コルモゴロフ複雑性の新規概算法に基づく仕様間の整合性判定

藤原由希子† 五藤智久† 谷幹也†

我々は，仕様間の整合性判定として，仕様間の類似度をコルモゴロフ複雑性の従来の概算法を用いて計算し，類似度のランク上位の仕様間を整合と判定する方法を提案してきた．本稿では，コルモゴロフ複雑性の新規な概算法を提案する．提案法は，2 つのプロジェクトにおけるランク上位の判定精度を用いて評価した．その結果，提案法の判定精度は従来の概算法以上だった．また，従来法である潜在的ディリクレ配分やベクトル空間法と比較したところ，提案法の判定精度が最も高かった．

A Consistency Detecting Method of Software Specifications using a New Approximate Calculation of Kolmogorov Complexity

Yukiko Fujiwara† Tomohisa Gotoh† and Mikiya Tani†

We have proposed a method of consistency detection between software specifications, in which the similarities of specification pairs are calculated using the approximate calculation method of Kolmogorov complexity and the specifications pairs corresponding to top-rank similarities are predicted as consist. In this paper, we propose a new approximate calculation method of Kolmogorov complexity. We evaluated our method using top-rank detection accuracy in two projects. We compared our method with the previous approximate calculation method and the results showed that our method performed equal or better than the previous method. Also, we compared our method with latent Dirichlet allocation and vector space model, which was used in predicting the traceability links. The results showed that our method performed better than the other methods.

1. はじめに

システム開発における仕様は，機能要件，画面レイアウトや帳票など，様々な観点で文書に記述される．記述の粒度も概要から詳細まで様々であり，記述形式も，自然言語テキストや図表，サンプルスクリプトなど多様である．システムを破綻なく構築するためには，これらの多様な観点・粒度・形式で記述された仕様が互いに矛盾なく対応漏れなく整合している必要がある．

従来，仕様間の整合性は，主に人手でのキーワード検索などにより確認されている．しかし，人手でキーワードを設定しながらの大量文書の確認はコスト（人件費と時間）がかかる．すべてを確認できずに不整合を見逃す可能性もある．

そのため，仕様間の関連性を調べる方法として，ベクトル空間法や潜在意味解析[1]，潜在的ディリクレ配分（Latent Dirichlet Allocation, LDA）を用いた方法が提案されている[2]．ベクトル空間法は，文書を単語の重みのベクトルで表し，ベクトルの類似度で仕様間の関連性を計算する．単語の重みとしては，文書に含まれる単語の出現頻度を示す TF と単語を含む文書の逆出現頻度を示す IDF という 2 つの指標を用いた TF-IDF（Term Frequency-Inverse Document Frequency）が用いられる[3]．潜在意味解析は，文書から概念の集合を生成することで仕様間の関連性を計算する方法である．LDA は，潜在意味解析の改良であり，文書が潜在的トピックの混合として構成され，トピックは単語の分布で特徴付けられると考え，仕様間の関連性を推定する．

これら従来法では，仕様を単語の集合として扱い，単語順を無視する．しかし，仕様間の整合性判定では，単語がどう構成されているかを示す単語順も重要である．例えば，「顧客から営業へ…」と「営業から顧客へ…」という 2 つの仕様は，含まれる単語が同じだが，仕様間の整合性という点では区別する必要がある．また，整合する仕様に共通に含まれるコンポーネント名や機能名などは，複数単語が連続した複合語で構成されがちであるため，単語が連続して一致する場合と離れて一致する場合では，連続一致する場合を重視したい．しかし，従来法では，単語の連続一致と離れた一致とを区別することができない．

そのため，我々は，単語順も考慮されるコルモゴロフ複雑性に基づく仕様間整合性判定法を提案してきた[4][5]．コルモゴロフ複雑性は，情報のランダム性の指標であり[6]，Xerox の業務分類における XML 形式データでは，有効性が確認されている[7]．

本稿では，共通部分列を用いたコルモゴロフ複雑性の新規概算法を提案する．提案法は，異なる工程の 2 つのプロジェクトに適用し，整合性判定で重要なランク上位の判定精度を用いて評価した．その結果，提案法の判定精度は従来の概算法以上だった．また，従来法である LDA やベクトル空間法と比較したところ，いずれのプロジェク

