分岐のあるイベント構造の類似度測定

町澤　大輔†　　澄川 靖信††

† 拓殖大学大学院 工学研究科 情報デザイン専攻　〒 193-0985 東京都八王子市館町 815-1
†† 拓殖大学 工学部 情報工学科　〒 193-0985 東京都八王子市館町 815-1
E-mail: †25m312@st.takushoku-u.ac.jp, ††ysumikaw@cs.takushoku-u.ac.jp

あらまし　過去に起きたイベントを分析することには，類似する現代の諸問題の解決法を考える土台となる効果があることが知られている．このような分析を実現するために，入力したイベントに類似するものを出力する検索エンジンが有用であるが，既存の検索エンジンは線形に並べたイベント群のみを入出力の対象としている．本研究では，木構造で表現したイベント群を入出力とする検索エンジンを実現するために，木構造同士の類似度を評価するアルゴリズム TES を提案する．TES は，ラベルの完全一致性と木構造の構造的類似性を評価する木編集距離の先行手法を，ラベルの部分一致性と文脈の一致性を用いて構造的類似性を求めるように拡張する．TES の有効性を評価したところ，既存の木編集距離アルゴリズムと比較して，TES は約 63%も高い平均適合率が得られることを確認した．

キーワード　イベント，類似度測定，木構造，検索モデル

1　はじめに

異なる過去のイベント同士の類似性を分析することは，過去に発生したイベントから共通するパターンを抽出し，現代の諸問題を深く理解することが，将来起こり得る展開を予測する能力を育成する効果 [1] があることが知られており，近年重要視されている．

このような能力を育成するためには個々のイベントに注目するのではなく，依存関係や因果関係を持つ複数のイベントを包括的に解析することが重要である．

イベント群間の類似度を測定する手法 [5], [6] が提案されているが，これらの手法はイベントが時系列に並べられていることを前提としている．

しかし，現実世界で起きるイベントは，一つのイベントと複数のイベントの間に依存関係があるのが一般的である．例えば，パンデミックの発生時には，感染拡大に伴う医療体制の逼迫，政府による行動制限や政策対応など，複数のイベントが並行して進行する．既存手法は，本来なら分岐を持つグラフの部分的な側面だけに着目して類似度を測定しなければならない．

イベント群を木構造として表現することで，イベントの分岐を適切に表現でき，線形系列では捉えきれないイベント構造を保持したまま類似度を評価することができる．

木構造同士の類似性を測定する手法として，木編集距離（tree edit distance，以降 TED と呼ぶ）[7] [8] [9] が提案されている．TED は，ノード数の差を表すノードの削除と挿入，ラベルの差を表す置換の三種類の操作で二つの木間の距離を求める．しかし，従来の TED はノードラベルごとの完全一致性や各ノードだけに着目して距離を算出するため，イベント群同士の比較では生じる，異なる単語を使用しているが意味的には同じ部分一致性や，背景や結論が同じ依存関係を表す文脈を考慮した距離は算出しない．

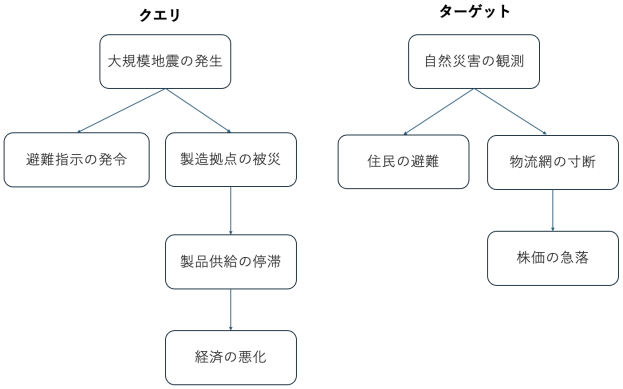

図 1　文脈の考慮の例

図 1 に文脈を考慮するのが適切な例を示す．本来であれば各ノードにはイベントの詳細が記述された文章を含めるが，本稿では紙面の都合上，イベントのタイトルだけを示し，文脈を考慮することの重要性を述べる．

クエリの「大規模地震の発生」とターゲットの「自然災害の観測」は，どちらも被害の起点となる災害事象を表している．しかし，従来の TED ではこれらを単なるラベルの不一致として扱い，編集コストを算出してしまう．その結果，意味的には同じだが，異なる単語が使われているイベント間の類似度を適切に評価できない．

さらに，背景や結論が同じ依存関係を表す文脈を考慮していない．ここで示すクエリとターゲットのイベント木は，共に災害による物流や生産体制の寸断を経て，経済的な損失という同じ文脈を持っている．しかし，クエリ側には「製品供給の停滞」というプロセスが記述されている一方で，ターゲット側では同様のプロセスが省略されている．従来の TED では，このノー

ドの有無を単純な構造の不一致として削除操作が行われる．その結果，イベントの流れが同一であるにもかかわらず，両者の距離が大きいと評価される．

本研究では，部分一致性と文脈情報を考慮しながらイベント群の類似度を求めるアルゴリズムである tree event similarity (TES) を提案する．TES は，既存の TED アルゴリズムとして広く認知されている Zhang– Shasha アルゴリズム（以降，ZS アルゴリズムと呼ぶ）に対して，親ノードの情報を伝播させる文脈ベクトルを定義し，文脈ベクトル同士の類似度を利用した編集コストを求めるように拡張する．

TES の有効性を評価するために，類似する木構造の組みを含むデータセットを構築し，木構造の類似度を求める先行手法と提案手法を比較した．この結果，提案手法は約 63%も高い平均適合率が得られたことを確認した．

