteq T_2$ を t_1 と t_2 に関するマッピングによって T_2 から削除されたノードの集合とする. このとき, $g^*(t_1, t_2)$ と $h^*(t_1, t_2)$ を以下のように計算する.

$$g^*(t_1, t_2) = g^*(t_1^p, t_2^p) + \gamma(l(t_1) \mapsto l(t_2)) + |N_v(t_1, t_2)|,$$

$$h^*(t_1, t_2) = \begin{cases} lb(T_1(t_1), S(t_2)) & \\ +lb\Big(T_1 - (T_1^m \cup V(T_1[t_1])), T_2 - (T_2^m \cup V(T_2[t_2]))\Big) & (l(t_2) \neq \varepsilon) \\ lb(T_1 - T^m, S - S^m) & (l(t_2) = \varepsilon) \end{cases}$$

このとき, A* アルゴリズムは, $ET(T_1, T_2)$ における根から葉への最短パスを探索しながら $f^*(t_1, t_2) = g^*(t_1, t_2) + h^*(t_1, t_2)$ の最小値を計算する. ここで, 編集距離探索木 $ET(T_1, T_2)$ の全体を構築すると冗長となるので, アルゴリズム 3 の A* アルゴリズム ASTAR では, 探索に必要なノードだけを構築している [2].

```
   procedure ASTAR(T1, T2)
   |  /* T1, T2 : 木, L : 3 つ組のリスト */
1  |  draw ラベルが 0 のノード v を ET(T1, T2) に追加;
2  |  (v, g*(t1, t2), h*(t1, t2)) ← (v, g*(0, 0), h*(0, 0));
3  |  while pre_T1(t1) ≠ |T1| − 1 do
4  |  |  draw ラベルが ε のノード u を ET(T1, T2) に v の子として追加;
5  |  |  L ← L ∪ {(u, g*(t1 + 1, ε), h*(t1 + 1, ε))};
6  |  |  foreach pre_T1(t1') = depth(v) + 1 を満たす (t1', t2') ∈ T1 × T2 do
7  |  |  |  if M_v ∪ {(t1', t2')} が T1, T2 間のマッピング then
8  |  |  |  |  draw pre_T2(t2') = l(u) となるノード u を ET(T1, T2) に v の子として追加;
9  |  |  |  |  L ← L ∪ {(u, g*(t1', t2'), h*(t1', t2'))};
10 |  |  select g*(t1, t2) + h*(t1, t2) が最小となる (v, g*(t1, t2), h*(t1, t2)) ∈ L;
11 |  output g*(t1, t2);
```

アルゴリズム 3: ASTAR.

アルゴリズム ASTAR では, 1 行目から 2 行目で $ET(T_1, T_2)$ の根を構築し, T_1 と T_2 のマッピングを計算している. 一方, 3 行目から 10 行目では, ("draw" 命令で)$ET(T_1, T_2)$ を構築しながら, $ET(T_1, T_2)$ の根から葉への最短パスを探している. 4 行目から 5 行目で t_1 が削除されるとき, 6 行目から 9 行目で t_1' が t_2' に対応付けられるとき, それぞれで $u \in ET(T_1, T_2)$ を構築し, そしてコストを計算する. 10 行目で最小コストのノード v を選択し, v が $ET(T_1, T_2)$ の葉であれば終了する.

例 5.6. T_1 と T_2 を図 5.4 の木とする. ここで, ノードの番号は先行走査順である. また, L をリストとし, T_1 の根は T_2 の根と対応するものとする.

このとき, A* アルゴリズムによって編集距離探索木 $ET(T_1, T_2)$ を探索することで, M がどのように構成されていくかを見ていく. ここで, $ET(T_1, T_2)$ のノードの値は $g^*(t_1, t_2) +$

[2] アルゴリズム 3 の ASTAR では, [16] の A* アルゴリズムから, 幅優先探索順 [16] の代わりに先行走査順を利用し, g^*, h^*, M_v の定義を修正している.

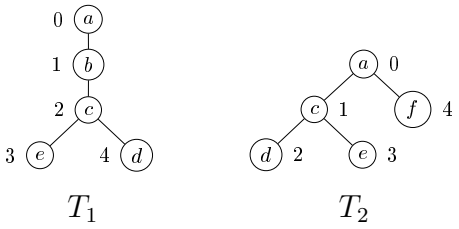
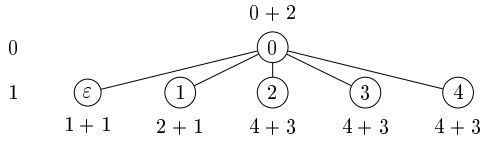


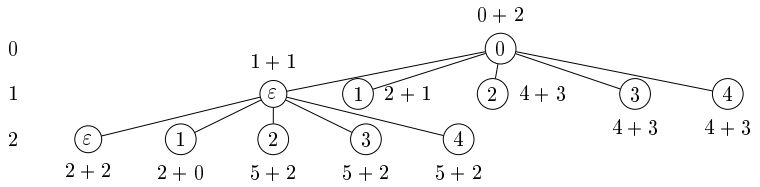
図 5.4: 木 T_1 と T_2.

$h^*(t_1, t_2)$ の値である.

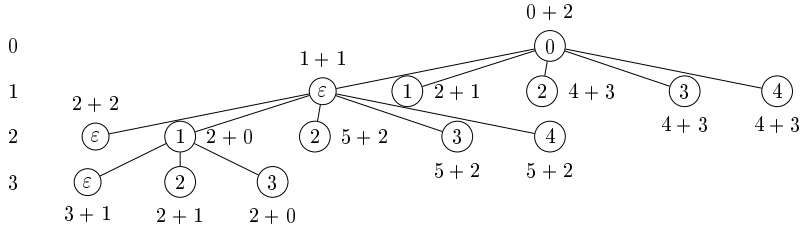
1. まず, A* アルゴリズムは M の組 $(0, 0)$ を構築し, $ET(T_1, T_2)$ に, ノード 0 から深さ 1 のノードの子を以下のように描く.



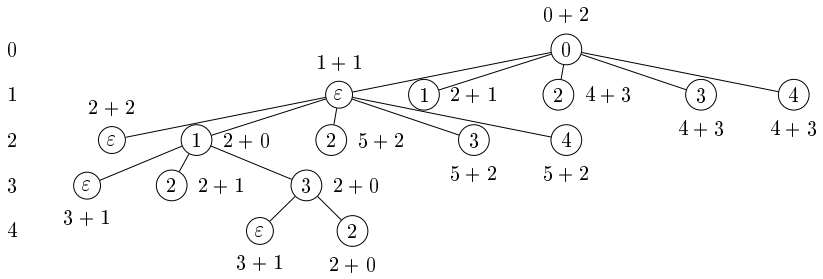
2. $ET(T_1, T_2)$ の深さ 1 のノード ε が最小値 $1 + 1$ となるので, A* アルゴリズムはパス $\langle 0, \varepsilon \rangle$ を選ぶ. 深さ 1 のノード ε は, M に組 $(1, t_2)$ が存在しないことを表している. このとき, A* アルゴリズムは, ノード ε の深さ 2 の子を以下のように描く.



3. $ET(T_1, T_2)$ の深さ 2 のノード 1 が最小値 $2 + 0$ となるので, A* アルゴリズムはパス $\langle 0, \varepsilon, 1 \rangle$ を選ぶ. 深さ 2 のノード 1 は, M が組 $(2, 1)$ を含むことを表している. A* アルゴリズムは, ノード 1 の深さ 3 の子を以下のように描く.



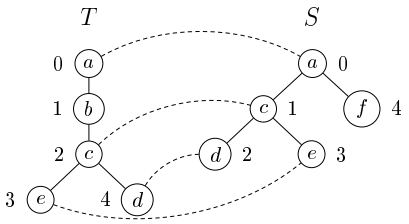
4. $ET(T_1, T_2)$ の深さ 3 のノード 3 が最小値 $2 + 0$ となるので, A* アルゴリズムはパス $\langle 0, \varepsilon, 1, 3\rangle$. を選ぶ. 深さ 3 のノード 3 は, M が組 $(3, 3)$ を含むことを表している. A* アルゴリズムは, ノード 3 の深さ 4 の子を以下のように描く.



5. $ET(T_1, T_2)$ の深さ 4 のノード 2 が最小値 $2 + 0$ となるので, A* アルゴリズムはパス $\langle 0, \varepsilon, 1, 3, 2\rangle$. を選ぶ. 深さ 4 のノード 2 は, M が組 $(4, 2)$ を含むことを表している.

6. $4 = pre_{T_1}(t_1) = |T_1| - 1$ なので, A* アルゴリズムは終了する.

したがって, A* アルゴリズムによって, マッピング $M = \{(0, 0), (2, 1), (3, 3), (4, 2)\}$ を以下のように得ることができる.



アルゴリズム ASTAR は編集距離探索木の最良先探索によって最小コストのマッピングを計算するが, ASTAR は部分マッピングの計算を何度も繰り返すという欠点がある. この欠点を解決するために, ASTAR に動的プログラミングと再帰呼び出しを組み合わせた**動的 A* アルゴリズム** (*dynamic programming* A* *algorithm*) であるアルゴリズム 4 の DPASTAR を設計する.

アルゴリズム DPASTAR では, 1 行目から 2 行目で t_1 か t_2 が葉のときを扱う. 葉は常に根と対応付けられるので, $\tau_{\mathrm{TAI}}(t_1, t_2)$ は簡単に計算できる. 6 行目から 8 行目で, DPASTAR

```
procedure DPAstar(v, t, s)
    /* v ∈ ET(T1, T2), t1 ∈ T1, t2 ∈ T2, L : 3 つ組のリスト */
1   if t1 か t2 が葉 then
2       output (v, τ(t1, t2), 0);
3   L ← {(v, g*(t1, t2), h*(t1, t2))};
4   select g*(t1, t2) + h*(t1, t2) が最小となる (v, g*(t1, t2), h*(t1, t2)) ∈ L;
5   while depth(v) ≠ |T1| − 1 do
6       if l(v) ≠ ε then
7           (v′, cost, 0) ← DPAstar(v, depth(v), l(v));
8           (v, g*(t1, t2), h*(t1, t2)) ← (v′, g*(t1, t2) + cost, 0);
9       if depth(v) = |T1| − 1 then
10          L ← L ∪ {(v, g*(t1, t2), h*(t1, t2))};
11      else
12          draw ラベルが ε となるノード u を ET(T1, T2) に v の子として追加;
13          L ← L ∪ {(u, g*(t1 + 1, ε), h*(t1 + 1, ε))};
14          foreach pre_T1(t1′) = depth(v) + 1 を満たす (t1′, t2′) ∈ T1 × T2 do
15              if Mv ∪ {(t1′, t2′)} が T1, T2 間のマッピング then
16                  draw pre_T2(t2′) = l(u) となるノード u を ET(T1, T2) に v の子として追加;
17                  L ← L ∪ {(u, g*(t1′, t2′), h*(t1′, t2′))};
18      select g*(t1, t2) + h*(t1, t2) が最小となる (v, g*(t1, t2), h*(t1, t2)) ∈ L;
19  output (v, g*(t1, t2), 0);
```

アルゴリズム 4: DPAstar.

を再帰的に計算することで, L のノード v に対する $pre_{T_1}(t_1) = depth(v)$ となる $T_1[t_1]$ と $pre_{T_2}(t_2) = l(v)$ となる $T_2[t_2]$ の最小コストマッピングを計算する. マッピングの最小コストを保持することにより, 部分マッピングの繰り返しを避けることができる. 9 行目から 10 行目で, v が最深ノードならば, v を L に再び追加する. そうでないときは, 11 行目から 18 行目で Astar と同様に $ET(T_1, T_2)$ の次のノードを探索する.

最後に, Astar [16], DPAstar, Shasha らの網羅的アルゴリズム [40], Fukagawa らのクリークアルゴリズム [12] による無順序木編集距離の計算時間を比較する. ここで, 計算機環境は以下の通りである.

アルゴリズム	OS	CPU	RAM
クリークアルゴリズム [12]	Microsoft Windows XP	Intel Core 2 Duo 2.8GHz	3.48GB
それ以外	Microsoft Windows 7	Intel Core i7 920 2.67GHz	3GB

また, [12] と同一のデータである 137 の白血病と 14 の赤血球を含む糖鎖データ, および, これらを含む 352 の糖鎖データを利用する. なお, クリークアルゴリズムの結果は文献 [12] の結果を引用している.

表 5.5 は, 4 つのアルゴリズムの無順序木編集距離の計算時間である.

表 5.5: 4 つのアルゴリズムの無順序木編集距離の計算時間 (秒).

アルゴリズム		糖鎖データ	白血病と赤血球
網羅的アルゴリズム	[40]	2133.0	751.3
ASTAR	[16]	161.6	21.4
DPASTAR		22.3	4.4
クリークアルゴリズム	[12]	—	48.3

したがって, 動的 A* アルゴリズム DPASTAR は網羅的アルゴリズム [40] と比較すると非常に高速であり, A* アルゴリズム ASTAR やクリークアルゴリズム [12] と比較しても高速である. なお, クリークアルゴリズム [12] は $\gamma(l_1, l_1) = 2$ かつ $\gamma(l_1, l_2) = 1$ $(l_1 \neq l_2)$ という特殊なコスト関数を利用しなければならないのに対して, ASTAR と DPASTAR は, 単一コスト関数だけではなく任意のコスト関数に適用することができるという利点がある.

さらに, 文献 [16] と同様に, DPASTAR における下限関数の効果について評価する. 表 5.6 は高々1 つの下限関数を除いた時の ASTAR [16] と DPASTAR の計算時間である.

したがって, 糖鎖データに対して, $n(T, S)$ が最も高速化に効果がある下限関数である. また, $d_\infty(T, S)$ と $l_\infty(T, S)$ は, $d_1(T, S)/3$ と $l_1(T, S)/2$ よりも高速化に効果がある.

5.2　Tai マッピングの根無し木への拡張

本研究では主に根付き木の距離とマッピングについて議論しているが, 進化系統樹や化合物データなど, 根無し木で表されるデータも数多く存在し, それらには根をつけるのが

表 5.6: DPAstar における下限関数の効果 (秒).

除く下限関数	糖鎖データ		白血病と赤血球	
	Astar [16]	DPAstar	Astar [16]	DPAstar
何も除かない	161.6	22.3	21.4	4.4
$n(T,S)$	338.6	43.2	41.6	7.3
$d_1(T,S)/3$	169.1	22.5	21.3	4.7
$d_\infty(T,S)$	207.1	23.4	32.6	4.9
$l_1(T,S)/2$	176.9	22.7	21.9	4.4
$l_\infty(T,S)$	187.4	23.2	24.3	5.1

妥当でない場合もある. 根付き木において Tai マッピングは木編集距離に対応する重要な概念であるが, この Tai マッピングを根無し木に拡張するためには, 一対一対応であることに加えて, 先祖子孫関係に代わる条件を導入する必要がある. Zhang ら [64] は, 挿入と削除を次数 2 以下のノードに制限した次数 2 距離 (LCA 保存距離) を根無し木に拡張する際, 根付き木の LCA 保存マッピングの拡張として, 3 つのノードの中心を保存する中心保存マッピングを導入した. ここで, $u, v, w \in T$ に対して, $[u, v], [v, w], [w, u]$ に共通するノードを $\{u, v, w\}$ の**中心** ($center$)[64] といい, $c(u, v, w)$ で表す. このとき, 以下の自明な補題を得る.

補題 5.7. 根付き木 T の根ノードを r とし, $u, v \in T$ とする. このとき $c(u, v, r) = u \sqcup v$ が成り立つ.

本節では, この中心に着目する.

葉に生物種のラベルが付いた根無し木もしくは根付き木として生物種の進化を表す進化系統樹が, すべての生物種 (葉) 間の距離である距離行列から, その距離がパス上の辺の重みの総和となるように再構成できるとき, 距離行列は**加法的である** ($additive$) という. また, 距離行列から葉の高さがすべて等しい根付き進化系統樹を構成できるとき, 距離行列は**超計量的である** ($ultrametric$) という [44]. このとき, 距離行列における任意の異なる 4

つの生物種が **4 点条件** (*4-point condition*) [6] を満たすとき, そのときに限り, 距離行列は加法的となる [44]. また, 加法的距離行列における任意の異なる 3 つの生物種が **3 点条件** (*3-point condition*) を満たすとき, そのときに限り, 距離行列は超計量的になる [44].

4 点条件と 3 点条件は, 辺の重みの総和としての距離についての条件である. 本節では, この 2 つの条件を, 中心を用いることで木のトポロジーを特徴づける条件に変更する. そして, 根無し木に対して, 4 点条件に基づく **4 点保存マッピング** (*4-point-preserving mapping*) を導入する. また, 根付き木の根ノードを 4 点条件に加えた**根付き 4 点保存マッピング** (*rooted 4-point-preserving mapping*) を導入する. さらに, 根付き木に対して, 3 点条件に基づく **3 点保存マッピング** (*3-point-preserving mapping*) を導入する. 加えて, T_1 の部分木と T_2 の部分木を対応付ける**部分木保存マッピング** (*subtree-preserving mapping*) と, ノードがパス上にある状態を保存するマッピングである**パス保存マッピング** (*path-preserving mapping*) を導入する.