† 日本電気(株) サービスプラットフォーム研究所
Service Platform Laboratories, NEC Corporation

トでも，提案法の判定精度が最も高かった．

本稿では，まず 2 章で，提案した仕様間整合性判定法を説明する．次に 3 章でコルモゴロフ複雑性と新規概算法について説明する．さらに 4 章で適用した 2 種のプロジェクト文書と適用結果を述べ，5 章で考察を述べ，6 章でまとめを述べる．

2. 仕様間の整合性判定法

仕様間の整合性判定法は，文書からの仕様の抽出，語句の統一，仕様間の類似度計算と整合性判定を自動で行う．

2.1 文書からの仕様の抽出

まず，RFP（提案依頼書）と提案書のような判定対象の 2 つの文書からテキストを抽出する．本稿では，文書のテキスト，テキスト主体の表，サンプルスクリプトは，テキストとして抽出し，表紙や改版履歴，目次などや図，記号主体の表（○×表など）は，抽出せず分析対象外とした．そして，抽出テキストの本文は 1 行ごとに 1 件の仕様とする．章・節・表のタイトル行は，単独では仕様とならないが文脈として重要なため，以降の本文に追加する．章・節・表の番号は，異なる文書では一致しないために削除する．

入力文書の例を図 1 に示す．図 1（A）は文書のテキストの記述例であり，1 件の仕様として「システムの要件．機能要件．外部インターフェース要件．データ形式．外部とのインターフェースについては，TXT 形式で実施する．必要なら，CSV 形式とする．」が抽出される．図 1（B）はテキスト主体の表の記述例であり，仕様として「ファイル仕様．…入力データファイル項目一覧．■■■■整数 1 以上■■者の ID」が抽出される．ここで，入力データファイルに関するという文脈情報は表のタイトル行にのみ存在しており，この例はタイトル行の本文への追加が必要とされる例である．

（A）

2. システムの要件
2.1. 機能要件
：
2.1.1. 外部インターフェース要件
(1) データ形式
外部とのインターフェースについては，TXT形式で実施する．必要なら，CSV形式とする．
(2) 開発言語

（B）

4. ファイル仕様
・・・

表4-2 入力データファイル項目一覧

データ名	データ型	有効範囲	説明
■■■■	整数	1以上	■■者のID

図 1　文書の例　（A）テキスト形式，（B）表形式

入力ファイル,入力データ,入力データファイル
■■出力ファイル,■■結果ファイル
制御,コントロール
パラメタ,引数

図 2　登録した同義語句の例

なお，本稿では 1 行を 1 件の仕様としたが， 1 行に複数仕様が含まれたり複数行が 1 件の仕様となったりする文書の場合には，句点や行頭・行末の文字（箇条書きの記号，行末での単語分割など）を用いて 1 件ごとの仕様へと分割して抽出する．

2.2 語句の統一

次に，仕様の語句を統一する．語句の統一では，予め辞書に同義語句とその代表語句を登録しておき，MeCab[8]を用いた形態素解析によって語句を抽出し，抽出語句が同義語句と一致するならその代表語句に変換する．同義語句は，一般的な同義語や冗長表現，業務用同義語である．一般的な同義語と冗長表現は，多くの文書から自動推定された同義語辞書 WordNet[9]と，これまでの研究[4][5]などで蓄積した辞書を用いた．業務用同義語は，業務上の特別な同義語句であり，例えば，「入力データファイル」，「入力データ」の代表語句を「入力ファイル」と登録した．また，RFP に「RFP に基づき」, 提案書に「提案書に基づき」という記述がある場合, これらは同義とみなせる．そこで,「RFP」と「提案書」の代表語句を「仕様書」と登録した．冗長表現は，同じ意味の簡略表現をもつ冗長な表現である．ただし,「本システム」,「当該機能」など文脈から意味を解釈すべき単語は，辞書に登録できず，語句統一できないままである．