2 関 連 研 究

本章では関連研究として，イベント系列および木構造データの類似度測定手法について述べる．

2.1 イベント構造の類似度測定

イベント構造の類似度測定に関する研究として，ECM [6] と PASS [5] が提案されている．ECM はイベント系列をクエリとして扱い，イベント間の時系列順序に制約を課した二部グラフを構築し，その最大重みマッチングを求めることで，因果関係を考慮した類似度測定を実現している．この手法により，単純な順序一致ではなく，因果的に対応するイベント同士の整合性を評価できる．PASS はイベント系列を配列と扱い，グローバルアライメント手法を応用することでイベント系列間の類似度を測定する手法である．二つの系列間で対応するイベントが存在しない場合にはギャップを挿入し，負のスコアを与えるペナルティ調整を行う．その結果，イベント系列に存在するノイズに影響を軽減しながら評価する．

一方で，ECM と PASS は線形構造のイベントを前提としており，複数のイベントが並行して進行する状況や，あるイベントから複数の派生的なイベントへと分岐するような非線形構造を評価できないという課題が存在する．

また，構造化データの類似度判定では，グラフカーネルや木カーネルを用いた手法も研究されている．Liu ら [3] は，因果グラフにおけるノードの意味的類似性とトポロジー構造の比較を統合したフレームワークを提案している．この手法では，各ノードを独立した変数として扱い，グラフ全体の構造をカーネル関数を通じて比較することで，因果グラフ間の類似性を定量化している．

しかし，これらのカーネルベースの手法では，ノードの意味は局所的に評価されるため，ノードがどのような親ノードから派生し，どのような文脈の中に位置付けられているかという情報が反映されない．これに対し，本研究では，イベント群を木構造として表現し，個々のイベントを単独で評価するのではなく，親ノードから継承される文脈情報を考慮した類似度を求める．

2.2 TED

木構造同士の類似度を定量化する手法として，TED が広く用いられている．TED は，一方の木構造を他方に変換するために必要なノードの挿入，削除，置換操作に対し，それらに割り当てられたコストの総和を最小化する問題として定義される．

Tai [7] によって，TED の概念を初めて定式化し，動的計画法による解法を提案した．その後，Zhang と Shasha [9] は，Tai の手法における冗長な計算を排除し，各ノードの左端の葉と KeyRoots に着目することで，より効率的なアルゴリズムを確立した．

一方で，TED を自然言語処理や知識構造の比較に適用しようとする試みもなされてきた．従来の TED はノードラベルの完全一致を前提としていたが，Wang ら [8] は，置換コストにベクトルから算出される類似度を導入することで，表層的なラベルの一致に依存せず，意味的に近いノード同士の対応関係を考慮した計算を行っている．

しかし，これらの既存手法は編集コストの算出が比較対象となるノード単体のラベルのみに依存している．既存の TED アルゴリズムでは，これらの文脈差を動的にコストへ反映する仕組みを持たないため，記述の詳細や省略に伴う構造の微差を捉えることができず，適切に評価できない．

そのため，本研究が提案する手法は，ZS アルゴリズムを拡張して，親から子ノードへの情報を伝搬する仕組みを導入する．

3 ZS アルゴリズム

本研究で採用する ZS アルゴリズムは，TED を効率的に算出するための動的計画法に基づく手法である．TED の計算は，単純な再帰な方法で行うと部分問題が重複し，計算量が増大する．ZS アルゴリズムは，この重複を回避するために，木構造中の特定のノード集合である KeyRoots に着目する．KeyRoots は，計算の最適化において重要な役割を果たすノードであり，これにより TED の計算問題を，より管理しやすい部分木間または部分森間の距離計算へと分解して処理する．その結果，重複計算を抑制しつつ効率的な TED 算出を実現する．

3.1 定　　義

まず，各木 T に対して後順走査を行い，ノードに 1 から $|T|$ までの番号を付与する．このとき，$T[i]$ は後順番号 i に対応するノードを表すものとする．次に，ノード i を根とする部分木に含まれる葉ノードのうち，最も番号が小さいノードのインデックスを $L(i)$ と定義する．

$$L(i) = \{j \mid j \in \text{Descendants}(i)\} \tag{1}$$

Descendants(i) はノード i のすべての子孫ノードの集合を意味する．このノード i が葉ノードである場合には $L(i) = i$ が成り立つ．また，$T[L(i)..i]$ は，後順ノード列においてノード $L(i)$ から i までの連続区間から構成される順序付き部分森

とする．

3.2 KeyRoots の選定

すべての部分森の組に対して計算することは計算量の観点から冗長であり，ZS アルゴリズムでは，計算に必要な最小限のノード集合として KeyRoots(T) を定義する．

$$KeyRoots(T) = \{k \mid k \in \{1, \ldots, |T|\}, L(k) \neq L(k{+}1)\} \cup \{|T|\} \tag{2}$$

この定義により，KeyRoots(T) は右兄弟を持つノードと木の根ノードから構成される集合となる．

アルゴリズムは，T_1 と T_2 のそれぞれについて得られた KeyRoots(T_1) と KeyRoots(T_2) のすべての組 (s, t) に対して処理する．