これらの準備の下で, 本節では, M が根付き木のマッピングの場合, 以下が成り立つことを示す.

$$M \text{ は孤立部分木マッピング } [59,\ 60] \Leftrightarrow M \text{ は 3 点保存マッピング.}$$

$$M \text{ は劣制限マッピング } [32] \Leftrightarrow M \text{ は 4 点保存マッピング}$$

$$\Leftrightarrow M \text{ は根付き 4 点保存マッピング.}$$

また, M が根無し木のマッピングの場合, 以下の含意関係が成り立つことを示す. 一般に逆は成り立たない.

$$M \text{ は部分木保存マッピング} \Rightarrow M \text{ は中心保存マッピング}$$

$$\Rightarrow M \text{ は 4 点保存マッピング} \quad \Rightarrow M \text{ はパス保存マッピング.}$$

本節では, **マッピング**を 2 つの木のノード間の一対一対応として定義する.

定義 5.8 (マッピング [45])**.** T_1, T_2 を木とし, $M \subseteq V(T_1) \times V(T_2)$ とする. M が以下を満たすとき, 3 つ組 (M, T_1, T_2) を木 T_1, T_2 間の**マッピング** (*mapping*) という:

$$\forall (u_1, u_2), (v_1, v_2) \in M, \Big(u_1 = v_1 \iff u_2 = v_2\Big) \text{(一対一対応)}.$$

誤解の恐れがない場合, 3 つ組 (M, T_1, T_2) を, 単に M で表す. また, 根付き木 (根無し木) 間のマッピングを単に**根付き** (**根無し**) **マッピング**という.

また, 本節ではマッピング M が定義 2.7.1(劣制限マッピング) の条件を満たすとき, M を劣制限マッピングという. 特に, 今までの劣制限マッピング (M が Tai マッピングかつ定義 2.7.1 を満たす) については, **劣制限 Tai マッピング** (*Less-constrained Tai mapping*) といい $M \in \mathcal{M}_{\text{LessTai}}$ で表す.

次に, 根無しマッピングの変種と, 4 点条件と 3 点条件に関連した根付きマッピングの変種を導入する.

定義 5.9 (根無しマッピングの変種)**.** M を T_1, T_2 間の根無しマッピングとする.

1. M が以下の**パス保存条件** (*path preserving condition*) を満たすとき, M を**パス保存マッピング** (*path-preserving mapping*) といい, $M \in \mathcal{M}_{\text{Path}}(T_1, T_2)$ と表す:

$$\forall (u_1, u_2), (v_1, v_2), (w_1, w_2) \in M \Big(w_1 \in [u_1, v_1] \iff w_2 \in [u_2, v_2]\Big).$$

2. M が以下の**中心保存条件** (*center preserving condition*) [64] を満たすとき, M を中心保存マッピング center-preserving mapping といい, $M \in \mathcal{M}_{\text{Cnt}}(T_1, T_2)$ と表す:

$$\forall (u_1, u_2), (v_1, v_2), (w_1, w_2) \in M \Big((c(u_1, v_1, w_1), c(u_2, v_2, w_2)) \in M\Big).$$

3. 集合 $\{u \in T_1 \mid (u, v) \in M\}$ と $\{v \in T_2 \mid (u, v) \in M\}$ が, それぞれ $T_1 = (V_1, E_1)$ と

$T_2 = (V_2, E_2)$ の部分木を誘導し, 任意の $(u_1, u_2), (v_1, v_2) \in M$ に対して, $(u_1, v_1) \in E_1 \iff (u_2, v_2) \in E_2$ を満たすとき, M を**部分木保存マッピング** (*subtree-preserving mapping*) といい, $M \in \mathcal{M}_{\text{SUBT}}(T_1, T_2)$ と表す.

4. M が以下の **4 点保存条件** (*4-pointed condition*) を満たすとき, M を **4 点保存マッピング** (*4-pointed mapping*) といい, $M \in \mathcal{M}_{\text{4PNT}}(T_1, T_2)$ と表す:

   任意の $(u_1, u_2), (v_1, v_2), (w_1, w_2), (x_1, x_2) \in M$ に対して, (u_1, v_1) と (w_1, x_1) が 4 点条件を満たすとき, かつそのときに限り, (u_2, v_2) と (w_2, x_2) が 4 点条件を満たすように, $u_i, v_i, w_i, x_i (i = 1, 2)$ を置き換えることができる. 言い換えると:

   $$c(u_1, v_1, w_1) = c(u_1, v_1, x_1) \text{ かつ } c(u_1, w_1, x_1) = c(v_1, w_1, x_1)$$
   $$\iff \quad c(u_2, v_2, w_2) = c(u_2, v_2, x_2) \text{ かつ } c(u_2, w_2, x_2) = c(v_2, w_2, x_2).$$

また, 根付き木の 4 点保存条件を以下の通りに導入する.

定義 5.10 (根付き 4 点保存条件)**.** T_1, T_2 をそれぞれ, r_1, r_2 を根とする根付き木, M を T_1, T_2 間の根付きマッピングとする. M が以下の**根付き 4 点保存条件** (*rooted 4-pointed condition*) を満たすとき, M を**根付き 4 点保存マッピング** (*rooted 4-pointed mapping*) といい, $M \in \mathcal{M}_{\text{R4PNT}}(T_1, T_2)$ と表す:

任意の $(u_1, u_2), (v_1, v_2), (w_1, w_2) \in M$ に対して, (u_1, v_1) と (w_1, r_1) が 4 点条件を満たすとき, かつそのときに限り, (u_2, v_2) と (w_2, r_2) が 4 点条件を満たすように, $u_i, v_i, w_i (i = 1, 2)$ を置き換えることができる. 言い換えると:

$$c(u_1, v_1, w_1) = c(u_1, v_1, r_1) \text{ かつ } c(u_1, w_1, r_1) = c(v_1, w_1, r_1)$$
$$\iff \quad c(u_2, v_2, w_2) = c(u_2, v_2, r_2) \text{ かつ } c(u_2, w_2, r_2) = c(v_2, w_2, r_2).$$

M が Tai マッピングであり, 根付き 4 点保存マッピングでもあるとき, すなわち, $M \in$

$\mathcal{M}_{\mathrm{TAI}}(T_1, T_2) \cap \mathcal{M}_{\mathrm{R4PNT}}(T_1, T_2)$ であるとき, M を**根付き 4 点保存 Tai マッピング** (*rooted 4-pointed Tai mapping*) といい, $M \in \mathcal{M}_{\mathrm{R4PNTTAI}}(T_1, T_2)$ と表す.

例 5.11. 図 5.7 の根付きマッピング M_i $(i = 1, 2, 3)$ を考える. このとき M_1, M_2 は根付き 4 点保存マッピングであるが, M_3 については, $c(u_1, v_1, w_1) = c(u_1, v_1, r_1) = c_1$ かつ $c(u_1, w_1, r_1) = c(v_1, w_1, r_1) = r_1$ であるにもかかわらず $c(u_2, v_2, w_2) = c_2 \neq r_2 = c(u_2, v_2, r_2)$ かつ $c(u_2, w_2, r_2) = r_2 \neq c_2 = c(v_2, w_2, r_2)$ であるため, M_3 は根付き 4 点保存マッピングではない.

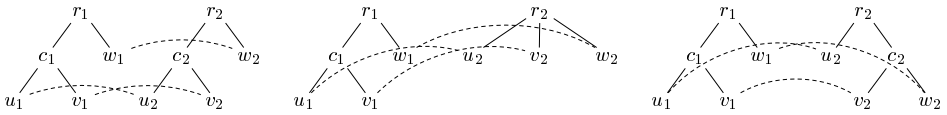


図 5.7: 例 5.11 のマッピング M_i.

定義 5.12 (3 点保存条件)**.** T_1, T_2 をそれぞれ, r_1, r_2 を根とする根付き木, M を T_1, T_2 間の根付きマッピングとする. M が以下の **3 点保存条件** (*3-pointed condition*) を満たすとき, M を **3 点保存マッピング** (*3-pointed mapping*) といい, $M \in \mathcal{M}_{3\mathrm{PNT}}(T_1, T_2)$ と表す:

> 任意の $(u_1, u_2), (v_1, v_2), (w_1, w_2) \in M$ に対して, (u_1, v_1, w_1) が 3 点条件を満たすとき, かつそのときに限り, (u_2, v_2, w_2) が 3 点条件を満たすように, $u_i, v_i, w_i (i = 1, 2)$ を置き換えることができる. 言い換えると:
>
> $$c(u_1, w_1, r_1) \leq c(v_1, w_1, r_1) = c(u_1, v_1, r_1)$$
>
> $(u_1 \sqcup w_1 \leq v_1 \sqcup w_1 = u_1 \sqcup v_1$ と等価$)$
>
> $$\Longleftrightarrow c(u_2, w_2, r_2) \leq c(v_2, w_2, r_2) = c(u_2, v_2, r_2)$$
>
> $(u_2 \sqcup w_2 \leq v_2 \sqcup w_2 = u_2 \sqcup v_2$ と等価$)$.

ここで, 根付きマッピングに対しては, 以下の階層が知られている ([29]):

$$\mathcal{M}_{\mathrm{Top}}(T_1,T_2) \subset \mathcal{M}_{\mathrm{LCA}}(T_1,T_2) \subset \mathcal{M}_{\mathrm{ILST}}(T_1,T_2) \subset \mathcal{M}_{\mathrm{LessTAI}}(T_1,T_2) \subset \mathcal{M}_{\mathrm{TAI}}(T_1,T_2).$$

まず, 3 点条件と 4 点条件に関する根付きマッピングの, 階層中での位置について議論する.

定理 5.13. 根付き木 T_1, T_2 に対して, $\mathcal{M}_{\mathrm{3PNT}}(T_1,T_2) = \mathcal{M}_{\mathrm{ILST}}(T_1,T_2)$ が成り立つ.

[**証明**]. 補題 5.7 より, 任意の $(u_1,u_2),(v_1,v_2),(w_1,w_2) \in M$ に対して, $u_1 \sqcup w_1 \leq v_1 \sqcup w_1 = u_1 \sqcup v_1 \iff u_2 \sqcup w_2 \leq v_2 \sqcup w_2 = u_2 \sqcup v_2$ であるとき, かつそのときに限り, $w_1 \leq u_1 \sqcup v_1 \iff w_2 \leq u_2 \sqcup v_2$ が成り立つことを示せば十分である.

$M \in \mathcal{M}_{\mathrm{ILST}}(T_1,T_2)$ と仮定すると, M は $w_1 \leq u_1 \sqcup v_1 \iff w_2 \leq u_2 \sqcup v_2$ を満たす. このとき, M は 図 5.8 で示す M_1, M_2, M_3 のどれかの形をとる. この内, 3 点保存条件を満たすのは M_1 と M_3 である. また, u_i を v_i と置き換えることにより, M_2 もまた, 3 点保存条件を満たす. したがって, $M \in \mathcal{M}_{\mathrm{3PNT}}(T_1,T_2)$ が成り立つ.

一方, $M \in \mathcal{M}_{\mathrm{3PNT}}(T_1,T_2)$ と仮定すると, M は $u_1 \sqcup w_1 \leq v_1 \sqcup w_1 = u_1 \sqcup v_1 \iff u_2 \sqcup w_2 \leq v_2 \sqcup w_2 = u_2 \sqcup v_2$ を満たす. このとき, M は M_1, M_2', M_3 のどれかの形をとる. M_2' について, $w_1 \leq u_1 \sqcup v_1 \iff w_2 \leq u_2 \sqcup v_2$ が成り立つ. したがって, $M \in \mathcal{M}_{\mathrm{ILST}}(T_1,T_2)$ が成り立つ. □

補題 5.14. 根付き木 T_1, T_2 に対して, $\mathcal{M}_{\mathrm{4PNT}}(T_1,T_2) = \mathcal{M}_{\mathrm{R4PNT}}(T_1,T_2)$ が成り立つ.

[**証明**]. 定義により $\mathcal{M}_{\mathrm{R4PNT}}(T_1,T_2) \subseteq \mathcal{M}_{\mathrm{4PNT}}(T_1,T_2)$ が成り立つので, $\mathcal{M}_{\mathrm{4PNT}}(T_1,T_2) \subseteq \mathcal{M}_{\mathrm{R4PNT}}(T_1,T_2)$ を示せば十分である. $M \notin \mathcal{M}_{\mathrm{4PNT}}(T_1,T_2)$ と仮定する. このとき, $c(u_1,v_1,w_1) = c(u_1,v_1,x_1)$ かつ $c(u_1,w_1,x_1) = c(v_1,w_1,x_1)$ となるが $c(u_2,v_2,w_2) \neq c(u_2,v_2,x_2)$ または $c(u_2,w_2,x_2) \neq c(v_2,w_2,x_2)$ となるような, $(u_1,u_2),(v_1,v_2),(w_1,w_2),(x_1,x_2) \in M$ が存在する. ゆえに, M は図 5.9 の M_1, M_2, M_3 のどれかの形をとる.

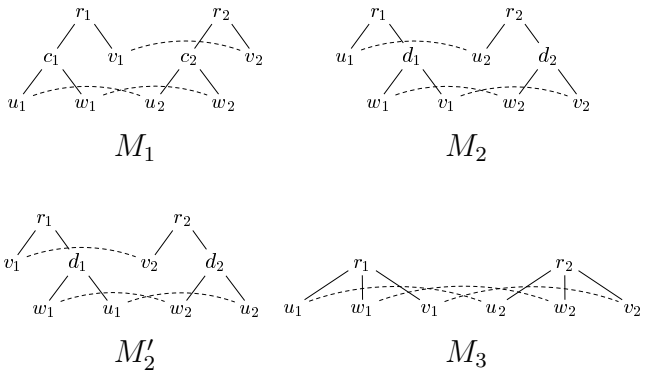


図 5.8: 定理 5.13 の証明に用いる根付きマッピング M_i.

ここで, M_1, M_2 は, $c(u_1, v_1, w_1) = c(u_1, v_1, r_1)$, かつ $c(u_1, w_1, r_1) = c(v_1, w_1, r_1)$ であるが, $c(u_2, v_2, w_2) = c_2 \neq r_2 = c(u_2, v_2, r_2)$ である. また, 定義 5.9.5 の w_2, x_2, v_2 を u_2, v_2, w_2 とみなすことにより, M_1 と M_3 は, $c(w_1, x_1, v_1) = c(w_1, x_1, r_1)$ と $c(w_1, v_1, r_1) = c(x_1, v_1, r_1)$ を満たすが, $c(w_2, v_2, r_2) = r_2 \neq d_2 = c(x_2, v_2, r_2)$ となる. ゆえに, $i = 1, 2, 3$ に対して $M_i \notin \mathcal{M}_{\text{R4PNT}}(T_1, T_2)$ となる. □

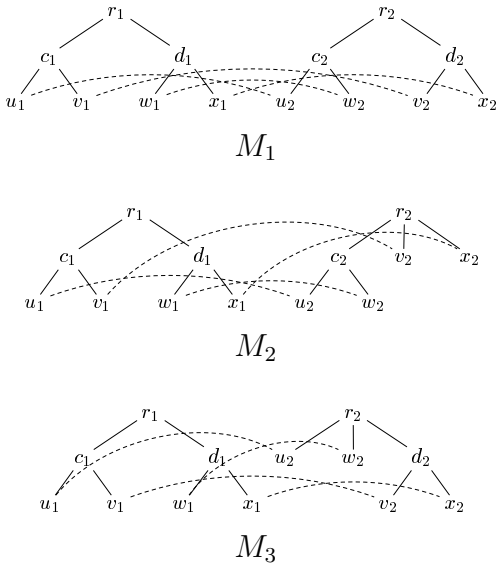


図 5.9: 補題 5.14, 5.15 の証明で用いる根付きマッピング M_i.

補題 5.15. $M \in \mathcal{M}_{\text{TAI}}(T_1, T_2) \setminus \mathcal{M}_{\text{R4PNT}}(T_1, T_2)$ なる根付きマッピング M が存在する.

[**証明**]. 図 5.9 に示すマッピング $M_i(i = 1, 2, 3)$ はすべて, $M_i \in \mathcal{M}_{\text{TAI}}(T_1, T_2) \backslash \mathcal{M}_{\text{4PNT}}(T_1, T_2)$ を満たす. □

補題 5.16. $M \in \mathcal{M}_{\text{R4PNT}}(T_1, T_2) \setminus \mathcal{M}_{\text{TAI}}(T_1, T_2)$ なる根付きマッピング M が存在する.

[**証明**]. 図 5.10 に示すマッピングは, $M \in \mathcal{M}_{\text{R4PNT}}(T_1, T_2) \setminus \mathcal{M}_{\text{TAI}}(T_1, T_2)$ を満たす. □

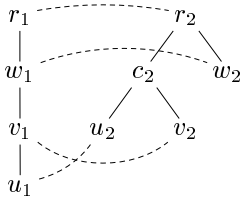


図 5.10: 補題 5.16 の証明に用いる根付きマッピング $M \in \mathcal{M}_{\text{R4PNT}}(T_1, T_2) \setminus \mathcal{M}_{\text{TAI}}(T_1, T_2)$.

定理 5.17. 根付き木 T_1, T_2 に対して, $\mathcal{M}_{\text{R4PNT}}(T_1, T_2) = \mathcal{M}_{\text{LESS}}(T_1, T_2)$ が成り立つ.