本稿の適用実験では，新たに約 50 件の語句を登録して用いた．登録した語句の例を図 2 に示す．図で，各行の先頭が代表語句，それ以降が同義語句を示す．なお，適用した 2 つのプロジェクトは，工程も開発内容も異なるが，同じ同義語句を用いた．

前述した第一の例は， WordNet の同義語に基づいて「形式」が「形態」に変換され，冗長表現「とする．」が簡略表現「．」に統一され,「システムの要件．機能要件．外部インターフェース要件．データ形態．外部とのインターフェースでは，TXT 形態で．必要なら，CSV 形式．」と変換される．また，第二の例では，図 2 の同義語句に基づいて「入力データファイル」が「入力ファイル」に変換され,「ファイル仕様．…入力ファイル項目一覧．■■■■整数 1 以上■■者の ID」となる．

2.3 仕様間の類似度計算と整合性判定

語句統一の後，2 つの文書の仕様間の類似度を次章で述べる方法で計算する．そして，予め高い閾値を設定し，類似度が閾値以上の仕様ペアを整合と自動判定する．そして，他の仕様と整合と判定されていない仕様は，不整合候補と検出する．検出した不整合候補に対しては，閾値を下げながら不整合候補との類似度が閾値以上の仕様を関連仕様として検出する. なお, 不整合候補と関連仕様との整合性は人手で確認する．

3. コルモゴロフ複雑性と新規概算法

本稿では，まず，仕様間の類似度の計算方法として，コルモゴロフ複雑性について説明し，次に，本稿で提案する新規概算法について述べる．

3.1 コルモゴロフ複雑性

コルモゴロフ複雑性は，情報量に基づいた文字列のランダム性の指標であり，文字列 x のコルモゴロフ複雑性 K(x)は，万能計算機（万能チューリングマシンなど）で x を出力することができる最も短いプログラムの長さである[6]．直感的には，K(x)は，あるアルゴリズムで x を生成するのに必要な最小の情報量である．

コルモゴロフ複雑性に基づいて， Li らは，以下の正規化情報距離 NID（Normalized Information Distance）を提案した[10].

$$\frac{\max\{K(x\mid y)+K(y\mid x)\}}{\max\{K(x),K(y)\}}$$

しかし，任意の文字列 x を入力として，そのコルモゴロフ複雑性 K(x)を出力する計算可能なプログラムは存在せず，K(x)は一般的には計算不能である．そのため，実問題への適用では，K(x)を概算する必要がある．コルモゴロフ複雑性の概算のため，Cilibrasi と Vitányi は, 圧縮を用いることを提案した[11]. コルモゴロフ複雑性 K(x)は，文字列 x の最高の圧縮と考えられるためである．ここで，圧縮により情報が欠落することを防ぐため，圧縮アルゴリズムとしては，圧縮前の文字列と，圧縮・展開の処理を経た文字列が完全に等しくなる可逆圧縮を用いる．NID からの導出で提案された正規化圧縮距離 NCD（Normalized Compression Distance）は以下の式である[11].

$$\mathrm{NCD}(x,y)=\frac{C(x\cdot y)-\min\{C(x),C(y)\}}{\max\{C(x),C(y)\}}$$

ここで，C(x)，C(y)は，それぞれ，文字列 x，y の圧縮列の長さ，C(x・y)は，文字列 x と y を連結させた圧縮列の長さである．NCD の計算式に C(y・x)が含まれないのは，可逆圧縮である zgip や bzip2 などでは, 対称性が定義や経験から成り立つためである．NCD は, 値が小さいほど 2 つのデータが類似していることを示す 0 以上 1+ε の値である．ε は，圧縮の不完全性に起因する値であり，理想的には，NCD は 1 以下の値となる．NCD は，値が小さいほど 2 つの文字列が類似することを示しており，類似度は，(1-NCD)*100 で計算する．したがって，類似度は概ね 0 から 100 までの実数である．

3.2 共通部分列を用いた新規概算法

NCD の計算では，まずテキストを 2 進記号列に変換してから圧縮するため，異なる言葉が類似の 2 進記号列に変換された場合に分析精度が低いという問題がある．例えば，「あ」が「1010010010100010」，「い」が「1010010010100100」という 2 進記号列に変換される場合，「あ」と「い」は，元の自然言語テキストでは全く異なる言葉であるにも関わらず，先頭から 13 文字が同一であるため，NCD ではある程度類似だと計算される．このように NCD では，元のテキストに同一の文字がなくても，類似度が高くなるという問題があった．