3.3 森の距離計算

各 KeyRoots の組 (s, t) に対して，部分森 $T_1[L(s)..i]$ と $T_2[L(t)..j]$ の間の距離 $D_{forest}(i, j)$ を計算する．ここで，i および j はそれぞれ $L(s) \leqq i \leqq s$，$L(t) \leqq j \leqq t$ を満たす．$D_{forest}(i, j)$ は，以下の漸化式に基づいて動的計画法により求める．

$$D_{forest}(i,j) = \min \begin{cases} D_{forest}(i-1, j) + Del(T_1[i]), \\ D_{forest}(i, j-1) + Ins(T_2[j]), \\ D_{forest}(L(i)-1, L(j)-1) + D_{tree}(i, j), \\ D_{forest}(i-1, j-1) + Rep(T_1[i], T_2[j]). \end{cases} \tag{3}$$

ただし，第 3 項は $L(i) = L(s)$ かつ $L(j) = L(t)$ の場合に適用され，それ以外の場合には第 4 項を適用する．

ここで，Del，Ins，Rep はそれぞれ削除，挿入，置換操作のコストを表す．$D_{tree}(i, j)$ は，ノード i および j を根とする部分木間の編集距離である．この値は，$i = s$ かつ $j = t$ となる時点で確定し，以降の計算に再利用する．第 3 の条件分岐は，現在のノード対が対象部分木の根に対応する場合に適用され，事前に計算された森の距離と部分木距離を統合する役割を果たす．最終的に，木 T_1 と T_2 の TED は，$D_{tree}(|T_1|, |T_2|)$ として得られる．

4 TES アルゴリズム

本研究では，ZS アルゴリズムをベースに，イベント同士の部分的一致性と文脈情報を考慮するための二つの拡張を導入する．一つ目の拡張点は，親ノードの意味情報を子ノードへ再帰的に伝播する機構の導入である．各ノードが保持するイベントのテキストに対して，その親ノードが持つ意味的情報を統合することで，これまでの流れを反映した文脈表現を持つ．

二つ目の拡張点は，編集操作におけるコストを動的に算出する枠組みの導入である．挿入・削除・置換の各編集操作に対し，対応するノード間の意味的類似度や親子関係に基づく文脈の整合性を考慮してコストを調整する．

以降ではこれらの拡張点の詳細を述べる．

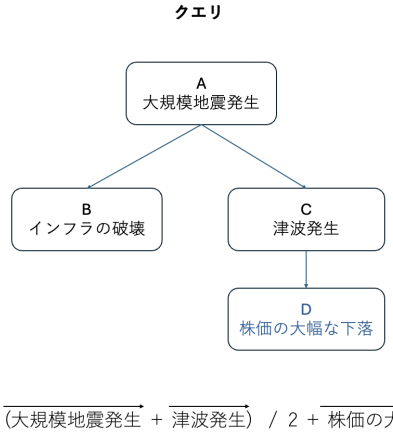

図 2　文脈情報を統合するベクトル計算の例

4.1 文脈情報の統合

TES では，各ノードが保持するテキスト情報に加え，根ノードから当該ノードに至るまでの各ノードのテキストを文脈情報として抽出し，ノードのベクトル表現に統合する．このアプローチの目的は，共通の背景や依存関係を持つイベント間の対応付け精度の向上である．ノード単位の局所的なベクトル表現のみでは，語彙の揺らぎや記述の抽象度の差異により，類似イベント間の比較が困難となる．そのため，親ノードからテキスト情報を統合することで，イベントの流れ全体を含めるようにする．

TES における文脈情報の付与は，図 2 のように親から子ノードへ情報を再帰的に伝播させることで実現する．各ノードは，自身のテキスト情報に加え，根から現在に至るまでの流れを保持したベクトル表現を得る．

文脈情報の統合処理を，以下のように定式化する．木 T に含まれる各ノード n に対し，初期ベクトルを $\mathbf{v}_n$ とする．文脈を考慮したベクトル表現 $\mathbf{c}_n$ は，次式によって計算する．

$$\mathbf{c}_n = \begin{cases} \mathbf{v}_n & \text{if } n \text{ is root} \\ \frac{\mathbf{v}_n + \mathbf{c}_{P(n)}}{2} & \text{otherwise} \end{cases} \tag{4}$$

ここで $P(n)$ はノード n の親ノードを表す．以降のノード同士の類似度計算ではこのベクトル表現を用いて行う．

4.2 編集コストの動的調整

TES では，TED の挿入・削除・置換コストを固定せず，文脈情報に基づいて動的に算出する．これにより，共通の背景や依存関係を持つイベント間の不要な編集コストを抑制し，文脈を反映した類似度評価を可能とする．

4.2.1 挿　　入

TES におけるクエリへのノード挿入は，ターゲット側で保

持される詳細情報をクエリ側に導入し，情報の欠落を補完する操作とみなす．クエリ側にノードを挿入すべきかを判断するために，親子関係に基づく文脈の整合性を考慮し，以下の二つの条件に基づいて挿入コストを低減する．

1. **ターゲット側の文脈の連続性：** ターゲット側において挿入候補ノードがその親ノードと高い類似度を持つ場合．これは，当該ノードが親ノードから派生した具体的な詳細情報であり，文脈的に連続していることを意味する．
2. **クエリ側の情報の欠落：** クエリ側において対応する箇所の親子間類似度が低い場合．これは，クエリ側の記述で中間プロセスが省略され，情報の欠落が生じていることを意味する．

具体例として，図 3 に示すような状況を考える．ここでは，クエリが「大規模地震発生」から「株価の大幅な下落」という中間プロセスを省略した記述であるのに対し，ターゲットは「大規模地震発生」,「津波による工業地帯の壊滅」,「株価の大幅な下落」という因果の連鎖を詳細に記述した構造を持つ．