[**証明**]. 補題 5.7 より, 根付き 4 点保存条件を以下のように書き換えることができる.

$$c(u_1, v_1, w_1) = u_1 \sqcup v_1 \text{かつ } u_1 \sqcup w_1 = v_1 \sqcup w_1$$

$$\Longleftrightarrow c(u_2, v_2, w_2) = u_2 \sqcup v_2 \text{かつ } u_2 \sqcup w_2 = v_2 \sqcup w_2.$$

$M \in \mathcal{M}_{\text{R4PNT}}(T_1, T_2)$ と仮定する. もし, $u_1 \sqcup v_1 < u_1 \sqcup w_1$, ならば T_1 中の u_1, v_1, w_1 のトポロジーは, 図 5.11 (1), (2), (3) に示す通りになる. このとき (2), (3) では $u_1 < v_1$ か $v_1 < u_1$ のどちらか一方が成り立つ. よって, $c(u_1, v_1, w_1) = u_1 \sqcup v_1 < u_1 \sqcup w_1 = v_1 \sqcup w_1$ が成り立つ. 根付き 4 点保存条件より, $c(u_2, v_2, w_2) = u_2 \sqcup v_2$ 及び $u_2 \sqcup w_2 = v_2 \sqcup w_2$ が成り立つ. ゆえに, $M \in \mathcal{M}_{\text{LESS}}(T_1, T_2)$ が成り立つ.

逆に, $M \in \mathcal{M}_{\text{LESS}}(T_1, T_2)$ を仮定する. $u_1 \sqcup v_1 < u_1 \sqcup w_1$ とすると, 図 5.11 (1), (2), (3) より, $c(u_1, v_1, w_1) = u_1 \sqcup v_1$ 及び $u_1 \sqcup w_1 = v_1 \sqcup w_1$ が成り立つ. 一方,M は劣制限マッピングであるため, $v_2 \sqcup w_2 = u_2 \sqcup w_2$ が成り立つ. そのため, $c(u_2, v_2, w_2) = u_2 \sqcup v_2$ を示せばよいことになる.

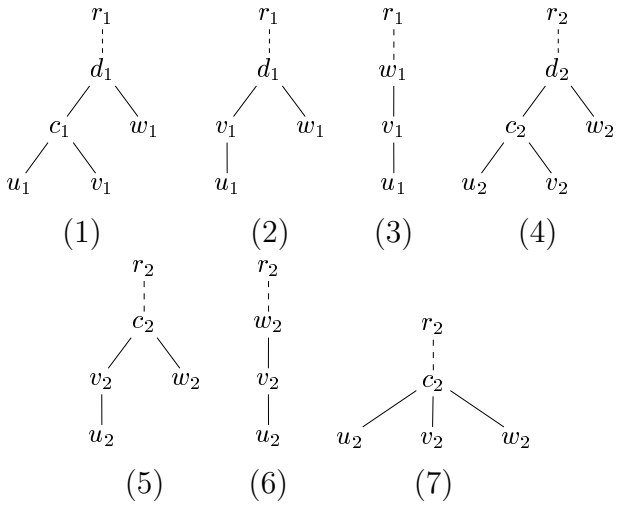


図 5.11: 定理 5.17 の証明に用いるトポロジ.

もし, $u_2 \sqcup v_2 < u_2 \sqcup w_2$ ならば, T_2 中での u_2, v_2, w_2 のトポロジーは図 5.11 (4), (5), (6) に示す通りになり, $c(u_2, v_2, w_2) = u_2 \sqcup v_2 = c_2$ が成り立つ. もし, $u_2 \sqcup v_2 = u_2 \sqcup w_2$ ならば, T_2 中での u_2, v_2, w_2 のトポロジーは図 5.11 (7) に示す通りになり, $c(u_2, v_2, w_2) = u_2 \sqcup v_2 = c_2$ が成り立つ.

したがって, $M \in \mathcal{M}_{\text{R4PNT}}(T_1, T_2)$ が成り立つ. □

注意 5.18. 定理 5.17 の $\mathcal{M}_{\text{R4PNTTAI}}(T_1, T_2) = \mathcal{M}_{\text{LESSTAI}}(T_1, T_2)$ を示すには, T_1, T_2 のトポロジーを図 5.11 の (1), (4), (7) と比較すればよい.

次に, 根無しマッピングの階層について議論する.

補題 5.19. 根無し木 T_1, T_2 に対して, $\mathcal{M}_{\text{SUBT}}(T_1, T_2) \subset \mathcal{M}_{\text{CNT}}(T_1, T_2)$ が成り立つ.

[**証明**]. $M \in \mathcal{M}_{\text{SUBT}}(T_1, T_2)$ とし, $(u_1, u_2), (v_1, v_2), (w_1, w_2) \in M$ とする. このとき, T_1 の $[u_1, v_1]$ ($[v_1, w_1]$, $[w_1, u_1]$) と, T_2 の $[u_2, v_2]$ ($[v_2, w_2]$, $[w_2, u_2]$) は同型であるので, $(c(u_1, v_1, w_1), c(u_2, v_2, w_2)) \in M$ が成り立つ. ゆえに, $M \in \mathcal{M}_{\text{CNT}}(T_1, T_2)$ が成り立つ.

一方, 図 5.12 のようなマッピング $M \in \mathcal{M}_{\text{CNT}}(T_1, T_2) \setminus \mathcal{M}_{\text{SUBT}}(T_1, T_2)$ が存在する. □

補題 5.20. 根無し木 T_1, T_2 に対して, $\mathcal{M}_{\text{CNT}}(T_1, T_2) \subset \mathcal{M}_{\text{4PNT}}(T_1, T_2)$ が成り立つ.

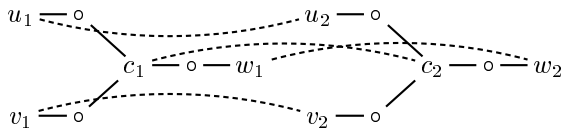


図 5.12: 補題 5.19 の証明で用いる根無しマッピング $M \in \mathcal{M}_{\text{CNT}}(T_1, T_2) \setminus \mathcal{M}_{\text{SUBT}}(T_1, T_2)$.

[**証明**]. $M \in \mathcal{M}_{\text{CNT}}(T_1, T_2)$ とし, $(u_1, u_2), (v_1, v_2), (w_1, w_2), (x_1, x_2) \in M$ とする. 補題 5.7 より, $c_1 = c(u_1, v_1, w_1) = c(u_1, v_1, x_1)$ かつ $d_1 = c(u_1, w_1, x_1) = c(v_1, w_1, x_1)$ を満たすように, u_1, v_1, w_1, x_1 を置き換えることができる. もし $(c_1, c_2) \in M$ なるノード $c_2 \in T_2$ が存在するなら, $c_1 = c(u_1, v_1, w_1) = c(u_1, v_1, x_1)$ であり M が中心保存マッピングであることから, $c_2 = c(u_2, v_2, w_2) = c(u_1, v_1, x_1)$ が成り立つ. また, $(d_1, d_2) \in M$ なるノード $d_2 \in T_2$ が存在するなら, $d_1 = c(u_1, w_1, x_1) = c(v_1, w_1, x_1)$ であり M が中心保存マッピングであることから, $d_2 = c(u_2, w_2, x_2) = c(v_2, w_2, x_2)$ が成り立つ. ゆえに, $M \in \mathcal{M}_{\text{4PNT}}(T_1, T_2)$ となる.

一方, 図 5.13 のようなマッピング $M \in \mathcal{M}_{\text{4PNT}}(T_1, T_2) \setminus \mathcal{M}_{\text{CNT}}(T_1, T_2)$ が存在する. □

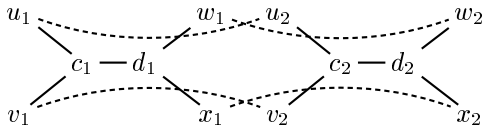


図 5.13: 補題 5.20 の証明で用いる根付きマッピング $M \in \mathcal{M}_{\text{4PNT}}(T_1, T_2) \setminus \mathcal{M}_{\text{CNT}}(T_1, T_2)$.

補題 5.21. 根付き木 T_1, T_2 に対して, $\mathcal{M}_{\text{4PNT}}(T_1, T_2) \subset \mathcal{M}_{\text{PATH}}(T_1, T_2)$ が成り立つ.

[**証明**]. $M \in \mathcal{M}_{\text{4PNT}}(T_1, T_2)$ とする. このとき, すべての $(u_1, u_2), (v_1, v_2), (w_1, w_2), (x_1, x_2) \in M$ のトポロジーは T_1, T_2 中で図 5.14 の通りになる.

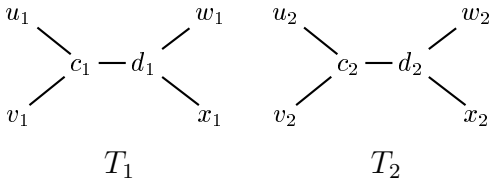


図 5.14: 補題 5.21 の証明で用いる M のトポロジー.

$(y_1, y_2) \in M$ となるような, $y_1 \in [c_1, d_1]$, $y_2 \in [u_2, c_2]$ を仮定する. このとき, $c(u_1, v_1, y_1) = c(u_1, v_1, w_1) = c_1$, $c(u_2, v_2, y_2) = y_2$, $c(u_2, v_2, w_2) = c_2$ が成り立つ. ゆえに, $y_2 = c_2$ となる.

$(y_1, y_2) \in M$ となるような, $y_1 \in [u_1, c_1]$, $y_2 \in [v_2, d_2]$ を仮定する. このとき, $c(y_1, v_1, x_1) = c(y_1, v_1, w_1) = c_1$, $c(y_2, v_2, x_2) = c(y_2, v_2, w_2) = y_2$ が成り立つ. また, $c(u_2, y_2, w_2) = c(u_2, y_2, x_2) = c_2$, $c(u_1, y_1, w_1) = c(u_1, y_1, x_1) = y_1$ が成り立つ.

$(y_1, y_2) \in M$ となるような, $y_1 \in [u_1, c_1]$, $y_2 \in [w_2, d_2]$ を仮定する. このとき, $c(y_1, v_1, x_1) = c(y_1, v_1, w_1)$, $c(y_2, v_2, x_2) = d_2$, $c(y_2, v_2, w_2) = y_2$ が成り立つ. また, $c(u_2, v_2, y_2) = c(u_2, v_2, x_2)$, $c(u_1, v_1, y_1) = y_1$, $c(u_1, v_1, x_1) = c_1$ が成り立つ. ゆえに, $y_1 = c_1$, $y_2 = d_2$ となる.

上記議論を組み合わせることにより, $(y_1, y_2) \in M$ に対して, 以下を得る. ここで, $k \in \{u, v\}$, $l \in \{w, x\}$ である.

1. $y_1 \in [c_1, d_1]$ iff $y_2 \in [c_2, d_2]$.
2. $y_1 \in [u_1, v_1]$ iff $y_2 \in [u_2, v_2]$.
3. $y_1 \in [w_1, x_1]$ iff $y_2 \in [w_2, x_2]$.
4. $y_1 \in [k_1, c_1]$ かつ $y_2 \in [l_2, d_2]$ ならば, $y_1 = c_1$ かつ $y_2 = d_2$.
5. $y_1 \in [l_1, d_1]$ かつ $y_2 \in [k_2, c_2]$ ならば, $y_1 = d_1$ かつ $y_2 = c_2$.

したがって, $M \in \mathcal{M}_{\text{PATH}}(T_1, T_2)$ となる.

一方, 図 5.15 のようなマッピング $M \in \mathcal{M}_{\text{PATH}}(T_1, T_2) \setminus \mathcal{M}_{\text{4PNT}}(T_1, T_2)$ が存在する. □

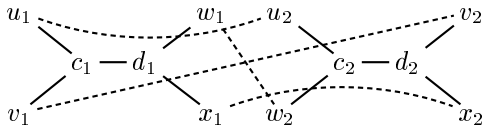


図 5.15: 補題 5.21 の証明に用いる根無しマッピング $M \in \mathcal{M}_{\text{PATH}}(T_1, T_2) \setminus \mathcal{M}_{\text{4PNT}}(T_1, T_2)$.

補題 5.19, 5.20, 5.21 をまとめると以下の通りになる.

定理 5.22. 根無し木 T_1, T_2 に対して, 以下が成り立つ.

$$\mathcal{M}_{\text{SUBT}}(T_1, T_2) \subset \mathcal{M}_{\text{CNT}}(T_1, T_2) \subset \mathcal{M}_{\text{4PNT}}(T_1, T_2) \subset \mathcal{M}_{\text{PATH}}(T_1, T_2).$$

5.3　巡回的順序木と次数限定無順序木のマッピングカーネル

無順序木におけるマッピングカーネルは, 一般には#P 完全であることが知られている [25]. 本節では, 順序木よりも制限が弱く無順序木よりも制限が強い巡回的順序木のマッピングカーネルについて解析する. 以後, 本節を通して, $\pi \in \{o, b, c, cb, u\}$ とし, $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ とする.

森 F_1 と F_2 の**マッピング**を, $(v, v) \notin M$ となる v に対する木 $v(F_1)$ と $v(F_2)$ のマッピング M として定義する. また, $\mathcal{M}^{\pi}_{\mathtt{A}}(F_1, F_2)$ を $\mathcal{M}^{\pi}_{\mathtt{A}}(T_1, T_2)$ と同様に定義する. $\sigma : \Sigma \times \Sigma \to \mathbf{R}^+$ を類似度関数とする. 木 T_1 と T_2 のマッピング $M \in \mathcal{M}^{\pi}_{\mathtt{A}}(T_1, T_2)$ の類似度 $\sigma(M)$ を $\sigma(M) = \prod_{(u,v)\in M} \sigma(l(u), l(v))$ と定義する. 森 F_1 と F_2 の**類似度** (*similarity*) を以下のように定義する.

$$\mathcal{K}^{\pi}_{\mathtt{A}}(F_1, F_2) = \sum_{M \in \mathcal{M}^{\pi}_{\mathtt{A}}(F_1, F_2)} \sigma(M).$$

系 5.23. $\pi \in \{o, b, c, cb, u\}$ と $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対して, $\mathcal{K}^{\pi}_{\mathtt{A}}$ は正定値である.

[**証明**]. $\mathcal{M}^{\pi}_{\mathtt{A}}$ は合成について閉じている [29, 57, 64] ので, 文献 [42] の結果から成り立つ. □

Kuboyama [29] は, 明示的には $\mathtt{A} \in \{\textsc{Acc}, \textsc{Ilst}\}$ に対して $\mathcal{K}^{o}_{\mathtt{A}}(T_1, T_2)$ を計算する再帰式を, 暗黙的には $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}\}$ に対して $\mathcal{K}^{o}_{\mathtt{A}}(T_1, T_2)$ を計算する再帰式を図 5.16 のように導入した. ここで, 下線の式は, 類似した式における異なる部分を明記したものである.

定理 5.24 ([29] 参照)**.** $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対して, 図 5.16 の再帰式は $O(nm)$ 時間で $\mathcal{K}^{o}_{\mathtt{A}}(T_1, T_2)$ を正しく計算する. ここで, $n = |T_1|$, $m = |T_2|$ である.

本節では, まず, 図 5.16 の再帰式を, $\pi \in \{o, b, c, cb\}$ と $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対して $\mathcal{K}^{\pi}_{\mathtt{A}}(T_1, T_2)$ を計算する再帰式に拡張する.