この問題を解決するため，本稿では，共通部分列を用いた新規概算法を提案する．提案法では, 2 つの仕様 A, B 間に文字列長 l の共通部分列がm件以上ある場合に NCD を用いて類似度を計算し，m件未満の場合には類似度を 0 とする．ただし，共通部分列は，ひらがななど以外という条件を満たし，共通部分列は重複なしに計算する．ここで，共通部分列数mが 0 の場合が従来の NCD の類似度計算と同一である．NCD の計算と異なり，共通部分列の計算では，元の自然言語テキストを用いるため，「あ」と「い」とが全く異なると区別することができる．

共通部分列の計算例を図 3 に示す．図は，「認証システムは，登録者かどうかの認証を行う.」と「認証システム．1 台のシングル構成.」という 2 つの仕様の例である．まず, 1 番目の仕様からは, ひらがなや句読点などの記号を除き, 図に示すように,「認証」,「証シ」,・・・「録者」という部分列を抽出する．その際，一度抽出した部分列「認証」は重複抽出しない. 次に, これらの部分列が 2 番目の仕様に含まれるかを調べる．その結果，共通部分列数としては，図に示すように「認証」,「証シ」,・・・,「テム」の 5 件となる．なお，共通部分列を用いた概算法では，章・節のタイトルが長いとそのタイトルだけで共通部分列数が大きくなる．これを防ぐため，1 層目（「1.」など）から 4 層目（「1.1.1.1」など）までの章・節のタイトルは除いて共通部分列を計算した．

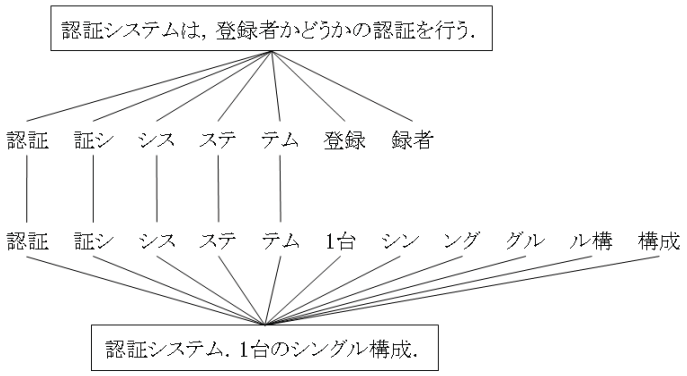

図 3　共通部分列の計算例

コルモゴロフ複雑性の従来の概算法は, パラメタフリーであるが[12], 提案法では, 共通部分列長 l と共通部分列数 m を予め設定する必要がある．共通部分列の抽出としては，要求工学において，要求文書から漏れなく情報を把握するための共通フレーズ抽出法が提案されている[13]．そこでは，英語テキストの分析として共通部分列長として 5 が選択されている．しかし，本稿で分析する日本語テキストでは，英語と異なり短い文字数で意味がある．そこで，意味を構成する最小単位となる単語の文字数を WordNet[9]で調べたところ，文字数が 2 の単語が最も多かった．そのため，共通部分列長 l として 2 を選択した．共通部分列数は，整合する仕様間がどれくらい一致すると期待できるかによって，最適な値が異なる．仕様間の整合性判定では，整合する仕様間でコンポーネント名や機能名などが一致することが期待できる．そのため，共通部分列数 m は, コンポーネント名や機能名の最小の文字数から 1 を減じた値と設定すればよい．例えば，コンポーネント名が 5 文字で，2 つの仕様にそのコンポーネント名が含まれる場合，2 つの仕様の共通部分列数は 4 以上となるためである．一般に，仕様書では，コンポーネント名や機能名，ファイル名などの連続文字が区別のために一定以上の文字数となっており，仕様が整合するのに共通部分列が小さい場合はほとんどないと考えられる. 実際, 本稿で用いた 2 種のプロジェクト文書を調べたところ，整合する仕様間で共通部分列数が小さい場合はほとんどなく， m を 2，4，6，8 と変化させながら予備実験を行うと， 4 または 6 のときの判定精度が高かった．そのため，本稿では，共通部分列数 m として小さい方の 4 を用いた．