クエリの構造に着目すると，自然災害である地震から，株価の変化へと直接的な接続がされている．そのため，親子間の意味的な類似度は低く，中間プロセスが省略され，情報の欠落が生じていることを意味する．

一方で，ターゲットにおける中間プロセスである「津波による工業地帯の壊滅」は，親ノードである地震の発生によって引き起こされる具体的な被害状況を表している．したがって，両者の間には，意味的な類似度は高くなる．

このような状況下において，クエリで発生している中間プロセスの省略による情報の欠落に対し，ターゲット側で保持されている整合性の取れた詳細情報を埋め込むことで，文脈を補完する．このような場合に，該当の挿入操作に対する編集コストを動的に低減する．

これに基づき，挿入コストを以下のように定義する．

$$Cost_{ins}(t,n) = \alpha \cdot \big(1 - Sim(\mathbf{c}_t, \mathbf{c}_{P(t)})\big) + (1-\alpha) \cdot Sim(\mathbf{c}_n, \mathbf{c}_{P(n)}) \tag{5}$$

第 1 項では，挿入対象ノード t がターゲット側において親ノード $P(t)$ と高い類似度を持つ場合，それを詳細情報とみなし，挿入コストを低減する．第 2 項では，挿入先となるクエリ側のノード n とその親 $P(n)$ との類似度である．この類似度が低い場合，クエリ側に情報の欠落が存在すると判断できるため，その空白を埋めるためにコストを低く設定する．なお，α は，ターゲット側の文脈の連続性とクエリ側の情報の欠落のどちらを重視するかを調整する重み係数である..

4.2.2 削　　除

TES におけるクエリからのノード削除は，情報の省略に伴う欠落とみなす．クエリ側のノードを削除するかどうかを判断するために，親子関係に基づく文脈の整合性を考慮し，以下の条件に基づいて削除コストを低減する．

1. **ターゲット側の文脈の飛躍：** ターゲット側において対応するノードとその親ノードとの類似度が低い場合．これは，ターゲット側で中間プロセスが省略され，因果関係が簡潔に記述されていることを意味する．
2. **クエリ側の詳細情報：** クエリ側において削除候補となるノードがその親ノードと高い類似度を持つ場合．これは，当該ノードが親イベントを補足する詳細情報であり，削除しても文脈を損なわないことを意味する．

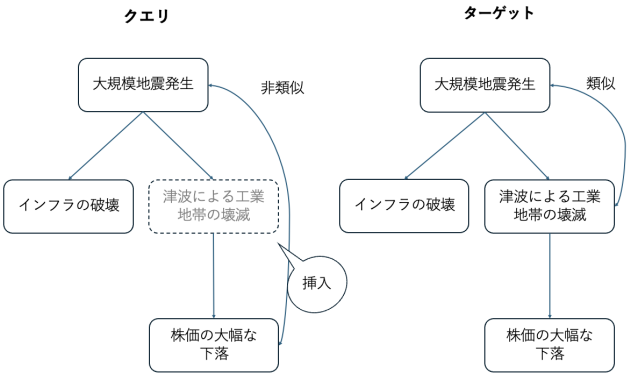

図 3　挿入する際の例

図 4 に削除する際の具体例を示す．この例では，ターゲット側において「大規模地震発生」から「株価の大幅な下落」へと直接接続された構造に着目する．ここでは，災害事象から経済事象へと文脈が飛躍しており，親子間の意味的類似度は低い．これは，ターゲット側で因果関係の中間プロセスが省略され，結果のみが要約して記述されていることを意味する．

対照的に，クエリ側に存在する中間ノード「津波による工業地帯の壊滅」は，親ノードである「大規模地震発生」から引き起こされる直接的な被害と関連するため，親子間の類似度は高い．これは，当該ノードが文脈を変える独立した話題ではなく，親ノードの内容をより具体的に説明する詳細情報である．TES では，ターゲット側に文脈的飛躍が見られ，かつクエリ側の削除候補が親ノードと高い類似関係を持つ場合，この削除はターゲット側の省略に合わせた文脈適合とみなし，削除コストを低減する．

これに基づき，削除コストを以下のように定義する．

$$Cost_{del}(n,t) = \beta \cdot \big(1 - Sim(\mathbf{c}_n, \mathbf{c}_{P(n)})\big) + (1-\beta) \cdot Sim(\mathbf{c}_t, \mathbf{c}_{P(t)}) \tag{6}$$

ここで，第 1 項はクエリ内の情報の詳細度に依存する項であり，ノード n とその親 $P(n)$ の類似度が高いほど，そのノードは補足情報と判断され，コストが低下する．第 2 項はターゲット側の文脈の連続性に依存する項である．ターゲット側における対応箇所のノード t とその親 $P(t)$ の類似度を表す．これら類似度が低い場合，ターゲット側で中間プロセスが省略されているとし，それに削除コストを低く設定する．なお，β は，ターゲット側の文脈の連続性とクエリ側の詳細情報のどちらを重視するかを調整する重み係数である．