$$
\begin{array}{rcl}
\mathcal{K}^o_{\mathrm{Top}}(u(F_1), v(F_2)) & = & \sigma(l(u), l(v)) \cdot (1 + \mathcal{F}^o_{\mathrm{Top}}(F_1, F_2)), \\
\mathcal{F}^o_{\mathrm{Top}}(\emptyset, F) & = & \mathcal{F}^o_{\mathrm{Top}}(F, \emptyset) = 0, \\
\mathcal{F}^o_{\mathrm{Top}}(T_1 \bullet F_1, T_2 \bullet F_2) & = & \underline{\mathcal{K}^o_{\mathrm{Top}}(T_1, T_2)} \cdot (1 + \mathcal{F}^o_{\mathrm{Top}}(F_1, F_2)) \\
 & & \underline{+\mathcal{F}^o_{\mathrm{Top}}(F_1, T_2 \bullet F_2)} + \mathcal{F}^o_{\mathrm{Top}}(T_1 \bullet F_1, F_2) - \mathcal{F}^o_{\mathrm{Top}}(F_1, F_2). \\
\hline
\mathcal{K}^o_{\mathrm{LCASG}}(T_1, T_2) & = & \displaystyle\sum_{u \in T_1} \sum_{v \in T_2} \mathcal{K}^o_{\mathrm{Top}}(T_1[u], T_2[v]). \\
\hline
\mathcal{K}^o_{\mathrm{LCA}}(T_1, T_2) & = & \displaystyle\sum_{u \in T_1} \sum_{v \in T_2} \mathcal{T}^o_{\mathrm{LCA}}(T_1[u], T_2[v]), \\
\mathcal{T}^o_{\mathrm{LCA}}(u(F_1), v(F_2)) & = & \sigma(l(u), l(v)) \cdot (1 + \mathcal{F}^o_{\mathrm{LCA}}(F_1, F_2)), \\
\mathcal{F}^o_{\mathrm{LCA}}(\emptyset, F) & = & \mathcal{F}^o_{\mathrm{LCA}}(F, \emptyset) = 0, \\
\mathcal{F}^o_{\mathrm{LCA}}(T_1 \bullet F_1, T_2 \bullet F_2) & = & \underline{\mathcal{K}^o_{\mathrm{LCA}}(T_1, T_2)} \cdot (1 + \mathcal{F}^o_{\mathrm{LCA}}(F_1, F_2)) \\
 & & \underline{+\mathcal{F}^o_{\mathrm{LCA}}(F_1, T_2 \bullet F_2)} + \mathcal{F}^o_{\mathrm{LCA}}(T_1 \bullet F_1, F_2) - \mathcal{F}^o_{\mathrm{LCA}}(F_1, F_2). \\
\hline
\mathcal{K}^o_{\mathrm{ACC}}(\emptyset, F) & = & \mathcal{K}^o_{\mathrm{ACC}}(F, \emptyset) = \mathcal{TF}^o_{\mathrm{ACC}}(T, \emptyset) = 0, \\
\mathcal{K}^o_{\mathrm{ACC}}(T_1 \bullet F_1, T_2 \bullet F_2) & = & \mathcal{T}^o_{\mathrm{ACC}}(T_1, T_2) \cdot (\mathcal{F}^o_{\mathrm{ACC}}(F_1, F_2) - 1) \\
 & & +\mathcal{TF}^o_{\mathrm{ACC}}(T_1, T_2 \bullet F_2) - \mathcal{TF}^o_{\mathrm{ACC}}(T_1, F_2) \\
 & & +\mathcal{TF}^o_{\mathrm{ACC}}(T_2, T_1 \bullet F_1) - \mathcal{TF}^o_{\mathrm{ACC}}(T_2, F_1) \\
 & & +\mathcal{K}^o_{\mathrm{ACC}}(T_1 \bullet F_1, F_2) - \mathcal{K}^o_{\mathrm{ACC}}(F_1, F_2), \\
\mathcal{TF}^o_{\mathrm{ACC}}(v(F_1), T_2 \bullet F_2) & = & \mathcal{T}^o_{\mathrm{ACC}}(v(F_1), T_2) - \mathcal{TF}^o_{\mathrm{ACC}}(T_2, F_1) + \mathcal{TF}^o_{\mathrm{ACC}}(v(F_1), F_2) \\
 & & -\mathcal{K}^o_{\mathrm{ACC}}(F_1, F_2) + \mathcal{K}^o_{\mathrm{ACC}}(F_1, T_1 \bullet F_2), \\
\mathcal{T}^o_{\mathrm{ACC}}(u(F_1), v(F_2)) & = & \sigma(l(u), l(v)) \cdot (1 + \underline{\mathcal{F}^o_{\mathrm{ACC}}(F_1, F_2)}) + \mathcal{TF}^o_{\mathrm{ACC}}(u(F_1), F_2) \\
 & & +\mathcal{TF}^o_{\mathrm{ACC}}(v(F_2), F_1) \underline{- \mathcal{K}^o_{\mathrm{ACC}}(F_1, F_2)}, \\
\mathcal{F}^o_{\mathrm{ACC}}(\emptyset, F) & = & \mathcal{F}^o_{\mathrm{ACC}}(F, \emptyset) = 0, \\
\mathcal{F}^o_{\mathrm{ACC}}(T_1 \bullet F_1, T_2 \bullet F_2) & = & \mathcal{T}^o_{\mathrm{ACC}}(T_1, T_2) \cdot (1 + \mathcal{F}^o_{\mathrm{ACC}}(F_1, F_2)) \\
 & & +\mathcal{F}^o_{\mathrm{ACC}}(F_1, T_2 \bullet F_2) + \mathcal{F}^o_{\mathrm{ACC}}(T_1 \bullet F_1, F_2) - \mathcal{F}^o_{\mathrm{ACC}}(F_1, F_2). \\
\hline
\mathcal{K}^o_{\mathrm{ILST}}(\emptyset, F) & = & \mathcal{K}^o_{\mathrm{ILST}}(F, \emptyset) = \mathcal{TF}^o_{\mathrm{ILST}}(T, \emptyset) = 0, \\
\mathcal{K}^o_{\mathrm{ILST}}(T_1 \bullet F_1, T_2 \bullet F_2) & = & \mathcal{T}^o_{\mathrm{ILST}}(T_1, T_2) \cdot (\mathcal{F}^o_{\mathrm{ILST}}(F_1, F_2) - 1) \\
 & & +\mathcal{TF}^o_{\mathrm{ILST}}(T_1, T_2 \bullet F_2) - \mathcal{TF}^o_{\mathrm{ILST}}(T_1, F_2) \\
 & & +\mathcal{TF}^o_{\mathrm{ILST}}(T_2, T_1 \bullet F_1) - \mathcal{TF}^o_{\mathrm{ILST}}(T_2, F_1) \\
 & & +\mathcal{K}^o_{\mathrm{ILST}}(T_1 \bullet F_1, F_2) - \mathcal{K}^o_{\mathrm{ILST}}(F_1, F_2), \\
\mathcal{TF}^o_{\mathrm{ILST}}(v(F_1), T_2 \bullet F_2) & = & \mathcal{T}^o_{\mathrm{ILST}}(v(F_1), T_2) - \mathcal{TF}^o_{\mathrm{ILST}}(T_2, F_1) + \mathcal{TF}^o_{\mathrm{ILST}}(v(F_1), F_2) \\
 & & -\mathcal{K}^o_{\mathrm{ILST}}(F_1, F_2) + \mathcal{K}^o_{\mathrm{ILST}}(F_1, T_1 \bullet F_2), \\
\mathcal{T}^o_{\mathrm{ILST}}(u(F_1), v(F_2)) & = & \sigma(l(u), l(v)) \cdot (1 + \underline{\mathcal{K}^o_{\mathrm{ILST}}(F_1, F_2)}) + \mathcal{TF}^o_{\mathrm{ILST}}(u(F_1), F_2) \\
 & & +\mathcal{TF}^o_{\mathrm{ILST}}(v(F_2), F_1) \underline{- \mathcal{K}^o_{\mathrm{ILST}}(F_1, F_2)}, \\
\mathcal{F}^o_{\mathrm{ILST}}(\emptyset, F) & = & \mathcal{F}^o_{\mathrm{ILST}}(F, \emptyset) = 0, \\
\mathcal{F}^o_{\mathrm{ILST}}(T_1 \bullet F_1, T_2 \bullet F_2) & = & \mathcal{T}^o_{\mathrm{ILST}}(T_1, T_2) \cdot (1 + \mathcal{F}^o_{\mathrm{ILST}}(F_1, F_2)) \\
 & & +\mathcal{F}^o_{\mathrm{ILST}}(F_1, T_2 \bullet F_2) + \mathcal{F}^o_{\mathrm{ILST}}(T_1 \bullet F_1, F_2) - \mathcal{F}^o_{\mathrm{ILST}}(F_1, F_2).
\end{array}
$$

図 5.16: $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対して $\mathcal{K}^o_{\mathtt{A}}(T_1, T_2)$ を計算する再帰式 [29].

木 $u(F_1)$ と $v(F_2)$ に対して, $F_1 = [T_1[u_1], \ldots, T_1[u_s]]$, $F_2 = [T_2[v_1], \ldots, T_2[v_t]]$ とする. すなわち, $ch(u) = \{u_1, \ldots, u_s\}$, $ch(v) = \{v_1, \ldots, v_t\}$, $d(u) = s$, $d(v) = t$ となる. また, $1 \leq p \leq s$, $1 \leq q \leq t$ とする. 森 $[T_1[u_{\sigma^+_{p,s}(1)}], \ldots, T_1[u_{\sigma^+_{p,s}(s)}]]$ を F_1^p, 森 $[T_2[v_{\sigma^+_{q,t}(1)}], \ldots, T_2[v_{\sigma^+_{q,t}(t)}]]$ を F_2^q と表す. さらに, 森 $[T_1[u_{\sigma^-_{p,s}(1)}], \ldots, T_1[u_{\sigma^-_{p,s}(s)}]]$ を F_1^{-p}, 森 $[T_2[v_{\sigma^-_{q,t}(1)}], \ldots, T_2[v_{\sigma^-_{q,t}(t)}]]$ を F_2^{-q} と表す. 明らかに, $F_1 = F_1^1$, $F_2 = F_2^1$ である.

さらに, p と q の値を, (1) $\pi = o$ ならば $p = q = 1$, (2) $\pi = b$ ならば $p = \pm 1$ かつ $q = \pm 1$, (3) $\pi = c$ ならば $1 \leq p \leq s$ かつ $1 \leq q \leq t$, (4) $\pi = cb$ ならば $1 \leq p \leq s$, $-s \leq p \leq -1$, $1 \leq q \leq t$ かつ $-t \leq q \leq -1$ とする. したがって, p と q の値により, 以下のような集

合を準備する．(1) $o(s) = o(t) = \{1\}$, (2) $b(s) = b(t) = \{-1, 1\}$, (3) $c(s) = \{1, \ldots, s\}$, $c(t) = \{1, \ldots, t\}$, (4) $cb(s) = \{-s, \ldots, -1, 1, \ldots, s\}$ かつ $cb(t) = \{-t, \ldots, -1, 1, \ldots, t\}$. これらの集合を, $\pi \in \{o, b, c, cb\}$ に対して $\pi(s)$ および $\pi(t)$ と表すものとする．

このとき, $\mathcal{K}^{\pi}_{\mathtt{A}}(T_1, T_2)$ を計算する再帰式を図 5.17 のように設計することができる．

$$
\begin{array}{rl}
\mathcal{K}^{\pi}_{\textsc{Top}}(u(F_1), v(F_2)) = & \sigma(l(u), l(v)) \cdot \left(1 + \sum_{p \in \pi(d(u))} \sum_{q \in \pi(d(v))} \mathcal{F}^{\pi}_{\textsc{Top}}(F_1^p, F_2^q)\right), \\
\mathcal{F}^{\pi}_{\textsc{Top}}(\emptyset, F) = & \mathcal{F}^{\pi}_{\textsc{Top}}(F, \emptyset) = 0, \\
\mathcal{F}^{\pi}_{\textsc{Top}}(T_1 \bullet F_1, T_2 \bullet F_2) = & \mathcal{K}^{\pi}_{\textsc{Top}}(T_1, T_2) \cdot (1 + \mathcal{F}^{\pi}_{\textsc{Top}}(F_1, F_2)) \\
& + \mathcal{F}^{\pi}_{\textsc{Top}}(F_1, T_2 \bullet F_2) + \mathcal{F}^{\pi}_{\textsc{Top}}(T_1 \bullet F_1, F_2) - \mathcal{F}^{\pi}_{\textsc{Top}}(F_1, F_2).
\end{array}
$$

$$
\mathcal{K}^{\pi}_{\textsc{LcaSg}}(T_1, T_2) = \sum_{u \in T_1} \sum_{v \in T_2} \mathcal{K}^{\pi}_{\textsc{Top}}(T_1[u], T_2[v]).
$$

$$
\begin{array}{rl}
\mathcal{K}^{\pi}_{\textsc{Lca}}(T_1, T_2) = & \sum_{u \in T_1} \sum_{v \in T_2} \mathcal{T}^{\pi}_{\textsc{Lca}}(T_1[u], T_2[v]), \\
\mathcal{T}^{\pi}_{\textsc{Lca}}(u(F_1), v(F_2)) = & \sigma(l(u), l(v)) \cdot \left(1 + \sum_{p \in \pi(d(u))} \sum_{q \in \pi(d(v))} \mathcal{F}^{\pi}_{\textsc{Lca}}(F_1^p, F_2^q)\right), \\
\mathcal{F}^{\pi}_{\textsc{Lca}}(\emptyset, F) = & \mathcal{F}^{\pi}_{\textsc{Lca}}(F, \emptyset) = 0, \\
\mathcal{F}^{\pi}_{\textsc{Lca}}(T_1 \bullet F_1, T_2 \bullet F_2) = & \mathcal{K}^{\pi}_{\textsc{Lca}}(T_1, T_2) \cdot (1 + \mathcal{F}^{\pi}_{\textsc{Lca}}(F_1, F_2)) \\
& + \mathcal{F}^{\pi}_{\textsc{Lca}}(F_1, T_2 \bullet F_2) + \mathcal{F}^{\pi}_{\textsc{Lca}}(T_1 \bullet F_1, F_2) - \mathcal{F}^{\pi}_{\textsc{Lca}}(F_1, F_2).
\end{array}
$$

$$
\begin{array}{rl}
\mathcal{K}^{\pi}_{\textsc{Acc}}(\emptyset, F) = & \mathcal{K}^{\pi}_{\textsc{Acc}}(F, \emptyset) = \mathcal{TF}^{\pi}_{\textsc{Acc}}(T, \emptyset) = 0, \\
\mathcal{K}^{\pi}_{\textsc{Acc}}(T_1 \bullet F_1, T_2 \bullet F_2) = & \mathcal{T}^{\pi}_{\textsc{Acc}}(T_1, T_2) \cdot (\mathcal{F}^{\pi}_{\textsc{Acc}}(F_1, F_2) - 1) \\
& + \mathcal{TF}^{\pi}_{\textsc{Acc}}(T_1, T_2 \bullet F_2) - \mathcal{TF}^{\pi}_{\textsc{Acc}}(T_1, F_2) \\
& + \mathcal{TF}^{\pi}_{\textsc{Acc}}(T_2, T_1 \bullet F_1) - \mathcal{TF}^{\pi}_{\textsc{Acc}}(T_2, F_1) \\
& + \mathcal{K}^{\pi}_{\textsc{Acc}}(T_1 \bullet F_1, F_2) - \mathcal{K}^{\pi}_{\textsc{Acc}}(F_1, F_2), \\
\mathcal{TF}^{\pi}_{\textsc{Acc}}(v(F_1), T_2 \bullet F_2) = & \mathcal{T}^{\pi}_{\textsc{Acc}}(v(F_1), T_2) - \mathcal{TF}^{\pi}_{\textsc{Acc}}(T_2, F_1) + \mathcal{TF}^{\pi}_{\textsc{Acc}}(v(F_1), F_2) \\
& - \mathcal{K}^{\pi}_{\textsc{Acc}}(F_1, F_2) + \mathcal{K}^{\pi}_{\textsc{Acc}}(F_1, T_1 \bullet F_2), \\
\mathcal{T}^{\pi}_{\textsc{Acc}}(u(F_1), v(F_2)) = & \sigma(l(u), l(v)) \cdot \left(1 + \sum_{p \in \pi(d(u))} \sum_{q \in \pi(d(v))} \mathcal{F}^{\pi}_{\textsc{Acc}}(F_1^p, F_2^q)\right) \\
& + \sum_{q \in \pi(d(v))} \mathcal{TF}^{\pi}_{\textsc{Acc}}(u(F_1), F_2^q) + \sum_{p \in \pi(d(u))} \mathcal{TF}^{\pi}_{\textsc{Acc}}(v(F_2), F_1^p) \\
& - \sum_{p \in \pi(d(u))} \sum_{q \in \pi(d(v))} \mathcal{K}^{\pi}_{\textsc{Acc}}(F_1^p, F_2^q), \\
\mathcal{F}^{\pi}_{\textsc{Acc}}(\emptyset, F) = & \mathcal{F}^{\pi}_{\textsc{Acc}}(F, \emptyset) = 0, \\
\mathcal{F}^{\pi}_{\textsc{Acc}}(T_1 \bullet F_1, T_2 \bullet F_2) = & \mathcal{T}^{\pi}_{\textsc{Acc}}(T_1, T_2) \cdot (1 + \mathcal{F}^{\pi}_{\textsc{Acc}}(F_1, F_2)) \\
& + \mathcal{F}^{\pi}_{\textsc{Acc}}(F_1, T_2 \bullet F_2) + \mathcal{F}^{\pi}_{\textsc{Acc}}(T_1 \bullet F_1, F_2) - \mathcal{F}^{\pi}_{\textsc{Acc}}(F_1, F_2).
\end{array}
$$

$$
\begin{array}{rl}
\mathcal{K}^{\pi}_{\textsc{Ilst}}(\emptyset, F) = & \mathcal{K}^{\pi}_{\textsc{Ilst}}(F, \emptyset) = \mathcal{TF}^{o}_{\textsc{Ilst}}(T, \emptyset) = 0, \\
\mathcal{K}^{\pi}_{\textsc{Ilst}}(T_1 \bullet F_1, T_2 \bullet F_2) = & \mathcal{T}^{\pi}_{\textsc{Ilst}}(T_1, T_2) \cdot (\mathcal{F}^{\pi}_{\textsc{Ilst}}(F_1, F_2) - 1) \\
& + \mathcal{TF}^{\pi}_{\textsc{Ilst}}(T_1, T_2 \bullet F_2) - \mathcal{TF}^{\pi}_{\textsc{Ilst}}(T_1, F_2) \\
& + \mathcal{TF}^{\pi}_{\textsc{Ilst}}(T_2, T_1 \bullet F_1) - \mathcal{TF}^{\pi}_{\textsc{Ilst}}(T_2, F_1) \\
& + \mathcal{K}^{\pi}_{\textsc{Ilst}}(T_1 \bullet F_1, F_2) - \mathcal{K}^{\pi}_{\textsc{Ilst}}(F_1, F_2), \\
\mathcal{TF}^{\pi}_{\textsc{Ilst}}(v(F_1), T_2 \bullet F_2) = & \mathcal{T}^{\pi}_{\textsc{Ilst}}(v(F_1), T_2) - \mathcal{TF}^{\pi}_{\textsc{Ilst}}(T_2, F_1) + \mathcal{TF}^{\pi}_{\textsc{Ilst}}(v(F_1), F_2) \\
& - \mathcal{K}^{\pi}_{\textsc{Ilst}}(F_1, F_2) + \mathcal{K}^{\pi}_{\textsc{Ilst}}(F_1, T_1 \bullet F_2), \\
\mathcal{T}^{\pi}_{\textsc{Ilst}}(u(F_1), v(F_2)) = & \sigma(l(u), l(v)) \cdot \left(1 + \sum_{p \in \pi(d(u))} \sum_{q \in \pi(d(v))} \mathcal{K}^{\pi}_{\textsc{Ilst}}(F_1^p, F_2^q)\right) \\
& + \sum_{q \in \pi(d(v))} \mathcal{TF}^{\pi}_{\textsc{Ilst}}(u(F_1), F_2^q) + \sum_{p \in \pi(d(u))} \mathcal{TF}^{\pi}_{\textsc{Ilst}}(v(F_2), F_1^p) \\
& - \sum_{p \in \pi(d(u))} \sum_{q \in \pi(d(v))} \mathcal{K}^{\pi}_{\textsc{Ilst}}(F_1^p, F_2^q), \\
\mathcal{F}^{\pi}_{\textsc{Ilst}}(\emptyset, F) = & \mathcal{F}^{\pi}_{\textsc{Ilst}}(F, \emptyset) = 0, \\
\mathcal{F}^{\pi}_{\textsc{Ilst}}(T_1 \bullet F_1, T_2 \bullet F_2) = & \mathcal{T}^{\pi}_{\textsc{Ilst}}(T_1, T_2) \cdot (1 + \mathcal{F}^{\pi}_{\textsc{Ilst}}(F_1, F_2)) \\
& + \mathcal{F}^{\pi}_{\textsc{Ilst}}(F_1, T_2 \bullet F_2) + \mathcal{F}^{\pi}_{\textsc{Ilst}}(T_1 \bullet F_1, F_2) - \mathcal{F}^{\pi}_{\textsc{Ilst}}(F_1, F_2).
\end{array}
$$

図 5.17: $\pi \in \{o, b, c, cb\}$ と $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対して $\mathcal{K}^{\pi}_{\mathtt{A}}(T_1, T_2)$ を計算する再帰式.