4. 適用実験

提案法を評価するため，提案法および従来法を異なる工程の 2 種のプロジェクト文書に適用し，自動判定結果と人手で調べた正解とを比較した．

4.1 適用プロジェクト文書

第一のプロジェクト文書（第一のデータ）は，RFP と提案書である．提案依頼および提案時期は, 2009 年度下期である. 文書形式は, RFP, 提案書とも MS Word であり, RFP は 8 ページ，提案書は 14 ページであった．開発内容は，検索機能や表示機能などの GUI 画面の開発がメインである．RFP の記述例は図 1（A）であり，提案書も類似文書である．第一のデータから仕様を自動抽出した結果，RFP から 101 件の仕様，提案書から 109 件の仕様が抽出された．抽出した仕様に対し，約 1 万の仕様ペアの整合性を人手で調べたところ，正解は 94 件であった．

第二のプロジェクト文書（第二のデータ）は，機能仕様書と製造仕様書である．開発時期は 2009 年 6 月から 9 月，開発規模は 3.0 人月である．開発方法は，ウォータフォール型だが，以降も継続開発されている．文書形式は，両方とも MS Word であり，機能仕様書 26 ページ，製造仕様書 9 ページであった．開発内容は，社内利用のためのツール開発であり，2 つのコンポーネントおよびそれらコンポーネントを統合する GUI の開発である．機能仕様書には，処理の流れやファイル仕様，画面仕様が記述されており，記述例は，図 1（B）である．第二のデータから仕様を自動抽出した結果, 機能仕様書から 171 件の仕様，製造仕様書から 125 件の仕様が抽出された．抽出した仕様に対し, 約 2 万の仕様ペアの整合性を人手で調べたところ, 正解は 80 件であった.

4.2 評価方法

評価には，以下のような再現率（recall）と適合率（precision）を用いた.

再現率＝検出された正解件数／全正解件数

適合率＝検出された正解件数／検出された総件数

再現率 1 は，すべての整合する仕様ペアが検出されたことを示し，適合率 1 は，整合と推定された仕様ペアがすべて誤りなく整合していることを示す．類似度の閾値を下げると，検出される正解が増えるが，同時に整合しない仕様ペアである不正解の検出も増える．よって，再現率と適合率は，トレードオフの関係にある．そのため，評価には，x 軸を再現率，y 軸を適合率とした再現率・適合率曲線を用いた.

要求仕様間の整合性を自動で確認するためには，類似度の高い仕様ペアが真に整合している必要がある．理想的には，正解総数を n とすると，類似度の高い方から順にランク上位 n 件がすべて正解であることが望ましい．しかし，実際には，ランク上位にも不正解が混在する．本稿の目的である整合性判定では，閾値以上のランク上位を整合と判定するため，適合率が高い（不正解が少ない）部分での再現率が高いことが必要である．そのため，同一適合率に対する再現率を判定精度として評価した.

4.3 適用方法

提案法および従来の概算法（NCD）は自作プログラムを用いて計算した.

ベクトル空間法では，仕様（テキスト）d_i は，各単語の重みから構成されるベクトル（w_{i1}，w_{i2}，・・・，w_{iM}）として表現する．そして，仕様 i での単語 m の重みw_{im}（TF-IDF）を，自作プログラムを用いて以下の式で計算した.

$$w_{im} = \frac{n_{im}}{\sum_{m=1}^{M} n_{im}} \left(\log(\frac{N}{n_m}) + 1) \right)$$

ここで，n_{im} は仕様 i での単語 m の出現頻度，M は全単語数，N は総仕様数，n_m は単語 m を含む仕様数である．ベクトル空間法では，用いる形態素の限定などの調整が可能である．しかし，精度の向上が見られなかったため，調整は行わなかった.