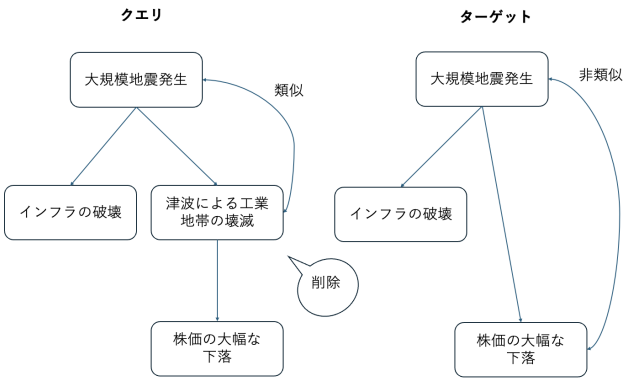

図 4　削除する際の例

4.2.3　置　　換

従来の標準的な TED の定義において，置換コストはノードラベルの完全一致を前提とした二値的な評価基準が用いられる．このような基準は，プログラムコードや数式の比較においては有効であるものの，テキストデータには適さない．テキストデータにおいては，同義語の使用や表現の揺らぎが生じる．例えば，「家屋の倒壊」と「建物の損壊」は，文字列としては異なるが意味的には同一である．完全一致を前提とする手法では，このようなノードのペアを不一致として扱い，対応付けができないという課題がある．この課題を解決するため，TES では記号的な表層の一致ではなく，意味的な類似性に基づく動的なコストを導入する．クエリ側のノード n をターゲット側のノード t へ置換する際のコスト $Cost_{upd}(n,t)$ は，各ノードのベクトル間の類似度 Sim を用いて以下のように定義する．

$$Cost_{upd}(n,t) = 1 - Sim(\mathbf{c}_n, \mathbf{c}_t) \qquad (7)$$

ここで，Sim はベクトル間の類似度を計算する関数であり，テキスト記述が完全には一致しない場合であっても，意味が類似していればコストは 0 に近づく．

4.3　アルゴリズム

提案手法のアルゴリズムは ZS アルゴリズムを拡張したものであり，Algorithm 1 と Algorithm 2 で構成されている．

Algorithm 1 は，木構造上の各ノードに対して文脈情報を親ノードから子ノードへと伝播させ，ベクトルを生成する手法を示している．

入力として与えられる T はイベント構造を表す木であり，V_{raw} は各ノードに対応する元のベクトルである．

ノード $node$ とその親ノードの文脈ベクトル $parent\_vec$ を入力として受け取り，$node$ に対応するベクトルを計算する（4 行目）．この関数は，木構造に沿って親ノードから子ノードへと深さ優先で呼び出される．$V_{raw}(node)$ は，現在対象となっているノード $node$ に対応する元のベクトルを取得する処理である（2 行目）．ノード $node$ が木の根ノードである場合には，親ノードが存在しないため，ベクトル $CV(node)$ は元のベクトル V_{curr} と設定する（4 行目）．

一方，$node$ が根以外のノードである場合には，式 (4) と同じ式で，親ノードのベクトル表現を含んだベクトルを生成する (6 行目)．この再帰処理によって，木全体が走査され，すべてのノードについて文脈ベクトルが計算する．最終的に，すべてのノードに対するベクトルを格納した集合 CV が返され，後続の TED 計算において，ノード間の意味的類似度評価に用いる．

Algorithm 2 は ZS アルゴリズムを拡張して編集コストを動的に計算する．PostOrderNodes(T_1) および PostOrderNodes(T_2) は，それぞれ木構造 T_1, T_2 に対して後順走査を行い，ノードを走査順に並べた列 A および B を生成する関数である（1-2 行目）．LeftmostLeafIndices(A) および LeftmostLeafIndices(B) は，ノード列 A, B に含まれる各ノードについて，そのノードを根とする部分木における左端葉ノードのインデックスを計算する関数であり，それぞれ L_1，L_2 として保持する（3-4 行目）．

KeyRoots(A) および KeyRoots(B) は，ノード列 A，B から KeyRoots の集合を抽出する（5-6 行目）．TD は，ノード列 A と B の任意の接頭部分に対する TED を格納するための行列である（7 行目）．具体的には，行列の各要素 $TD_{x,y}$ は，A の先頭から x 番目までのノードと B の先頭から y 番目までのノードから構成する部分木同士の最小編集距離を表す．

その後，KeyRoots(A) および KeyRoots(B) によって得られたすべての KeyRoots の組 (i,j) に対して，対応する部分木間の編集距離を計算する（8 行目以降）．i_{off} および j_{off} は，それぞれ KeyRoots i および j を根とする部分木において，左端葉ノードの直前に位置する後順ノード列上のインデックスを表す変数である（10-11 行目）．m および n は，それぞれ比較対象となる二つの部分木に含まれるノード数を表す（12-13 行目）．表 FD は，KeyRoots (i,j) に対応する部分木同士の編集距離を算出するための表である（14 行目）．表 $FD_{x,y}$ は，部分木 $A[i_{\text{off}}+1, i]$ のうち先頭から x 個のノードと，部分木 $B[j_{\text{off}}+1, j]$ のうち先頭から y 個のノードから構成する二つの部分構造間の最小編集距離を表す．u，v は現在処理しているノードを表す．

CostDel および CostIns は，それぞれノード削除およびノード挿入に伴うコストを計算する関数であり，u，v とそれぞれの親ノードとの類似度を式（５）と式（６）と同じ計算する (25-26 行目)．置換操作に対応するコスト c_{upd} は，対応する二つのノード u，v のベクトル間のコサイン類似度を用いてコストを導出する（27 行目）．動的計画法の主ループでは $FD_{x,y}$ の各セルに対して，削除，挿入，置換の三つの操作に対応するコストを算出し，その最小値を選択することで部分木間の最小編集距離を更新する（21 行目以降）．