定理 5.25. $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対して, 図 5.17 の再帰式は, $\mathcal{K}^{b}_{\mathtt{A}}(T_1, T_2)$ を $O(nm)$ 時間, $\mathcal{K}^{c}_{\mathtt{A}}(T_1, T_2)$ と $\mathcal{K}^{cb}_{\mathtt{A}}(T_1, T_2)$ を $O(nmdD)$ 時間で正しく計算する．ここで,

$n = |T_1|$, $m = |T_2|$, $d = \min\{d(T_1), d(T_2)\}$, $D = \max\{d(T_1), d(T_2)\}$ である.

[**証明**]. 式 $\mathcal{K}^{\pi}_{\mathrm{Top}}$ と $\mathcal{T}^{\pi}_{\mathrm{LCA}}$ において, $\mathcal{F}^{\pi}_{\mathrm{Top}}(F^p_1, F^q_2)$ と $\mathcal{F}^{\pi}_{\mathrm{LCA}}(F^p_1, F^q_2)$ の数は $\pi = o$ のとき 1, $\pi = b$ のとき 4, $\pi = c$ のとき $d(u) \cdot d(v)$, $\pi = bc$ のとき $2d(u) \cdot 2d(v)$ である. また, 式 $\mathcal{T}^{\pi}_{\mathrm{Acc}}$ と $\mathcal{T}^{\pi}_{\mathrm{Ilst}}$ において, $\mathcal{F}^{\pi}_{\mathrm{Acc}}(F^p_1, F^q_2)$ と $\mathcal{F}^{\pi}_{\mathrm{Ilst}}(F^p_1, F^q_2)$ の数は $\pi = o$ のとき 1, $\pi = b$ のとき $4+2+2+4 = 12$, $\pi = c$ のとき $d(u) \cdot d(v) + d(u) + d(v) + d(u) \cdot d(v) = 2d(u) \cdot d(v) + d(u) + d(v)$, $\pi = bc$ のとき $2d(u) \cdot 2d(v) + 2d(u) + 2d(v) + 2d(u) \cdot 2d(v) = 8d(u) \cdot d(v) + 2d(u) + 2d(v)$ である. よって, $\pi \in \{o, b\}$ のときはこれらの再帰式を $O(1)$ 時間で計算でき, $\pi \in \{c, cb\}$ のときは $O(d(u) \cdot d(v)) = O(dD)$ 時間で計算できる. したがって, 時間計算量については定理が成り立つ. また, 定理 5.24 を拡張することにより正当性も成り立つ. □

次に, 本節では, 図 5.16 の再帰式を, $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対する $\mathcal{K}^u_{\mathtt{A}}(T_1, T_2)$ を計算する再帰式に拡張する.

非負整数 s と t に対して, $B_{s,t}$ を, $X = \{1, \ldots, s\}$ かつ $Y = \{1, \ldots, t\}$ となる完全二部グラフ $(X \cup Y, E)$ とし, $BM(s,t)$ を $B_{s,t}$ のすべての最大マッチングの集合とする. 任意の $M \in BM(s,t)$ に対して, $M \subset E$ かつ $|M| = \min\{s, t\}$ である.

木 $u(F_1)$ と $v(F_2)$ に対して, $F_1 = [T_1[u_1], \ldots, T_1[u_s]]$ かつ $F_2 = [T_2[v_1], \ldots, T_2[v_t]]$ とする. すなわち, $ch(u) = \{u_1, \ldots, u_s\}$, $ch(v) = \{v_1, \ldots, v_t\}$, $d(u) = s$, $d(v) = t$ である. このとき, $M \in BM(s,t)$ に対して, 順序付き森 $\bullet_{(i,j)\in M} T_1[u_i]$ を F^M_1, 順序付き森 $\bullet_{(i,j)\in M} T_2[v_j]$ を F^M_2 と表す. ここで, 森の中の木は M の順序に沿って順序付けられていると仮定する. さらに, 順序付き森 F に対して, $pm(F)$ を F のすべての順列森の集合とする. このとき, 図 5.18 は $\mathcal{K}^u_{\mathtt{A}}(T_1, T_2)$ を計算する再帰式となる.

定理 5.26. $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対して, 図 5.18 の再帰式は, $O(nmD^D)$ 時間で $\mathcal{K}^u_{\mathtt{A}}(T_1, T_2)$ を正しく計算する. ここで, $n = |T_1|$, $m = |T_2|$ $D = \max\{d(T_1), d(T_2)\}$ である. したがって, 無順序木の次数がある定数以下である (次数限定である) ならば, $\mathcal{K}^u_{\mathtt{A}}(T_1, T_2)$ を $O(nm)$ 時間で計算できる.

$$
\begin{aligned}
\mathcal{K}^u_{\mathrm{Top}}(u(F_1), v(F_2)) &= \sigma(l(u), l(v)) \cdot \left(1 + \textstyle\sum_{M \in BM(d(u), d(v))} \mathcal{F}^u_{\mathrm{Top}}(F_1^M, F_2^M)\right), \\
\mathcal{F}^u_{\mathrm{Top}}(\emptyset, F) &= \mathcal{F}^u_{\mathrm{Top}}(F, \emptyset) = 0, \\
\mathcal{F}^u_{\mathrm{Top}}(T_1 \bullet F_1, T_2 \bullet F_2) &= \mathcal{K}^u_{\mathrm{Top}}(T_1, T_2) \cdot (1 + \mathcal{F}^u_{\mathrm{Top}}(F_1, F_2)) \\
&\quad + \mathcal{F}^u_{\mathrm{Top}}(F_1, T_2 \bullet F_2) + \mathcal{F}^u_{\mathrm{Top}}(T_1 \bullet F_1, F_2) - \mathcal{F}^u_{\mathrm{Top}}(F_1, F_2).
\end{aligned}
$$

$$
\mathcal{K}^u_{\mathrm{LcaSg}}(T_1, T_2) = \sum_{u \in T_1} \sum_{v \in T_2} \mathcal{K}^u_{\mathrm{Top}}(T_1[u], T_2[v]).
$$

$$
\begin{aligned}
\mathcal{K}^u_{\mathrm{Lca}}(T_1, T_2) &= \sum_{u \in T_1} \sum_{v \in T_2} \mathcal{T}^u_{\mathrm{Lca}}(T_1[u], T_2[v]), \\
\mathcal{T}^u_{\mathrm{Lca}}(u(F_1), v(F_2)) &= \sigma(l(u), l(v)) \cdot \left(1 + \textstyle\sum_{M \in BM(d(s), d(t))} \mathcal{F}^u_{\mathrm{Lca}}(F_1^M, F_2^M)\right), \\
\mathcal{F}^u_{\mathrm{Lca}}(\emptyset, F) &= \mathcal{F}^u_{\mathrm{Lca}}(F, \emptyset) = 0, \\
\mathcal{F}^u_{\mathrm{Lca}}(T_1 \bullet F_1, T_2 \bullet F_2) &= \mathcal{K}^u_{\mathrm{Lca}}(T_1, T_2) \cdot (1 + \mathcal{F}^u_{\mathrm{Lca}}(F_1, F_2)) \\
&\quad + \mathcal{F}^u_{\mathrm{Lca}}(F_1, T_2 \bullet F_2) + \mathcal{F}^u_{\mathrm{Lca}}(T_1 \bullet F_1, F_2) - \mathcal{F}^u_{\mathrm{Lca}}(F_1, F_2).
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{K}^u_{\mathrm{Acc}}(\emptyset, F) &= \mathcal{K}^u_{\mathrm{Acc}}(F, \emptyset) = \mathcal{TF}^u_{\mathrm{Acc}}(T, \emptyset) = 0, \\
\mathcal{K}^u_{\mathrm{Acc}}(T_1 \bullet F_1, T_2 \bullet F_2) &= \mathcal{T}^u_{\mathrm{Acc}}(T_1, T_2) \cdot (\mathcal{F}^u_{\mathrm{Acc}}(F_1, F_2) - 1) \\
&\quad + \mathcal{TF}^u_{\mathrm{Acc}}(T_1, T_2 \bullet F_2) - \mathcal{TF}^u_{\mathrm{Acc}}(T_1, F_2) \\
&\quad + \mathcal{TF}^u_{\mathrm{Acc}}(T_2, T_1 \bullet F_1) - \mathcal{TF}^u_{\mathrm{Acc}}(T_2, F_1) \\
&\quad + \mathcal{K}^u_{\mathrm{Acc}}(T_1 \bullet F_1, F_2) - \mathcal{K}^u_{\mathrm{Acc}}(F_1, F_2), \\
\mathcal{TF}^u_{\mathrm{Acc}}(v(F_1), T_2 \bullet F_2) &= \mathcal{T}^u_{\mathrm{Acc}}(v(F_1), T_2) - \mathcal{TF}^u_{\mathrm{Acc}}(T_2, F_1) + \mathcal{TF}^u_{\mathrm{Acc}}(v(F_1), F_2) \\
&\quad - \mathcal{K}^u_{\mathrm{Acc}}(F_1, F_2) + \mathcal{K}^u_{\mathrm{Acc}}(F_1, T_1 \bullet F_2), \\
\mathcal{T}^u_{\mathrm{Acc}}(u(F_1), v(F_2)) &= \sigma(l(u), l(v)) \cdot \left(1 + \textstyle\sum_{M \in BM(d(s), d(t))} \mathcal{F}^u_{\mathrm{Acc}}(F_1', F_2')\right) \\
&\quad + \textstyle\sum_{F_2' \in pm(F_2)} \mathcal{TF}^u_{\mathrm{Acc}}(u(F_1), F_2') + \sum_{F_1' \in pm(F_1)} \mathcal{TF}^u_{\mathrm{Acc}}(v(F_2), F_1') \\
&\quad - \textstyle\sum_{M \in BM(d(s), d(t))} \mathcal{K}^u_{\mathrm{Acc}}(F_1^M, F_2^M), \\
\mathcal{F}^u_{\mathrm{Acc}}(\emptyset, F) &= \mathcal{F}^u_{\mathrm{Acc}}(F, \emptyset) = 0, \\
\mathcal{F}^u_{\mathrm{Acc}}(T_1 \bullet F_1, T_2 \bullet F_2) &= \mathcal{T}^u_{\mathrm{Acc}}(T_1, T_2) \cdot (1 + \mathcal{F}^u_{\mathrm{Acc}}(F_1, F_2)) \\
&\quad + \mathcal{F}^u_{\mathrm{Acc}}(F_1, T_2 \bullet F_2) + \mathcal{F}^u_{\mathrm{Acc}}(T_1 \bullet F_1, F_2) - \mathcal{F}^u_{\mathrm{Acc}}(F_1, F_2).
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{K}^u_{\mathrm{Ilst}}(\emptyset, F) &= \mathcal{K}^u_{\mathrm{Ilst}}(F, \emptyset) = \mathcal{TF}^u_{\mathrm{Ilst}}(T, \emptyset) = 0, \\
\mathcal{K}^u_{\mathrm{Ilst}}(T_1 \bullet F_1, T_2 \bullet F_2) &= \mathcal{T}^u_{\mathrm{Ilst}}(T_1, T_2) \cdot (\mathcal{F}^u_{\mathrm{Ilst}}(F_1, F_2) - 1) \\
&\quad + \mathcal{TF}^u_{\mathrm{Ilst}}(T_1, T_2 \bullet F_2) - \mathcal{TF}^u_{\mathrm{Ilst}}(T_1, F_2) \\
&\quad + \mathcal{TF}^u_{\mathrm{Ilst}}(T_2, T_1 \bullet F_1) - \mathcal{TF}^u_{\mathrm{Ilst}}(T_2, F_1) \\
&\quad + \mathcal{K}^u_{\mathrm{Ilst}}(T_1 \bullet F_1, F_2) - \mathcal{K}^u_{\mathrm{Ilst}}(F_1, F_2), \\
\mathcal{TF}^u_{\mathrm{Ilst}}(v(F_1), T_2 \bullet F_2) &= \mathcal{T}^u_{\mathrm{Ilst}}(v(F_1), T_2) - \mathcal{TF}^u_{\mathrm{Ilst}}(T_2, F_1) + \mathcal{TF}^u_{\mathrm{Ilst}}(v(F_1), F_2) \\
&\quad - \mathcal{K}^u_{\mathrm{Ilst}}(F_1, F_2) + \mathcal{K}^u_{\mathrm{Ilst}}(F_1, T_1 \bullet F_2), \\
\mathcal{T}^u_{\mathrm{Ilst}}(u(F_1), v(F_2)) &= \sigma(l(u), l(v)) \cdot \left(1 + \textstyle\sum_{M \in BM(d(s), d(t))} \mathcal{K}^u_{\mathrm{Ilst}}(F_1^M, F_2^M)\right) \\
&\quad + \textstyle\sum_{F_2' \in pm(F_2)} \mathcal{TF}^u_{\mathrm{Ilst}}(u(F_1), F_2') + \sum_{F_1' \in pm(F_1)} \mathcal{TF}^u_{\mathrm{lst}}(v(F_2), F_1') \\
&\quad - \textstyle\sum_{M \in BM(d(s), d(t))} \mathcal{K}^u_{\mathrm{Ilst}}(F_1^M, F_2^M), \\
\mathcal{F}^u_{\mathrm{Ilst}}(\emptyset, F) &= \mathcal{F}^u_{\mathrm{Ilst}}(F, \emptyset) = 0, \\
\mathcal{F}^u_{\mathrm{Ilst}}(T_1 \bullet F_1, T_2 \bullet F_2) &= \mathcal{T}^u_{\mathrm{Ilst}}(T_1, T_2) \cdot (1 + \mathcal{F}^u_{\mathrm{Ilst}}(F_1, F_2)) \\
&\quad + \mathcal{F}^u_{\mathrm{Ilst}}(F_1, T_2 \bullet F_2) + \mathcal{F}^u_{\mathrm{Ilst}}(T_1 \bullet F_1, F_2) - \mathcal{F}^u_{\mathrm{Ilst}}(F_1, F_2).
\end{aligned}
$$

図 5.18: $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対して $\mathcal{K}^u_{\mathtt{A}}(T_1, T_2)$ を計算する再帰式.

[**証明**]. $|BM(s,t)| = {}_sP_t$ であり, F_1 のすべての順列森の数は ${}_sP_1$ かつ F_2 のすべての順列森の数は ${}_tP_1$ なので, 式 $\mathcal{F}^u_{\mathtt{A}}(F_1^M, F_2^M)$ の出現数は D^D 以下であり, $\mathtt{A} \in \{\textsc{Acc}, \textsc{Ilst}\}$ に対する $\mathcal{F}^u_{\mathtt{A}}(u(F_1), F_2')$ と $\mathcal{F}^u_{\mathtt{A}}(F_1', v(F_2))$ の出現数は D^D 以下である. どちらの場合も, 式の出現数は $O(D^D)$ である. 任意の組 $(u, v) \in T_1 \times T_2$ はただ一度だけ呼び出されるので, 動的プログラミングを用いることで, $K^u_{\mathtt{A}}(T_1, T_2)$ を $O(nmD^D)$ 時間で計算できる. したがって, 時間計算量については定理が成り立つ. また, 定理 5.24 を拡張することにより正当性も成り立つ. □

最後に, 無順序木のマッピングカーネル計算の困難性について議論する. まず, 文献 [14, 25] における#P 完全性は, 無順序木の一般のトップダウンマッピングカーネルに対して適用することができない. そこで, 以下では, 特定のトップダウンマッピング (もしくはボトムアップマッピング) を数え上げる問題が#P 完全であることを証明する.

M を T_1 と T_2 のマッピングとする. 任意の $(u,v) \in M$ に対して $l(u) = l(v)$ となるとき, M を**ラベル保存マッピング** (*label-preserving mapping*) (または**インデルマッピング** (*indel mapping*)) という. また, 任意の $(u,v) \in M$ に対して, $u \in anc(u')$, $v \in anc(v')$, $u' \in lv(T_1)$, $v' \in lv(T_2)$ となる $(u',v') \in M$ が存在するとき, M を**葉拡張** (*leaf-extended*) という. 以下では, 無順序木 T_1 と T_2 のラベル保存葉拡張トップダウンマッピングの集合を $\mathcal{M}^u_{\text{LLTOP}}(T_1,T_2)$ と表す.

定理 5.27 ([14] 参照)**.** $\mathcal{M}^u_{\text{LLTOP}}(T_1,T_2)$ のすべてのマッピングを数え上げる問題は#P 完全である.