潜在的ディリクレ配分（LDA）[14]は，統計解析ソフト R のパッケージ lda を用いて計算した．LDA は，トピック数という設定パラメタによって類似度の計算結果が変化する．そのため，トピック数 50 から 50 刻みで 300 まで変化させ，計算結果を比較し，ランク上位の判定精度が最も高いトピック数の場合を選択した．その結果，第一のデータではトピック数 150，第二のデータではトピック数 200 を選択した．なお，

正解情報なしにより細かくトピック数や設定パラメタを最適化することは困難なため，必要以上にパラメタを最適化はしなかった．

LDA やベクトル空間法（TF-ID）での仕様間の類似度は，計算した単語の重みを用いてコサイン類似度を用いて計算した．

なお，評価では，適用するすべての方法に対し，「2.1 仕様の抽出」と「2.2 語句の統一」は行い，「2.3 仕様間の類似度計算」での整合性判定の精度を比較した．

4.4 適用結果

提案法，従来の NCD，TF-IDF を用いたベクトル空間法（以降，TF-IDF と記す），LDA を用いた場合の再現率-適合率曲線を図 4 に示す．図 4（A）は，第一のデータ，図 4（B）は第二のデータの再現率-適合率曲線である．図で，提案法を実線，従来の NCD を破線，TF-IDF を点線，LDA を鎖線で示す．図に示すように，特に第一のデータで，提案法は従来の NCD より判定精度（同一適合率に対する再現率）が高かった．また， いずれのデータでも，LDA を用いた方法は，他の方法に比べ，判定精度が大幅に低かった．提案法と TF-IDF との比較では，判定精度は，第一のデータの再現率 0.4 付近で若干の逆転はあるものの，いずれのデータでも適合率が高いランク上位の判定精度は提案法の方が高かった．

判定精度（同一適合率に対する再現率）を表 1 に示す．表に示すように，いずれの適合率でも提案法の再現率は従来の NCD 以上であり，LDA の再現率は低く，適合率 0.80 以上のランク上位の再現率では，提案法が TF-IDF より高かった．例えば，適合率 0.90 での再現率は，第一のデータでは提案法が 0.81 と最も高く，次いで NCD が 0.67，TF-IDF が 0.59，LDA が最も低く 0.01 であった．また，第二のデータでは，提案法と NCD が再現率 0.39 と高く，次いで TF-IDF が 0.05， LDA が最も低く 0.02 であった．

5. 考察

提案法は，従来法 NCD 以上の判定精度となった．提案法では，元の自然言語テキストでは全く異なる言葉であるにも関わらず，類似度が高くなることが防止され，判定精度が向上したと考えられる．

従来法のうち，TF-IDF は，第一のデータでの再現率 0.4 付近を除き，提案法よりランク上位の判定精度が低かった．特に第二のデータでは，大幅に判定精度が低かった．文書の特徴を調べてみると，第一のデータは企画・要求獲得工程の文書で，機能の概要を仕様ごとに異なる単語を用いて説明する場合が多かった．一方，第二のデータは設計工程の文書で，異なる仕様でも同一の単語を用いて詳細に説明する場合が多かった．よって，第二のデータでは，仕様間の整合性に単語の出現だけでなく単語順が重要であり，単語順を考慮できない TF-IDF の精度が顕著に低くなったと考えられる．