左端葉インデックスが一致する場合には，現在のセルが部分木の根同士の対応に相当するため，$FD_{x,y}$ の値を木全体の行列 $TD_{x+i_{\text{off}},\, y+j_{\text{off}}}$ に格納する．一方，一致しない場合には，既に計算済みの部分木距離 TD を参照することで，部分木単位の距離を再利用し，計算の冗長性を抑制する．以上の処理をすべて

Algorithm 1 文脈統合アルゴリズム

Input: Tree T, Raw vectors V_{raw}
Output: Context vectors CV

function F($node, parent\_vec$)
  $V_{curr} \leftarrow V_{raw}(node)$
  if $node$ is Root **then**
    $CV(node) \leftarrow V_{curr}$
  else
    $CV(node) \leftarrow \frac{V_{curr} + parent\_vec}{2}$
  end if
  for each $child$ in $node.children$ **do**
    F($child, CV(node)$)
  end for
end function
return CV

の KeyRoots の組に対して繰り返すことで，木全体の編集距離を動的計画法に基づいて算出し，最終的に TD の右下の値を類似度として返す．

5 評価実験

本章では，TES の有効性を検証するために，事前に作成したデータセットを用いた評価実験の設計と結果について述べる．TES の性能を既存手法と比較し，どの程度精度向上が得られるかを評価する．

5.1 データセット

本研究では，TES の有効性を検証するための評価用データセットとして，StoryNetworks [4] を使用した．StoryNetworks は，2016 年から 2017 年にかけての英語版 Wikipedia の Current events ポータルに掲載された記事を基に構築されたデータセットである．本データセットは，現実世界で発生したイベントをイベント単位で整理し，それらの因果関係や時間的依存関係をアノテーションしている．

各イベントは短い自然言語テキストとして記述されている．そして，イベント間の関係は有向グラフとして表現されており，イベント同士の因果関係や前後関係を構造的に整理しており，あるイベントが別のイベントの原因となる場合や，同一のイベントから複数の派生的なイベントが発生するような関係性を表現している．また，各イベントには「政治・選挙」「災害・事故」「経済」「国際関係」などのカテゴリラベルが付与されている．

一方で，元の StoryNetworks データセットには，複数の分岐を含む複雑なイベントグラフだけでなく，単一のイベントが時間順に直線的に連なっただけの単純なイベント系列も多数含まれている．このような線形構造のみから成るイベントグラフは，時系列的な関係性の分析には適しているものの，イベントの分岐や並行的な進行といった構造的な複雑性を十分に含んでいない．本研究の目的は，イベントの分岐構造や並行的な進行を内包する複雑なイベントに対して，TES がどの程度有効に機能するかを評価することである．

そこで本研究では，データセット全体に対して事前処理として，すべてのノードが最大一つ以下の子ノードしか持たないグラフを除外し，少なくとも 1 箇所の分岐を有する木構造のみを抽出した．

このフィルタリングの結果，木構造を有するイベントグラフ

Algorithm 2 TES アルゴリズム

Input: Tree T_1, T_2,
Output: Tree edit distance $TD_{|T_1|,|T_2|}$

$A \leftarrow$ PostOrderNodes(T_1)
$B \leftarrow$ PostOrderNodes(T_2)
$L_1 \leftarrow$ LeftmostLeafIndices(A)
$L_2 \leftarrow$ LeftmostLeafIndices(B)
$KR_1 \leftarrow$ KeyRoots(A)
$KR_2 \leftarrow$ KeyRoots(B)
$TD \leftarrow$ zero matrix of size $|A| \times |B|$
for $i \in KR_1$ **do**
  for $j \in KR_2$ **do**
    $i_{\text{off}} \leftarrow L_{1,i} - 1$
    $j_{\text{off}} \leftarrow L_{2,j} - 1$
    $m \leftarrow i - i_{\text{off}}$
    $n \leftarrow j - j_{\text{off}}$
    $FD \leftarrow$ zero matrix of size $(m+1) \times (n+1)$
                                ▷ Initialization
    for $x \in [1, m]$ **do**
      $FD_{x,0} \leftarrow FD_{x-1,0} + \text{CostDel}(A_{x+i_{\text{off}}})$
    end for
    for $y \in [1, n]$ **do**
      $FD_{0,y} \leftarrow FD_{0,y-1} + \text{CostIns}(B_{y+j_{\text{off}}})$
    end for
                                ▷ Calculation of each cell
    for $x \in [1, m]$ **do**
      for $y \in [1, n]$ **do**
        $u \leftarrow A_{x+i_{\text{off}}}$
        $v \leftarrow B_{y+j_{\text{off}}}$
        $c_{\text{del}} \leftarrow \text{CostDel}(u, v)$
        $c_{\text{ins}} \leftarrow \text{CostIns}(v, u)$
        $c_{\text{upd}} \leftarrow 1 - \text{CosineSim}(u, v)$
        if $L_{1,x+i_{\text{off}}} = L_{1,i}$ **and** $L_{2,y+j_{\text{off}}} = L_{2,j}$ **then**
          $FD_{x,y} \leftarrow \min\Big(FD_{x-1,y} + c_{\text{del}},$
                $FD_{x,y-1} + c_{\text{ins}},$
                $FD_{x-1,y-1} + c_{\text{upd}}\Big)$
          $TD_{x+i_{\text{off}},\, y+j_{\text{off}}} \leftarrow FD_{x,y}$
        else
          $p \leftarrow L_{1,x+i_{\text{off}}} - 1 - i_{\text{off}}$
          $q \leftarrow L_{2,y+j_{\text{off}}} - 1 - j_{\text{off}}$
          $FD_{x,y} \leftarrow \min\Big(FD_{x-1,y} + c_{\text{del}},$
                $FD_{x,y-1} + c_{\text{ins}},$
                $FD_{p,q} + TD_{x+i_{\text{off}},\, y+j_{\text{off}}}\Big)$
        end if
      end for
    end for
  end for
end for
return $TD_{|A|,|B|}$