[**証明**]. Valiant [46] は, 二部グラフにおけるすべてのマッチングを数え上げる問題は#P 完全であることを示した. そこで, すべてのラベル保存葉拡張トップダウンマッピングの数と#Bipartite Matching の出力が同一になるような 2 つの木を構成する. ここで, 森 F と $l(v) = a$ となるノード v に対して, $v(F)$ を $a(F)$ と表す.

$G = (X \cup Y, E)$ を二部グラフとする. $v \in X \cup Y$ に対して, v の近傍を $N(v)$ と表す. このとき, $v \in X$ ならば $N(v) \subseteq Y$ であり, $v \in Y$ ならば $N(v) \subseteq X$ である. このとき, 任意の $x \in X$ に対して, 木 $T_x = a(\{xy \mid y \in N(x)\})$ を構成し, 木 $T_1 = a(\{T_x \mid x \in X\})$ を構成する. 同様に, $y \in Y$ に対して, 木 $T_y = a(\{xy \mid x \in N(y)\})$ を構成し, 木 $T_2 = a(\{T_y \mid y \in Y\})$ を構成する. ここで, G の辺 xy を T_x と T_y の葉のラベルとみなしている. 図 5.19 は, 二部グラフ G, および, G から構成される木 T_1 と T_2 の例である.

G のマッチング $B \subseteq E$ に対して, T_1 と T_2 のラベル保存葉拡張トップダウンマッピング M を以下のように構成する.

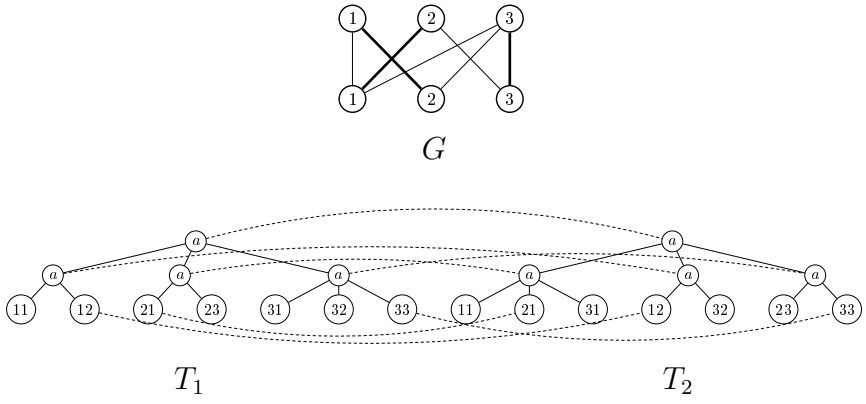


図 5.19: 二部グラフ G と木 T_1 と T_2.

$$M = \begin{cases} \emptyset & \text{if } B = \emptyset, \\ \{(r(T_1), r(T_2))\} \cup \bigcup_{xy \in B} M_{xy} & \text{if } B \neq \emptyset, \end{cases}$$

$$M_{xy} = \left\{ (u_1, v_1), (u_2, v_2) \in V(T_x) \times V(T_y) \;\middle|\; \begin{array}{l} u_1 = par(u_2), v_1 = par(v_2), \\ u_2 \in lv(T_x), v_2 \in lv(T_y) \\ l(u_1) = l(v_1) = a, l(u_2) = l(v_2) = xy \end{array} \right\}.$$

例えば, B を, 図 5.19 の太線で表される G のマッチング $\{12, 21, 33\}$ とする. このとき, T_1 と T_2 のラベル保存葉拡張トップダウンマッピング M は, 図 5.19 の破線で表されているマッピングとなる.

T_x と T_y の定義より, M_{xy} は T_x と T_y のラベル保存葉拡張トップダウンマッピングとなる. また, M_{xy} は G のマッチングの要素 xy と対応する. さらに, 根から T_x もしくは T_y の 2 つ以上の葉へのパスを含むような T_1 と T_2 のラベル保存葉拡張トップダウンマッピング M_{xy} は存在しない. すなわち, M_{xy} は T_x と T_y の高々1 つのパスしか含まない.

したがって, G のマッチング B は T_1 と T_2 のラベル保存葉拡張トップダウンマッピング M を一意に決定し, また逆に, M は B を一意に決定する. よって, G のすべてのマッチングの数 (#BipartiteMatching の出力) は T_1 と T_2 のすべてのラベル保存葉拡張トップダウンマッピングと一致する. □

無順序木 T_1 と T_2 のすべてのラベル保存ボトムアップマッピングの集合を $\mathcal{M}^u_{\mathrm{LBOT}}(T_1, T_2)$ と表す. このとき, 文献 [14, 25] の証明もしくは上の証明から, 以下の系を得ることができる. この証明は, 例えば, 図 5.19 のマッチング B から, 図 5.20 のマッピング:

$$\bigcup_{xy \in B} \{(u, v) \in lv(T_x) \times lv(T_y) \mid l(u) = l(v) = xy\}$$

を構成できることで得られる.

系 5.28 ([14, 25] 参照)**.** $\mathcal{M}^u_{\mathrm{LBOT}}(T_1, T_2)$ のすべてのマッピングを数え上げる問題は#P 完全である.

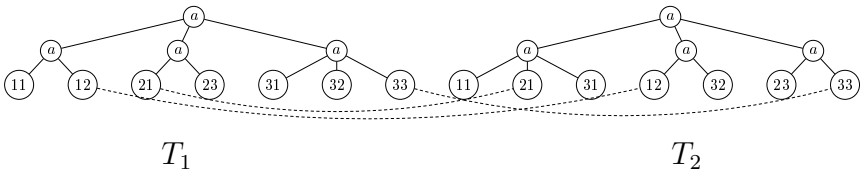


図 5.20: 系 5.28 の木 T_1 と T_2.

第6章　結論と今後の課題

本章では, 本論文における, 共通部分森に基づく Tai マッピングの階層, 木アライメント距離の計算, Tai マッピングおよび編集距離のさまざまな拡張について結論と今後の課題についてまとめる.

6.1　共通部分森に基づく Tai マッピングの階層

3 章では, Tai マッピング階層を, 共通部分森内のノードの接続に着目した埋め込み部分森, 誘導部分森, 完全部分森, および, 共通部分森内の部分木の並びに着目したねじれ無し部分森, 並列部分森, 部分木, 根保存部分木, という観点から特徴付けた. そして, この観点から見て不完全であった箇所に新たなマッピングを導入した. 次に, 上記のマッピングの最小コストとして定式化される編集距離の変種について, メトリック性とこれらの計算の時間計算量を解析した.

その結果を図 6.1 および図 6.2 にまとめる. 図 6.1 は, 順序木における Tai マッピング階層の各マッピングに対する距離を計算する時間計算量である. ここで, 各ノードに対応する距離計算の時間計算量は, 黒の実線で囲まれたノードが $O(nm^2(1+\log\frac{n}{m}))$ 時間, 黒の破線で囲まれたノードが $O(nmD^2)$ 時間, 黒の点線で囲まれたノードが $O(nm)$ 時間, 灰色の実線で囲まれたノードが $O(n+m)$ 時間である. また, 図 6.2 は, 無順序木における Tai マッピング階層の各マッピングに対する距離を計算する時間計算量である. ここで, 黒の実線で囲まれたノードに対応する距離の計算問題は T_1, T_2 が二分木であっても MAX SNP 困難,

黒の破線で囲まれたノードに対応する距離の計算問題は T_1, T_2 の次数が定数以下なら多項式時間計算可能, 黒の点線で囲まれたノードに対応する距離計算の時間計算量は $O(nmd)$ 時間, 灰色の実線で囲まれたノードに対応する距離計算の時間計算量は $O(nm)$ 時間, 灰色の破線で囲まれたノードに対応する距離計算の時間計算量は $O(n+m)$ 時間である.

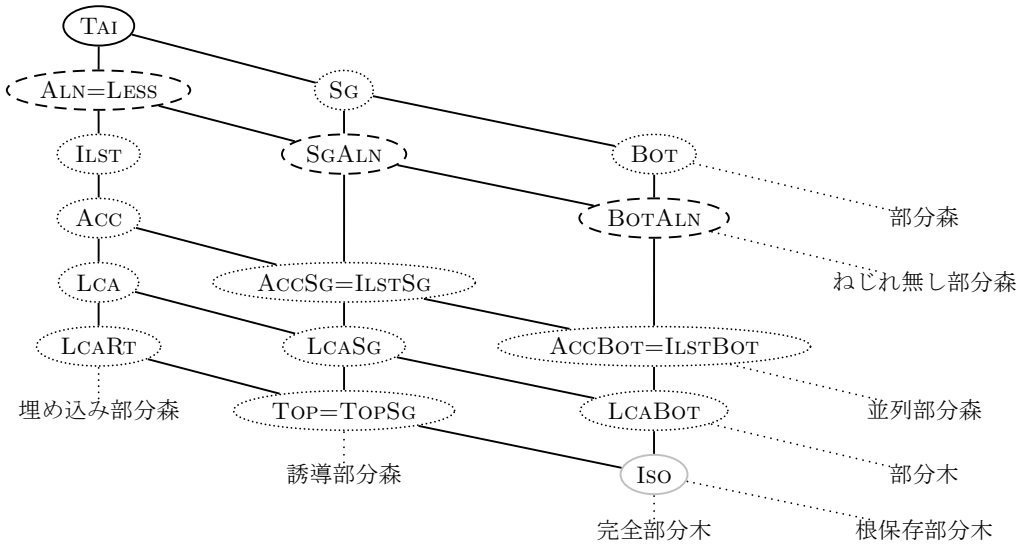


図 6.1: $\mathcal{M}_{\mathtt{A}}(T_1, T_2)$ に対する $\tau^{o}_{\mathtt{A}}(T_1, T_2)$ を計算する時間計算量.

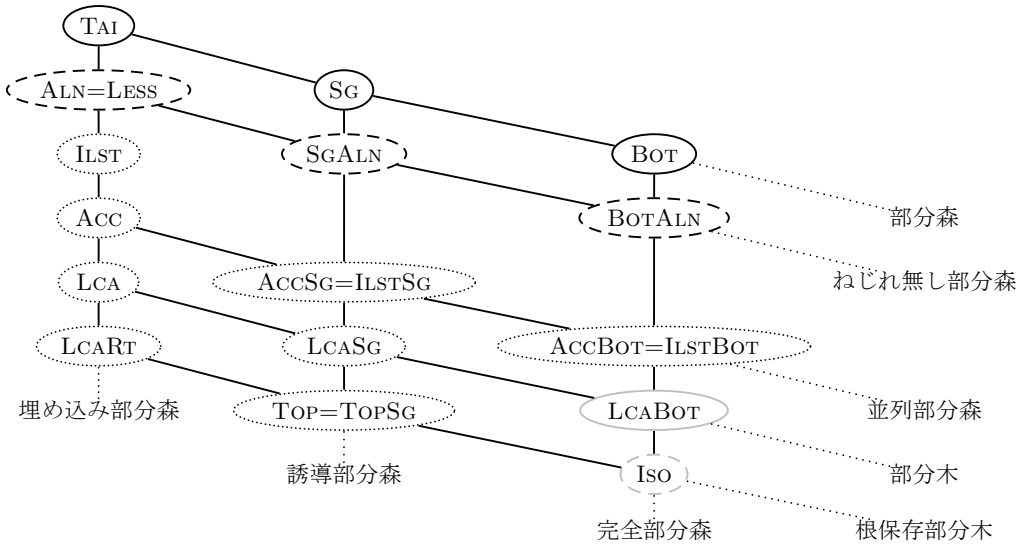


図 6.2: $\mathcal{M}_{\mathtt{A}}(T_1, T_2)$ に対する $\tau^{u}_{\mathtt{A}}(T_1, T_2)$ を計算する時間計算量.

Bringmann ら [5] による, APSP の仮定の下で任意の定数 $\varepsilon > 0$ に対して順序木編集距離は $O(n^{3-\varepsilon})$ 時間で計算できない (ただし, n は木のノード数の最大値), という最適性に

関連して, 以下の結論を簡単に導くことができる.

Backurs と Indyk [3] は, SETH (Strongly Exponential Time Hypothesis) の仮定の下で, 任意の定数 $\varepsilon > 0$ に対して文字列編集距離は $O(n^{2-\varepsilon})$ では計算できないことを示した. この結果を利用することで, SETH の仮定の下で, 任意の定数 $\varepsilon > 0$ に対して順序木のトップダウン距離とボトムアップ距離は $O(n^{2-\varepsilon})$ では計算できないことを示すことができる. なぜならば, 同一ラベルのノードに対して, 文字列におけるそれぞれの文字を子として持つ高さ 1 の木を構成すれば, 2 つの文字列から 2 つの木が構成でき, これらの文字列間の編集距離と木間のトップダウン距離は一致し, “文字列編集距離 +2” とボトムアップ距離は一致するからである. したがって, 順序木編集距離の変種の計算における $O(n^2)$ 時間は最適な計算時間であると言える.

Tai マッピング階層の今後の課題として, 以下を挙げることができる.

現時点で, 順序木の誘導部分森と完全部分森に関する編集距離の変種の計算はまだ未実装である. これは, 断片距離計算アルゴリズムが他の編集距離の変種のアルゴリズムと異なるため, 実装における対応ができていないからである. したがって, まずはこの実装が今後の課題である.

Tai マッピング階層において, 図 6.1 から分かるように, 順序木においてねじれ無し部分森に関する距離計算における時間計算量 $O(nmD^2)$ が他の時間計算量 $O(nm)$ と比較しても大きい. 6.2 章でも改めて述べるが, この問題を解決すること, すなわち, ねじれ無し共通部分森に関する距離計算に $O(nmD^2)$ 時間よりも効率がよいアルゴリズムが存在するか否かを解析することは, 今後の大きな課題である.

また, 図 6.2 から分かるように, 無順序木において並列部分森であるか否かが, 多項式時間計算可能であるか否かの境界となっている. これは, 並列部分森であれば, 最大重み二部マッチングは編集距離の計算に適用できるからである [51, 64]. Akutsu ら [2] は, 最大重み二部マッチングの適用可能性を拡げて, 無順序木編集距離の厳密アルゴリズムを設計

している. この手法を利用して, ねじれ無し部分森, すなわち, 無順序木アライメント距離の厳密アルゴリズムを設計することは今後の課題である.

6.2　木アライメント距離の計算

4 章では, 木編集距離の変種の中でも, 特に計算時間の点で特殊な立ち位置にある, 木アライメント距離に着目し, 2 つの観点で議論を進めた.

1 つ目の観点として, アンカーアライメント問題を定式化し, アンカーアライメント距離の導入を行った. まず, 任意の Tai マッピングをアンカーとして用いるアンカーアライメント [37] における問題, すなわち, アンカーが必ずしも劣制限でないためアライメント距離が計算できない場合があるという問題を解決するために, 本論文では, Less=Aln [29] の別証明を与えることで, アンカー M が劣制限でないならば “no” を返し, 劣制限ならばアンカーアライメントを $O(H|M|^2 + n)$ 時間で計算するアルゴリズムを設計した. ここで, H は 2 つの木の最大の高さである.

次に, Tai マッピングの階層の中で最もアライメント可能マッピングに近い孤立部分木マッピングを用いて, 孤立部分木マッピングによって対応していない葉の組 (葉マッピング) を, そのマッピングに可能な限り追加してアンカーを作成した上で, これらのスコアの最小値をアンカーアライメント距離と定式化, そして, このアンカーアライメント距離に対して, N 型糖鎖データおよびランダム生成木を用いて計算実験を行った. N 型糖鎖データの実験では, 多くの木が, アライメント距離とアンカーアライメント距離, 孤立部分木距離が一致した. このときのすべての距離が異なった木の組の例から, 孤立部分木マッピングによって対応していなかった葉と中間ノードを対応付けることにより最適なアライメント可能マッピングが得られるような場合, アンカーアライメントのマッピングではアライメント距離と同じ値を得ることができないことが分かった. 一方, ランダム木の実験では, ランダム木の最大次数を変化させて各次数ごとに計算した結果, アンカーアライメント距

離の平均値は次数の増加によって変化しなかったが, 計算時間は次数の増加により指数的に増加し, アンカーアライメント距離が孤立部分木距離より真に小さくなる木の組は次数の増加により増加した. このことから, アンカーアライメント距離と孤立部分木距離は木の次数が大きければ同じ値を取りにくくなる傾向があった.

2 つ目の観点として, アライメント距離は無順序木の場合計算が困難であるが, 次数を定数以下に限定すると多項式時間計算可能になる点に着目し, 順序木と無順序木の中間の木, すなわち, 順序木より一般的で, 無順序木より制約された木である巡回的順序木に対するアライメント距離を求めるアルゴリズムを設計した. そして, 両順序木の場合に $O(nmD^2)$ 時間, 巡回順序木と巡回両順序木の場合に $O(nmdD^3)$ 時間でアライメント距離が計算できることを示した. また, N 型糖鎖データを用いて, 両順序木アライメント距離と順序木編集距離を比較し, 両順序木アライメント距離が順序木編集距離の近似になり得そうであるといった結果を得た.

木アライメント距離の計算に関する今後の課題として, 以下を挙げることができる.

まず, 木アライメント距離の計算時間解析が木編集距離の計算時間解析よりも進んでいないため, それを詳細に解析することが今後の課題として挙げられる. 1 章の表 1.1 を現時点での結果として再掲すると表 6.3 のようになる.

表 6.3: 木編集距離 τ_{TAI} と木アライメント距離 τ_{ALN} の計算時間. ここで, n はノード数の最大値, D は次数の最大値である.