全体的に，LDA の整合性検出精度は極めて低かった．LDA の最適なトピック数は，

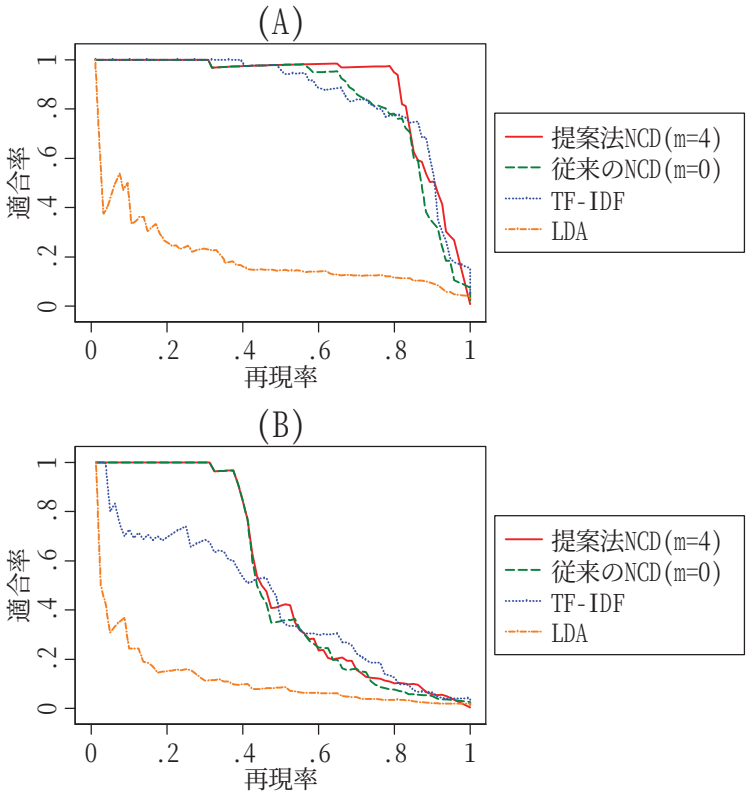

図 4 再現率-適合率曲線．（A）第一のデータ，（B）第二のデータ．

表 1 判定精度（同一適合率に対する再現率）

適合率	第一のデータ				第二のデータ			
	提案法	NCD	TF-IDF	LDA	提案法	NCD	TF-IDF	LDA
0.90	**0.81**	0.67	0.59	0.01	**0.39**	**0.39**	0.05	0.02
0.80	**0.83**	0.78	0.77	0.02	**0.41**	**0.41**	0.21	0.02
0.70	0.84	0.84	**0.87**	0.02	**0.42**	**0.42**	0.18	0.02
0.60	0.86	0.85	**0.89**	0.05	**0.43**	0.42	0.36	0.02
0.50	**0.89**	0.87	**0.89**	0.05	**0.45**	0.44	0.41	0.03

整合した仕様を同一とみなした場合の実質仕様数に近く，トピック数に対するデータ数が 2 倍程度と少なかった．LDA では，単語順が考慮できないだけでなく，トピック数に対するデータ数が少なすぎて計算の収束が困難だったことも精度が低くなった原因と考えられる．

提案法を用いて不整合が検出できるかを調べるため，データの実際の不整合を人手で調べた．第一のデータでは，RFP に対して提案内容に変更はあったが，矛盾や漏れなどの不整合はなかった．一方，第二のデータでは，製造仕様書にあるが機能仕様書にない漏れた仕様があった．また，機能仕様書では「正の整数値」，製造仕様書では「0 以上の整数値」とパラメタ値が不一致の仕様もあった．そこで，第二のデータで，経験上十分高いと考えられる閾値 55 を用いて不整合候補を検出したところ，漏れた仕様もパラメタ値不一致の仕様も不整合候補として検出することができた．さらに，閾値を１つずつ下げながら関連仕様を検出したところ，漏れた仕様に対しては，閾値 39 で関連仕様 2 件が検出され，人手で確認しても漏れた仕様と関連仕様は当然ながら整合していなかった．漏れた仕様は，第 2 回会議からの追加仕様であり，以降の開発でエラー処理の考慮漏れが生じていた．したがって，提案法を用いて漏れを検出することで，エラー処理の考慮漏れを防止できた可能性がある．また，パラメタ不一致の仕様（「正の整数値」）に対しては，提案法を用いないと，製造仕様書の 109 件の仕様との整合性を確認する必要があるが，提案法を用いると閾値 34 で関連仕様 1 件（「0 以上の整数値」）が検出され，人手でその 1 件の仕様との整合性を確認することでパラメタ値の不一致を確認できた．このように提案法は，不整合候補を検出し，人手での整合性確認を支援することができる．