が 78 件となった．これらを本研究における実験用データセットとして採用した．

5.2 実験設計

本節では，抽出した 78 件のイベントグラフデータを用い，TES での有効性を検証するための実験設計について述べる．既存手法との比較実験を行い，適合率および平均適合率を定量的に評価する．

5.3 評価方法

本研究では，TES の性能を評価するために，情報検索や推薦システムの評価で広く用いられる指標である上位 k 件における適合率（Precision at k; P@k） および平均適合率（mAP）を採用した．P@k は，検索結果や推薦リストの上位 k 件における正解の割合を示す指標であり，TES がどの程度効率的に関連する項目を上位に返せるかを評価である．一方，mAP は各クエリに対する平均適合率を算出し，複数の評価対象における全体的なランキング性能を定量化する指標である．

評価には，本研究で独自に作成した正解データセットを使用した．正解データセットは，二名の評価者による二段階の検証プロセスを経て構築した．まず一名の評価者が，元となる 78 件のデータセットに含まれるすべての組み合わせを確認し，内容的に適切と判断されるペアを作成した．次に，もう一名の評価者がレビューを行い，必要に応じて誤って含まれたペアや不適切なペアを削除した．この手順により，最終的に 83 件の正解データが得られた．

この正解データを用いて，TES および比較手法の性能を評価した．

5.4 ハイパーパラメータ設定

TES は二つのハイパーパラメータに依存する．これらは，TED 計算におけるコストの重み係数 α, β である．各パラメータの役割は以下の通りである．

- **重み係数** α: 式 (5) における挿入コスト $Cost_{ins}$ を制御する．これは，ターゲット側のノードが持つ詳細情報の重要性と，クエリ側における情報の欠落のどちらを重視するかを決める．
- **重み係数** β: 式 (6) における削除コスト $Cost_{del}$ を制御する．クエリ側のノードが詳細情報である可能性と，ターゲット側での情報の欠落の度合いを評価する．

各パラメータ α, β を $[0.0, 1.0]$ の範囲において 0.1 刻みで変化させ，グリッドサーチを実施した．評価指標には平均適合率を採用し，最大化する値を探索した．実験の結果，最も高い検索精度が得られた $\alpha = 0.8$ $\beta = 0.4$ を本研究の設定として採用する．

5.5 比較手法

TES の有効性を検証するため，以下の 二つの手法と比較する．テキストノードのベクトルは，すべての手法で同一の学習済み BERT モデル [1]を用いて生成し，類似度はコサイン類似度で算出する．

1. **ZS アルゴリズム：** ZS アルゴリズムに基づき，挿入・削除の編集コストを 1.0 とし，置換の編集コストは，各ノードのテキスト類似度によって定義する TED である．
2. **Wang らの手法：** 各ノードのテキスト類似度と，ハンガリー法を用いた最適な子ノードの対応づけを行う手法である．
3. **木カーネル：** 二つの木構造に共通する部分木に基づき，類似度を算出する手法 [2] である．
4. **文脈統合のみ：** TES の構成要素のうち，文脈情報の統合のみを適用し，編集コストについては一律 1.0 に固定して算出する手法である．
5. **動的コストのみ：** TES の構成要素のうち，文脈情報の統合を行わず，各ノードの元のベクトルのみを用いて，編集コストの動的調整を適用する手法である．

表 1　検索精度に関する結果

手法	P@1	P@3	P@5	P@10	mAP
木カーネル	0.205	0.091	0.077	0.043	0.152
ZS アルゴリズム	0.205	0.144	0.095	0.066	0.254
Wang らの手法	0.182	0.106	0.082	0.055	0.218
文脈統合のみ	0.227	0.152	0.109	0.089	0.284
動的コストのみ	0.023	0.015	0.027	0.036	0.082
TES	**0.318**	**0.220**	**0.182**	**0.114**	**0.413**

5.6 実験結果

実験の結果を表 1 に示す．表より，文脈情報の統合と動的コスト調整の両者を組み合わせた TES は，mAP において 0.413 を達成した．これは標準的な TED の 0.254 と比較して約 63% の精度向上であり，TES の有効性が定量的に確認された．

P@k においても，TES はすべての k で最高値を記録した．特に，P@1 では 0.318 を達成し，ZS アルゴリズムの 0.205 から 0.113 ポイント向上した．P@3, P@5, P@10 においても，TES はそれぞれ 0.220, 0.182, 0.114 を記録し，いずれも比較手法の中で最大の値であった．

また，文脈統合のみを適用した条件では mAP 0.284 を記録した．これは標準的な TED を 0.03 ポイント上回り，ノードのベクトル表現自体に文脈情報を含めることが有効であることを示している．しかし，動的コストのみを適用した条件では，mAP 0.082，P@1 0.023 となり，他の手法と比べると低下した．これは，動的コストのみを調整しただけでは正確に文脈を捉えることができないことを示している．ただし，これら二つの手法を統合した TES では，mAP 0.413 と精度が向上した．これは，文脈情報の統合と動的なコスト調整が相乗的に作用した結果であると考えられる．