距離	順序木		無順序木		
	計算時間	最適性	一般	次数 (D) が定数	深さが定数
τ_{TAI}	$O(n^3)$ [8]	$O(n^{3-\varepsilon})$ で計算できない [5]	MAX SNP [61]	MAX SNP[18]	MAX SNP[2, 18]
τ_{ALN}	$O(n^2D^2)$ [21]	未解決	MAX SNP[21]	$O(n^2D^2D!)$ [21]	未解決

アンカーアライメント問題は, もともと Jiang らの順序木アライメント距離計算のアルゴリズム [21] に代わるアルゴリズムを設計することを目的として定式化したものである. したがって, 順序木アライメント距離計算において $O(n^2D^2)$ 時間よりも効率がよいアル

ゴリズムの設計は今後の重要な課題である. 特に, Bringmann ら [5] の証明に用いられているマッピングは, アライメント可能マッピング (劣制限マッピング) ではない. ゆえに, $O(n^2D^2)$ 時間が最適でない可能性が高い.

また, アンカーアライメント問題は, 順序木, 無順序木に関係なく適用できるという利点がある. 実際, 4 章のアンカーアライメント距離は, アンカーとして葉マッピングを利用した無順序木アライメント距離を計算する手法である. 理論的にはアンカーアライメント距離の計算時間は葉の数の指数時間となったが, 無順序木において適切なアンカーを選択する方法を見つけることができれば, 無順序木アライメント距離を多項式時間で計算することができる. したがって, そのようなアンカーの選択方法を考察することは今後の課題である.

さらに, 無順序木編集距離計算では MAX SNP 困難となる深さが定数の場合に, 無順序木アライメント距離が多項式時間で計算可能かどうかも未解決であるので, このことを解析することも今後の課題である. 実際, XML や HTML に基づくデータは, 深さが小さい場合が多いため, この問題を解決することは, 実データの解析にも有用であると考えられる.

最後に, 巡回的順序木をより抽象的に定式化することも今後の課題である. 特に, 子の入れ替えや移動を許した順序木を定式化し, そのような木における木アライメント距離計算アルゴリズムの設計とその計算時間の解析は今後の重要な課題である. なお, 巡回的順序木と同じく, このように定式化された木の編集距離計算は MAX SNP 困難となる.

6.3　さまざまな拡張

5 章では, 木編集距離および Tai マッピングに対して, 3 つの議論を行っている. 1 つ目は, 計算が困難である無順序木編集距離に対する, Higuchi ら [16] が提案した A*アルゴリズムの繰り返し計算を排除した動的 A*アルゴリズムの設計, 2 つ目は, 根付き木に対して定義された Tai マッピングの根無し木への拡張, 3 つ目は, 巡回的順序木と無順序木に対す

る多項式時間計算可能な木カーネルの設計である.

まず, 無順序木編集距離を計算する A* アルゴリズム Astar を改良した動的 A* アルゴリズム DPAstar を設計した. そして, Astar, DPAstar, Shasha らの網羅的アルゴリズム [40] を実装し, また, これらに加えて, Fukagawa らのクリークアルゴリズム [12] と計算時間を比較した. その結果, 糖鎖データに対して, DPAstar が網羅的アルゴリズムよりも非常に高速であり, かつ, Astar やクリークアルゴリズムと比較しても高速であることが検証できた.

また, Tai マッピングを根無し木に拡張し, 3 点保存条件と 4 点保存条件に関するマッピングを加えた. これにより得た根無しマッピングの階層を図 6.4 に示す.

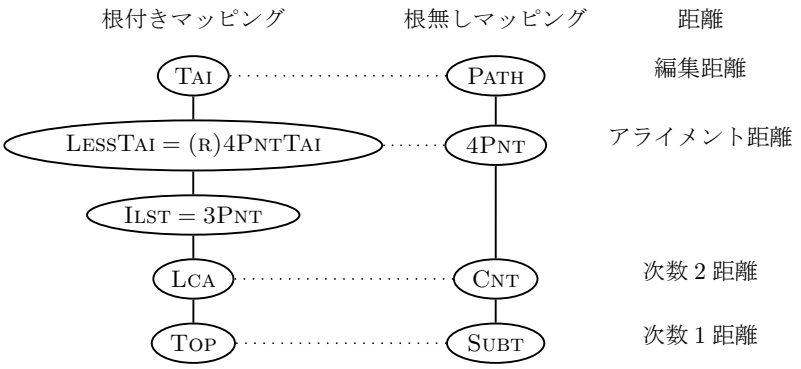


図 6.4: 根付きマッピングの階層と根無しマッピングの階層の比較.

操作的には, $\mathcal{M}_{\mathrm{Top}}$ [38] と $\mathcal{M}_{\mathrm{Subt}}$ は, 挿入と削除を次数 1 のノード (根付き木の場合は葉, 根無し木の場合は端点) に制限したものと対応する. また, $\mathcal{M}_{\mathrm{Lca}}$ [64] と $\mathcal{M}_{\mathrm{Cnt}}$ [64] は, 挿入と削除を次数 2 以下のノードに制限したものと対応する.

さらに, マッピング $\mathtt{A} \in \{\mathrm{Top}, \mathrm{LcaSg}, \mathrm{Lca}, \mathrm{Acc}, \mathrm{Ilst}\}$ に対して, マッピングカーネル $\mathcal{K}^{\pi}_{\mathtt{A}}(T_1, T_2)$ を計算するアルゴリズムを設計した. このマッピングカーネルは, 両順序木 ($\pi = b$) のときは $O(nm)$ 時間, 巡回順序木 ($\pi = c$) または巡回両順序木 ($\pi = cb$) のときは $O(nmdD)$ 時間, 無順序木 ($\pi = u$) のときは $O(nmD^D)$ 時間で計算可能である. ここで, $n = |T_1|$, $m = |T_2|$, $d = \min\{d(T_1), d(T_2)\}$, $D = \max\{d(T_1), d(T_2)\}$ である. したがって,

巡回的順序木のときは常に多項式時間計算可能であり, 無順序木 ($\pi = u$) のときは次数限定であれば多項式時間計算可能である. この結果をまとめると表 6.5 となる.

表 6.5: $\mathtt{A} \in \{\textsc{Top}, \textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ に対する $\mathcal{K}^{\pi}_{\mathtt{A}}(T_1, T_2)$ の計算時間.

π	o	b	c	cb	u
$\mathcal{K}^{\pi}_{\mathtt{A}}(T_1, T_2)$	$O(nm)$	$O(nm)$	$O(nmdD)$	$O(nmdD)$	$O(nmD^D)$

これらの研究における今後の課題は以下のとおりである.

動的 A* アルゴリズムについては, 5 章の表 5.6 から下限の定数が小さい下限関数の方が高速化に効果があった. したがって, 他の編集距離の下限関数を導入し, その効果を考察することは今後の課題である. また, Mori ら [35] は Fukagawa ら [12] のクリークアルゴリズムを改良した無順序木編集距離計算アルゴリズムを設計した. この Mori らのアルゴリズムと DPAstar を比較することも今後の課題である.

根無し木のマッピングについては, まず, マッピングの最小コストである距離を計算するアルゴリズムの設計が挙げられる. ただし, これらの距離は無順序木となるので, MAX SNP 困難となる可能性が高い. そこで, オイラーツアーなどを用いて, 根無し木のノードの近傍に順序を与えて定式化される根無し順序木についてのマッピングとその編集距離について解析することも今後の課題である. また, それらの距離と, 進化系統樹で知られる Robinson-Foulds 距離, Nearest Neighbor Interchange 距離, Subtree Transfer 距離, Quartet 距離 [44] を比較することも今後の課題である.

マッピングカーネルについては, 4 章の結果から, アライメント可能マッピング $\mathcal{M}_{\textsc{Aln}}(T_1, T_2)$ に対して, $\mathcal{K}^{b}_{\textsc{Aln}}(T_1, T_2)$ を $O(nmD^2)$ 時間, $\mathcal{K}^{\pi}_{\textsc{Aln}}(T_1, T_2)$ を $O(nmdD^3)$ 時間 ($\pi \in \{c, cb\}$), $\mathcal{K}^{u}_{\textsc{Aln}}(T_1, T_2)$ を次数限定無順序木の場合に多項式時間で計算できると推測できる. この推測が正しいか否かを解析することは今後の課題である. また, これらのカーネルを実装し, 実データの分類に適用することも, 今後の重要な課題となる.

無順序木に対して, ラベル保存葉拡張トップダウンマッピング, および, ラベル保存ボト

ムアップマッピングを数え上げる問題が#P 完全であることを示した. ここで, 定理 5.27 と系 5.28 において, ラベル保存と葉拡張という条件は本質的である. もしこれらの条件がない場合, 他のトップダウンもしくはボトムアップマッピングを数え上げる必要がある. したがって, 次数を限定しない場合に $\mathcal{K}^u_{\mathtt{A}}(T_1, T_2)$ を計算する問題が#P 完全か否かを解明することは今後の重要な課題である. なお, $\mathcal{K}^u_{\mathrm{Top}}(T_1, T_2)$, $\mathcal{K}^u_{\mathrm{Bot}}(T_1, T_2)$, さらには $\mathtt{A} \in \{\textsc{LcaSg}, \textsc{Lca}, \textsc{Acc}, \textsc{Ilst}\}$ における $\mathcal{K}^u_{\mathtt{A}}(T_1, T_2)$ の計算が#P 完全であることを証明するためには, #BipartiteMatching 問題からの Cook 還元 [25, 46] が必要となることが予想できるが, これは定理 5.27 の証明と比較すると非常に難しくなる可能性が高い.

謝辞

本研究を終えて, 平田 耕一 教授から, 6 年間もの間の篤いご指導を賜りました. ここに深謝申し上げます. また, 本論文を執筆するにあたって, 時間を割いていただき, ご意見, アドバイスを頂きました篠原 武 教授, 坂本 比呂志 教授, 久代 紀之 教授, 宮野 英二 教授, 久保山 哲二 教授には, 御礼の言葉もございません. さらに, 平田研究室の先輩・同期・後輩の皆様の支えなくしては今日の自分は無かったと思います. 共に研究に打ち込んだ平田研究室の皆様に感謝いたします. 最後に, 研究に際して, さまざまなご意見, ご指摘を頂きましたことに感謝の意を述べさせていただきまして, 謝辞にかえさせていただきます.

付 録A　記号一覧

記号	意味	頁
$V(T)$	木 T のノード集合	18
$E(T)$	木 T の辺集合	18
$\|T\|$	木 T の大きさ	18
$\emptyset$	空の木	18
$[u, v]$	ノード u から v へのパス	18
$[\![u, v]\!]$	ノード u から v へのパスから u, v を除いたもの	18
Σ	アルファベット (有限集合)	18
$l(v)$	ノード v のラベル	18
ε	空文字 (アルファベット Σ に含まれない特殊な記号)	18
Σ_ε	$\Sigma \cup \{\varepsilon\}$	18
$r(T)$	木 T の根ノード	19
T^r	ノード r を根として選んだ根付き木 T	19
$T_1 \equiv T_2$	木 T_1 と T_2 が (根付き木として) 同型	19
$u < v$	ノード u はノード v の子孫 (v は u の先祖)	19
$u \leq v$	$u < v$ または $u = v$	19
$u \# v$	$u \leq v$ でも $v \leq u$ でもない	19
$depth(v)$	ノード v の深さ	19
$par(v)$	ノード v の親ノード	19
$ch(v)$	ノード v の子ノードの集合	19

記号	意味	頁
$d(v)$	ノード v の次数	19
$d(T)$	木 T の次数	19
$lv(T)$	木 T の葉ノードの集合	19
$T[v]$	木 T の v を根とした部分木	19
$u \sqcup v$	ノード u と v の最近共通先祖	19
$pre_T(v)$	木 T のノード v の先行走査順による順番	20
$post_T(v)$	木 T のノード v の後行走査順による順番	20
$v_1 \preceq_v v_2$	ノード v の子ノード v_1, v_2 に対して, v_1 が v_2 の左にある	20
$u \preceq v$	ノード u がノード v の左にある	20
$T_1 \equiv^o T_2$	木 T_1 と T_2 が (順序木として) 同型	20
$T_1 \equiv^u T_2$	木 T_1 と T_2 が (無順序木として) 同型	20
Π_v	ノード v の許容される置換	21
Π_T	木 T の許容される置換	21
$\pi(T)$	順列木 T の兄弟関係を $\pi \in \Pi_T$ で並び替えることで得られる順序木	22
$\Pi(T)$	順列木 T の許容される順序木の集合	22
$T_1 \equiv^b T_2$	木 T_1 と T_2 が (両順序木として) 同型	22
$T_1 \equiv^c T_2$	木 T_1 と T_2 が (巡回順序木として) 同型	22
$T_1 \equiv^{cb} T_2$	木 T_1 と T_2 が (巡回両順序木として) 同型	22
$T(v)$	木 T の v を根とする完全部分木から根を取り除くことで得られる完全部分森	22
$[T_1, ..., T_n]$	木 $T_1, \ldots, T_n$ を集めて得られる森	22
$v(F)$	森 F に根ノード v を付けた木	22
$T \bullet F$	森 F の左に木 T を並べることで得られる森	22

記号	意味	頁
$F(v_i, v_j)$	森 $T(v)$ の左から i 番目から j 番目の木を集めて得られる森	22
$(l_1 \mapsto l_2)$	編集操作	23
$\tau_{\mathrm{TAI}}(T_1, T_2)$	木 T_1 と T_2 間の編集距離	24
$\mathcal{M}_{\mathrm{TAI}}(T_1, T_2)$	木 T_1, T_2 間の Tai マッピングすべてからなる集合	24
$M\|_1$	マッピング M に対する $\{u \mid (u, v) \in M\}$	24
$M\|_2$	マッピング M に対する $\{v \mid (u, v) \in M\}$	24
$\mathcal{M}_{\mathrm{LESS}}(T_1, T_2)$	木 T_1, T_2 間の劣制限マッピングすべてからなる集合 (5.2 章とその他では条件が異なる)	25, 90
$\mathcal{M}_{\mathrm{ILST}}(T_1, T_2)$	木 T_1, T_2 間の孤立部分木マッピングすべてからなる集合	25
$\mathcal{M}_{\mathrm{ACC}}(T_1, T_2)$	木 T_1, T_2 間の調和的マッピングすべてからなる集合	26
$\mathcal{M}_{\mathrm{LCA}}(T_1, T_2)$	木 T_1, T_2 間の LCA 保存マッピングすべてからなる集合	26
$\mathcal{M}_{\mathrm{TOP}}(T_1, T_2)$	木 T_1, T_2 間のトップダウンマッピングすべてからなる集合	26
$\mathcal{M}_{\mathrm{BOT}}(T_1, T_2)$	木 T_1, T_2 間のボトムアップマッピングすべてからなる集合	26
$\mathcal{M}_{\mathrm{SG}}(T_1, T_2)$	木 T_1, T_2 間の断片マッピングすべてからなる集合	27
$\mathcal{M}_{\mathrm{TOPSG}}(T_1, T_2)$	木 T_1, T_2 間のトップダウン断片マッピングすべてからなる集合	27
$\tau_{\mathrm{LESS}}(T_1, T_2)$	木 T_1 と T_2 間の劣制限距離	27
$\tau_{\mathrm{ILST}}(T_1, T_2)$	木 T_1 と T_2 間の孤立部分木距離	27
$\tau_{\mathrm{ACC}}(T_1, T_2)$	木 T_1 と T_2 間の調和的距離	27
$\tau_{\mathrm{LCA}}(T_1, T_2)$	木 T_1 と T_2 間の LCA 保存距離	27
$\tau_{\mathrm{TOP}}(T_1, T_2)$	木 T_1 と T_2 間のトップダウン距離	27
$\tau_{\mathrm{BOT}}(T_1, T_2)$	木 T_1 と T_2 間のボトムアップ距離	27
$\tau_{\mathrm{SG}}(T_1, T_2)$	木 T_1 と T_2 間の断片距離	27
$\tau_{\mathrm{TOPSG}}(T_1, T_2)$	木 T_1 と T_2 間のトップダウン断片距離	27

記号	意味	頁
$\mathcal{M}^{*}_{\mathtt{A}}(T_1, T_2, \gamma)$	コスト関数 γ による木 T_1 と T_2 間の最適な A マッピングすべてからなる集合	28
$\mathcal{M}^{o}_{\mathtt{A}}(T_1, T_2)$	木 T_1, T_2 間の順序木 A マッピングすべてからなる集合	28
$\mathcal{M}^{b}_{\mathtt{A}}(T_1, T_2)$	木 T_1, T_2 間の両順序木 A マッピングすべてからなる集合	28
$\mathcal{M}^{c}_{\mathtt{A}}(T_1, T_2)$	木 T_1, T_2 間の巡回順序木 A マッピングすべてからなる集合	28
$\mathcal{M}^{cb}_{\mathtt{A}}(T_1, T_2)$	木 T_1, T_2 間の巡回両順序木 A マッピングすべてからなる集合	28
$\mathcal{M}^{u}_{\mathtt{A}}(T_1, T_2)$	木 T_1, T_2 間の無順序木 A マッピングすべてからなる集合	28
$\mathcal{M}_{\mathrm{ALN}}(T_1, T_2)$	木 T_1, T_2 間のアライメント可能マッピングすべてからなる集合	30
$\tau_{\mathrm{ALN}}(T_1, T_2)$	木 T_1 と T_2 間のアライメント距離	30
$\mathcal{M}_{\mathrm{ILSTSG}}(T_1, T_2)$	木 T_1 と T_2 間の孤立部分木断片マッピングすべてからなる集合	31
$\mathcal{M}_{\mathrm{ACCSG}}(T_1, T_2)$	木 T_1 と T_2 間の調和的断片マッピングすべてからなる集合	31
$\mathcal{M}_{\mathrm{LCASG}}(T_1, T_2)$	木 T_1 と T_2 間の LCA 保存断片マッピングすべてからなる集合	31
$\mathcal{M}_{\mathrm{SGALN}}(T_1, T_2)$	木 T_1 と T_2 間の断片アライメント可能マッピングすべてからなる集合	31
$\mathcal{M}_{\mathrm{BOTALN}}(T_1, T_2)$	木 T_1 と T_2 間のボトムアップアライメント可能マッピングすべてからなる集合	31
$\mathcal{M}_{\mathrm{ILSTBOT}}(T_1, T_2)$	木 T_1 と T_2 間の孤立部分木ボトムアップマッピングすべてからなる集合	31
$\mathcal{M}_{\mathrm{ACCBOT}}(T_1, T_2)$	木 T_1 と T_2 間の調和的ボトムアップマッピングすべてからなる集合	31
$\mathcal{M}_{\mathrm{LCABOT}}(T_1, T_2)$	木 T_1 と T_2 間の LCA 保存ボトムアップマッピングすべてからなる集合	31
$\mathcal{M}_{\mathrm{LCART}}(T_1, T_2)$	木 T_1 と T_2 間の LCA 保存根保存マッピングすべてからなる集合	31
$\tau_{\mathrm{ILSTSG}}(T_1, T_2)$	木 T_1 と T_2 間の孤立部分木断片距離	38
$\tau_{\mathrm{ACCSG}}(T_1, T_2)$	木 T_1 と T_2 間の調和的断片距離	38