整合性判定法では，「本システム」，「当該機能」など文脈依存単語の意味が解釈できない．文脈依存の仕様は，整合していても不整合候補として検出され，関連仕様を調べることになる．ただし，文脈依存の仕様は，人手のキーワード検索でも検索できず，人にとっても誤解が生じがちで，同様に問題である．人にとっても整合性判定法にとっても適切な仕様書の記述形式をどうすればよいかは，今後の研究課題である．また，現状の整合性判定法は，表の○×などの記号や図の情報も解釈できない．提案法の適用範囲を広げる方法を考えることも，今後の研究課題である．

別の従来法として，連続する n 個の単語（または文字）を用いる n-gram がある．しかし，n-gram では，すべての計算で同一の n を用いる必要があり，計算箇所ごとに n を最適化できない．例えば，「顧客から営業へ…」と「営業から顧客へ…」を区別するには単語数 2 以上が必要だが，単語数 2 以上では「顧客ファイル」と「顧客用ファイル」を類似とみなすことができない．これに対し，提案法は，共通部分列を用いた計算では 2 文字ごとの一致を見て無関係な仕様ペアをフィルタリングするものの，主な計算である NCD は，計算する箇所ごとに n に相当する連続文字数が変化して圧縮列の長さを計算するため，n を最適化する必要がないパラメタフリーな方法である．

6. おわりに

本稿では，コルモゴロフ複雑性の新規な概算法を提案した．提案法は，2 つのプロジェクトにおけるランク上位の判定精度を用いて評価した．その結果，提案法の判定精度は従来の概算法以上だった．また，従来法である LDA やベクトル空間法と比較したところ，提案法の判定精度が最も高かった．また，提案法を用いて検出した不整合候補とその関連仕様から，実際に仕様の漏れやパラメタ不一致を見つけることができた．今後は，さらに適用範囲を拡大し精度向上する方法を考えるとともに，適した仕様書の記述形式を考え，多くのデータで性能比較を行っていきたい．

参考文献

1) Hayes, H. J. and Dekhtyar, A: A Framework for Comparing Requirements Tracing Experiments. International Journal of Software Engineering and Knowledge Engineering, Vol.15, No.5, pp. 751-782 (2005).

2) Asuncion, H. U. et al.: Software Traceability with Topic Modeling, Proc. of the 32th ACM/IEEE Int. Conf. on Software Engineering (ICSE), Vol. 1, pp.95-104 (2010).

3) Robertson, S.: Understanding Inverse Document Frequency: On theoretical arguments for IDF, J. of Documentation, Vol. 60, No. 5, pp.503-520 (2004).

4) 藤原由希子，五藤智久，谷幹也: コルモゴロフ複雑性に基づく文書間の整合性確認，ソフトウェアエンジニアリングシンポジウム(SES) (2009).

5) 藤原由希子，五藤智久，谷幹也: トレーサビリティリンク自動生成に関する研究，電子情報通信学会総合大会, D-13-2 (2010).

6) Bennett, C.H., et al: Information Distance, IEEE Trans .Information Theory, Vol.44, No.4, pp.1407-1423 (1998).

7) Keogh, E., et al: Compression-based data mining of sequential data, Data Mining and Knowledge Discovery, Vol. 14, No. 1, pp.99-129 (2007).

8) MeCab: Yet Another Part-of-Speech and Morphological Analyzer, http://mecab.sourceforge.net/

9) Bond, F., et al: Enhancing the Japanese WordNet, Proc. of 7th Workshop on Asian Language Resources, in conjunction with ACL-IJCNLP (2009).

10) Li M., et al: The Similarity Metric, IEEE Transactions on Information Theory, Vol.50, No.12, pp.3250-3264 (2004).

11) Cilibrasi R. and Vitányi P., Clustering by Compression, IEEE Transactions on Information Theory, Vol.51, No.4, pp.1523-1545 (2005).

12) Keogh E., et al: Towards Parameter-Free Data Mining, Proc. of the tenth ACM SIGKDD international conference on knowledge discovery and data mining, pp.206-215 (2004).

13) Goldin, L., Berry, D. M.: AbstFinder, A Prototype Natural Language Text Abstraction Finder for Use in Requirements Elicitation, J. of Automated Software Engineering, pp.375-412 (1997).

14) Blei, D. M. et al.:, Latent Dirichlet Allocation, J. of Machine Learning, Vol.3, pp.993-1022 (2003).