1：https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

5.7 考　　察

ZS アルゴリズムや Wang らの手法らなどの TED をベースとした手法はノードの増減に敏感に捉える傾向がある．イベント内容が類似していても，記述の詳細度や省略に伴うノード数の変化によって評価が変わってしまい，精度が低下する．これに対し，TES では，イベント間の文脈的連続性を考慮した動的なコスト計算を導入している．そのため，省略や詳細化については低コストで許容され，多少ノード数が増減していても類似しているものとして評価する．この点が mAP や P@k の向上につながったと考えられる．

また，木カーネルは部分木での一致を集約する手法であり，イベントグラフ全体の構造を直接的に捉えることは困難である．そのため，イベント系列全体の順序を適切に捉えられず，構造的な対応関係を十分に反映できない．一方で，TES は木全体を比較するため，この違いが評価の向上に寄与したと考えられる．

次に，動的コストのみの導入で精度が悪化した理由は，コスト計算が記述の類似度のみで文脈を判断したためである．イベントの流れを無視して，文章が類似しているという基準だけで編集操作が選択される．その結果，本来は別の展開であるイベント間での不適切な対応付けが多発し，精度が低下したと考えられる．

最後に，文脈統合のみを導入した手法が標準的な TED を上回った理由は，親ノードからの文脈情報によって，各ノードがイベントの流れを反映したベクトルを獲得したためである．個別のラベルの一致度のみに依存せず，根からの推移を反映して類似度が算出される．その結果，語彙の選択や表現の抽象度が異なる場合でも，同じの文脈を持つノード同士の整合性が保たれ，精度が改善したと考えられる．

最終的な TES ではすべての手法よりも最も高い値を達成しており，文脈情報を付与したベクトル表現に基づいて編集コストを動的に変化させることが，非線形イベント構造の類似度測定において有効である．

以上の結果から，TES は，複雑な分岐構造を含むイベントを対象とした検索において有効であることが示された．

6 ま と め

本研究では，複数の分岐や並行的な進行があるイベント構造に対し，その類似性を評価するための新たな手法として TES を提案した．

従来のイベント類似度測定手法は，線形構造のイベントを扱うため，非線形性を捉えきれないという課題があった．一方，木構造を用いる既存の TED 手法は，ノードラベルの文字列一致や，固定的なコストに基づく構造的一致を前提としていた．その結果，異なる単語で表現されているが意味的には等価なイベントや，記述の詳細度は異なるが，背景や結論といった構造的役割が同じイベントを適切に評価できない．

本研究では ZS アルゴリズムを拡張し，以下の二つの改良を行った第一に，親ノードから子ノードへと情報を再帰的に伝播する機構である．これにより，各ノードは局所的なテキスト情報だけでなく，根ノードから自身に至るまでのイベントの経緯や流れを反映したベクトル表現を獲得する．第二に，この文脈ベクトルを用いて編集コストを動的に算出する拡張である．挿入・削除・置換の各編集操作に対し，従来の一律なコストではなく，文脈の連続性や意味的類似度に基づいてコストを調整する．これにより，詳細化や省略による構造的な不一致に影響されずに評価できる．

提案手法の有効性を検証するために実施した評価実験において，TES は既存の TED 手法および関連手法をすべての評価指標で大幅に上回る性能を示した．特に，ランキング全体の品質を示す mAP において，ベースラインの ZS アルゴリズムに対し約 63%という精度が向上した．また，P@1, P@3, P@5, P@10 の各指標においても一貫して最高値を記録したことは，TES が単に最上位の結果を改善するだけでなく，関連性の高いイベント群をランキング全体にわたって安定して上位に配置できていることを示している．

今後の課題としては，木構造に限定されない一般的な因果グラフへの拡張が挙げられる．

文　　献

[1] Veronica Boix-Mansilla. Historical understanding: Beyond the past and into the present. *Knowing, teaching, and learning history: National and international perspectives*, pp. 390–418, 2000.

[2] Michael Collins and Nigel Duffy. Convolution kernels for natural language. In *Proceedings of the 15th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, p. 625 – 632, Cambridge, MA, USA, 2001. MIT Press.

[3] Ning-Yuan Georgia Liu, Flower Yang, and Mohammad S. Jalali. Measuring similarity in causal graphs: A framework for semantic and structural analysis, 2025.

[4] Daisuke Machizawa, Naoki Sawahata, Ryohei Ikejiri, and Yasunobu Sumikawa. Storynetworks: An annotated dataset of event dependencies from short descriptions. In *New Trends in Theory and Practice of Digital Libraries*, pp. 46–56, Cham, 2026. Springer Nature Switzerland.

[5] Daisuke Machizawa, Naoki Sawahata, and Yasunobu Sumikawa. Similarity measurement between event sequences with penalty adjustment. WI-IAT '25, 2025.

[6] Yasunobu Sumikawa. Event causal relationship retrieval. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, WI-IAT '21, p. 318 – 325, New York, NY, USA, 2022. Association for Computing Machinery.

[7] Kuo-Chung Tai. The tree-to-tree correction problem. *J. ACM*, Vol. 26, No. 3, p. 422 – 433, July 1979.

[8] Guanghui Wang, Jinze Yu, Xing Zhang, Dayuan Jiang, Yin Song, Tomal Deb, Xuefeng Liu, and Peiyang He. Sted and consistency scoring: A framework for evaluating llm structured output reliability, 2025.

[9] Kaizhong Zhang and Dennis Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, Vol. 18, No. 6, pp. 1245–1262, 1989.