記号	意味	頁
$\tau_{\mathrm{LCASG}}(T_1, T_2)$	木 T_1 と T_2 間の LCA 保存断片距離	38
$\tau_{\mathrm{SGALN}}(T_1, T_2)$	木 T_1 と T_2 間の断片アライメント距離	38
$\tau_{\mathrm{BOTALN}}(T_1, T_2)$	木 T_1 と T_2 間のボトムアップアライメント距離	38
$\tau_{\mathrm{ILSTBOT}}(T_1, T_2)$	木 T_1 と T_2 間の孤立部分木ボトムアップ距離	38
$\tau_{\mathrm{ACCBOT}}(T_1, T_2)$	木 T_1 と T_2 間の調和的ボトムアップ距離	38
$\tau_{\mathrm{LCABOT}}(T_1, T_2)$	木 T_1 と T_2 間の LCA 保存ボトムアップ距離	38
$\tau_{\mathrm{LCART}}(T_1, T_2)$	木 T_1 と T_2 間の LCA 保存根保存距離	38
$M_1 \circ M_2$	マッピング M_1 と M_2 の合成	40
$\delta^o_{\mathtt{A}}(F_1, F_2)$	森 F_1 と F_2 間の順序木 A 距離	42
$\delta^u_{\mathtt{A}}(F_1, F_2)$	森 F_1 と F_2 間の無順序木 A 距離	42
$T_1 \equiv^o_l T_2$	木 T_1 と T_2 が (順序木として) ラベルなし同型 ($T_1 \equiv^o T_2$ に同じ)	44
$T_1 \equiv^u_l T_2$	木 T_1 と T_2 が (無順序木として) ラベルなし同型 ($T_1 \equiv^u T_2$ に同じ)	44
$C_T(v, U)$	ノード $v \in T$ の $U \subseteq V(T)$ に関する被覆集合	53
$S_T(v, U)$	ノード $v \in T$ の $U \subseteq V(T)$ に関する被覆列	53
$P_{T_1}(u)$	$S_{T_1}(u, M\|_1)$ の u から $r(T_1)$ までのパス	53
$P_{T_2}(v)$	$S_{T_2}(v, M\|_2)$ の v から $r(T_2)$ までのパス	53
$\mathcal{G}_M$	劣制限マッピング M の併合グラフ	60
$\tau^{\gamma}_{\mathrm{ACH}}(T_1, T_2, M)$	木 T_1, T_2 のマッピング M によるアンカーアライメント距離	63
$ach(T_1, T_2, M)$	アンカーアライメント問題で木 T_1, T_2 とマッピング M を入力として得られる木	63
$lm(M)$	木 T_1 と T_2 間のマッピング M に含まれない T_1 と T_2 の葉の組すべてからなる集合	66

記号	意味	頁
$\tau^{\gamma}_{\mathrm{ACH}}(T_1, T_2)$	木 T_1 と T_2 間のアンカーアライメント距離	66
$ET(T_1, T_2)$	木 T_1 と T_2 の編集距離探索木	79
$n(T_1, T_2)$	木 T_1 と T_2 のノード数の差	80
$d_1(T_1, T_2)$	木 T_1 と T_2 の次数ヒストグラム L_1 距離	80
$d_\infty(T_1, T_2)$	木 T_1 と T_2 の次数ヒストグラム L_∞ 距離	80
$l_1(T_1, T_2)$	木 T_1 と T_2 のラベルヒストグラム L_1 距離	80
$l_\infty(T_1, T_2)$	木 T_1 と T_2 のラベルヒストグラム L_∞ 距離	80
$lb(T_1, T_2)$	木 T_1 と T_2 間の無順序木編集距離の下限関数	82
$c(u, v, w)$	ノード u, v, w の中心	88
$\mathcal{M}_{\mathrm{LessTai}}(T_1, T_2)$	木 T_1, T_2 間の劣制限 Tai マッピングすべてからなる集合 (5.2 章以外での劣制限マッピング)	90
$\mathcal{M}_{\mathrm{Path}}(T_1, T_2)$	根無し木 T_1 と T_2 間のパス保存マッピングすべてからなる集合	90
$\mathcal{M}_{\mathrm{Cnt}}(T_1, T_2)$	根無し木 T_1 と T_2 間の中心保存マッピングすべてからなる集合	90
$\mathcal{M}_{\mathrm{Subt}}(T_1, T_2)$	根無し木 T_1 と T_2 間の部分木保存マッピングすべてからなる集合	90
$\mathcal{M}_{\mathrm{4Pnt}}(T_1, T_2)$	根無し木 T_1 と T_2 間の 4 点保存マッピングすべてからなる集合	90
$\mathcal{M}_{\mathrm{R4Pnt}}(T_1, T_2)$	木 T_1 と T_2 間の根付き 4 点保存マッピングすべてからなる集合	91
$\mathcal{M}_{\mathrm{R4PntTai}}(T_1, T_2)$	木 T_1 と T_2 間の根付き 4 点保存 Tai マッピングすべてからなる集合	91
$\mathcal{M}_{\mathrm{3Pnt}}(T_1, T_2)$	木 T_1 と T_2 間の 3 点保存マッピングすべてからなる集合	92
$\mathcal{K}^{\pi}_{\mathtt{A}}(F_1, F_2)$	π で与えられる順序の森 F_1, F_2 間の A マッピングを用いた類似度	99

参考文献

[1] T. Akutsu: *Tree edit distance problems: Algorithms and applications to bioinformatics* IEICE Transactions on Information and Systems, vol. 93-D, no. 2, 208–218, 2010.

[2] T. Akutsu, D. Fukagawa, M. M. Halldórsson, A. Takasu, K. Tanaka: *Approximation and parameterized algorithms for common subtrees and edit distance between unordered trees*, Theoret. Comput. Sci. **470**, 10–22, 2013.

[3] A. Backurs, P. Indyk: *Edit distance cannot be computed in strongly subquadratic time (unless SETH is false)*, Proc. STOC'15, 51–58, 2015.

[4] P. Bille: *A survey on tree edit distance and related problems*, Theoret. Comput. Sci. **337**, 217–239, 2005.

[5] K. Brigmann, P. Gawrychowski, S. Mozes, O. Weiman: *Tree edit distance cannot be computed in strongly subcubic time (unless APSP can)*, Proc. SODA'18 (to appear). Also available to arXiv:1703.08940.

[6] P. Buneman: *A note on the metric properties of trees*, J. Comb. Theory (B) **17**, 48–50, 1974.

[7] S. S. Chawathe: *Comparing hierarchical data in external memory*, Proc. VLDB'99, 90–101, 1999.

[8] E. Demaine, S. Mozes, B. Rossman, O. Weimann: *An optimal decomposition algorithm for tree edit distance*, ACM Trans. Algo. **6**, 2009.

[9] M. M. Deza, E. Deza: *Encyclopedia of distances* (4th edition), Springer, 2016.

[10] P. Ferraro, C. Godin: *A distance measure between plant architectures* , Annals of Forest Science **57**, 445-461, 2000.

[11] P. Ferraro, C. Godin: *Optimal mappings with minimum number of connected components in tree-to-tree comparison problems*, J. Algo. **48**, 385–406, 2003.

[12] D. Fukagawa, T. Tamura, A. Takasu, E. Tomita, T. Akutsu: *A clique-based method for the edit distance between unordered trees and its application to analysis of glycan structures*, BMC Bioinformatics 2011 **12**, 2011.

[13] T. Gärtner: *Kernels for structured data*, World Scientific, 2008.

[14] I. Hamada, T. Shimada, D. Nakata, K. Hirata, T. Kuboyama: *Agreement subtree mapping kernel for phylogenetic trees*, JSAI-isAI Post-Workshop Proc., LNAI **8417**, 1–16, 2014.

[15] A. Hamou-Lhadj, T. C. Lethbridge: *An efficient algorithm for detecting patterns in traces of procedure calls*, Proc. ICSE Workshop on Dynamic Analysis (WODA'03), 33–36, 2003.

[16] S. Higuchi, T. Kan, Y. Yamamoto, K. Hirata: *An A* algorithm for computing edit distance between rooted labeled unordered trees*, New Frontiers in Artificial Intelligence, LNAI **7258**, 186–196, 2012.

[17] K. Hirata, T. Kuboyama, T. Yoshino: *Mapping kernels between rooted labeled trees beyond ordered trees*, JSAI-isAI Post-Workshop Proc., LNAI **9067**, 317–330, 2015.

[18] K. Hirata, Y. Yamamoto, T. Kuboyama: *Improved* MAX SNP*-hard results for finding an edit distance between unordered trees*, Proc. CPM'11, LNCS **6661**, 402–415, 2011.

[19] Y. Horesh, R. Mehr, R. Unger: *Designing an A* algorithm for calculating edit distance between rooted-unordered trees*. J. Comput. Bio. **13**, 1165–1176, 2006.

[20] Y. Ishizaka, T. Yoshino, K. Hirata: *Anchored alignment problem for rooted labeled trees*, New Frontiers in Artificial Intelligence, LNAI **9067**, 296–309, 2015.

[21] T. Jiang, L. Wang, K. Zhang: *Alignment of trees – an alternative to tree edit*, Theoret. Comput. Sci. **143**, 137–148, 1995.

[22] T. Kan, S. Higuchi, K. Hirata: *Segmental mapping and distance for rooted ordered labeled trees*, Fundam. Inform. **132**, 1–23, 2014.

[23] K. Kailing, H.-P. Kriegel, S. Schönauer, T. Seidl: *Efficient similarity search for hierarchical data in large databases*, Proc. EBDT'04. LNCS **2992**, 676–693, 2004.

[24] V. Kann: *Maximum bounded* 3*-dimensional matching is* MAX SNP*-complete*, Inform. Process. Lett. **37**, 27–35, 1991.

[25] 鹿島久嗣, 坂本比呂志, 小柳光生: 木構造データに対するカーネル関数の設計と解析, 人工知能学会論文誌 **21**, 1–9, 2006.

[26] KEGG: *Kyoto Encyclopedia of Genes and Genomes*, `http://www.kegg.jp/`.

[27] 木村大翼, 久保山哲二, 渋谷哲朗, 鹿島久嗣: 部分パスに基づいた木カーネル, 人工知能学会誌文誌 **26**, 473–482, 2011.

[28] P. N. Klein: *Computing the edit-distance between unrooted ordered trees*, Proc. ESA'98, 91–102, 1998.

[29] T. Kuboyama: *Matching and learning in trees*, Ph.D thesis, University of Tokyo, 2007.

[30] T. Kuboyama, K. Hirata, K. F. Kinoshita: *An efficient unordered tree kernel and its application to glycan classification*, Proc. PAKDD'08, LNAI **5012**, 184–195, 2008.

[31] T. Kuboyama, K. Shin, H. Kashima: *Flexible tree kernels based on counting the number of tree mappings*, Proc. MLG'06, 61–72, 2006.

[32] C. L. Lu, Z.-Y. Su, C. Y. Yang: *A new measure of edit distance between labeled trees*, Proc. COCOON'01, LNCS **2108**, 338–348, 2001.

[33] S.-Y. Lu: *A tree-to-tree distance and its application to cluster analysis*, IEEE Trans. Pattern Anal. Mach. Intell. **1**, 219–224, 1979.

[34] S. Luke, L. Panait: *A survey and comparison of tree generation algorithms*, Proc. GECCO'01, 81–88, 2001.

[35] T. Mori, T. Tamura, D. Fukagawa, A. Takasu, E. Tomita, T. Akutsu: *A clique-based method using dynamic programming for computing edit distance between unordered trees*, J. Comput. Bio. **19**, 1089–1104, 2012.

[36] C. H. Papadimitoriou, M. Yannakakis: *Optimization, approximation and complexity*, J. Comput. System Sci. **43**, 425–440, 1991.

[37] S. Schiermer, R. Giegerich: *Forest alignment with affine gaps and anchors, applied in RNA structure comparison*, Theoret. Comput. Sci. **483**, 51–67, 2013.

[38] S. M. Selkow: *The tree-to-tree editing problem*, Inform. Process. Lett. **6**, 184–186, 1977.

[39] J. Shawe-Taylor, N. Cristianini: *Kernel methods for pattern analysis*, Cambridge University Press, 2004.

[40] D. Shasha, J. T.-L. Wang, K. Zhang, F. Y. Shih: *Exact and approximate algorithms for unordered tree matching*, IEEE Trans. Sys. Man. and Cybernet. **24**, 668–678, 1994.

[41] 申吉浩: 木の半正定値カーネル – フレームワークとサーベイ, 人工知能学会論文誌 **24**, 459–468, 2009.

[42] K. Shin, M. Cuturi, T. Kuboyama: *Mapping kernels for trees*, Proc. ICML'11, 2011.

[43] K. Shin, T. Kuboyama: *A generalization of Haussler's convolution kernel - Mapping kernel and its application to tree kernels*, J. Comput. Sci. Tech. **25**, 1040–1054, 2010.

[44] W.-K. Sung: *Algorithms in bioinformatics: A practical introduction*, Chapman & Hall/CRC, 2009.

[45] K.-C. Tai: *The tree-to-tree correction problem*, J. ACM **26**, 422–433, 1979.

[46] L. G. Valiant: *The complexity of enumeration and reliability problems*, SIAM J. Comput. **8**, 410–421, 1979.

[47] G. Valiente: *An efficient bottom-up distance between trees*, Proc. SPIRE'01, 212–219, 2001.

[48] G. Valiente: *Algorithms on trees and graphs*, Springer, 2002.

[49] J. T. L. Wang, K. Zhang: *Finding similar consensus between trees: An algorithm and a distance hierarchy*, Pattern Recog. **34**, 127–137, 2001.

[50] Y. Wang, D. J. DeWitt, J-Y. Cai: *X-Diff: An effective change detection algorithm for XML documents*, Proc. ICDE'03, 519-530, 2003

[51] Y. Yamamoto, K, Hirata, T. Kuboyama: *Tractable and intractable variations of unordered tree edit distance*, Internat. J. Found. Comput. Sci. **25**, 307–329, 2014.

[52] W. Yang: *Identifying syntactic differences between two programs*, Software - Practice and Experience 21, no. 7, 739-755, 1991.

[53] T. Yoshino, S. Higuchi, K. Hirata: *A dynamic programming A^* algorithm for computing unordered tree edit distance*, Proc. IIAI AAI '13, 135–140, 2013.

[54] T. Yoshino, K. Hirata: *Hierarchy of segmental and alignable mapping for rooted labeled trees*, Proc. DDS 2013, 62–69, 2013.

[55] T. Yoshino, K. Hirata: *Alignment of cyclically ordered trees*, Proc. ICPRAM'15, 263–270, 2015.

[56] T. Yoshino, K. Hirata: *Enhanced Tai mapping for unrooted trees*, SISA'16, 6–11, 2016.

[57] T. Yoshino, K. Hirata: *Tai mapping hierarchy for rooted labeled trees through common subforest*, Theory of Comput. Sys. **60**, 759–783, 2017.

[58] T. Yoshino, Y. Ishizaka, K. Hirata: *Anchored alignment distance between rooted labeled unordered trees*, Proc. FedCSIS'17, 439–446, 2017.

[59] K. Zhang: *Algorithms for the constrained editing distance between ordered labeled trees and related problems*, Pattern Recog. **28**, 463–474, 1995.

[60] K. Zhang: *A constrained edit distance between unordered labeled trees*, Algorithmica **15**, 205-222, 1996.

[61] K. Zhang, T. Jiang: *Some* MAX SNP*-hard results concerning unordered labeled trees*, Inform. Process. Lett. **49**, 249–254, 1994.

[62] K. Zhang, D. Shasha: *Simple fast algorithms for the editing distance between trees and related problems*, SIAM J. Comput. **18**, 1245–1262, 1989.

[63] K. Zhang, D. Shasha: *Tree pattern matching*, in: A. Apostolico, Z. Galil (eds): *Pattern matching algorithms*, Oxford University Press, 341-371, 1997.

[64] K. Zhang, J. Wang, D. Shasha: *On the editing distance between undirected acyclic graphs*, Internat. J. Found. Comput. Sci. **7**, 43-58. 1996.