様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　３０　年　　６　月　１５　日現在

機関番号：　１２１０２
研究種目：　基盤研究(B)（一般）
研究期間：　2015～2017
課題番号：　１５Ｈ０２７４４

研究課題名（和文）統計的機械翻訳における翻訳・言語モデルの高速かつコンパクトな実装方法に関する研究

研究課題名（英文）A study on compact and fast translation and language models for statistical machine translation

研究代表者

山本　幹雄（YAMAMOTO, Mikio）

筑波大学・システム情報系・教授

研究者番号：４０２１０５６２

交付決定額（研究期間全体）：（直接経費）　　12,400,000円

研究成果の概要（和文）：ダブル配列言語モデル(Double-Array Language Model: DALM) は、TRIE のコンパクトかつ高速な実装であるダブル配列をベースとし、言語モデルの性質を最大限利用することでバランスのよい実装を実現している。しかし、DALMはモデルパラメータとインデックスを共通の配列に格納しているため、パラメータの量子化による圧縮効果が得られないという問題があった。本研究では、量子化圧縮を可能とするため、確率値を格納した配列を分離した効率的なデータ構造とアルゴリズムを提案した。特に、配列サイズを小さくするために提案した「部分転置ダブル配列」が主要な成果である。

研究成果の概要（英文）：Although DALM (Double-Array Language Model) is a fast and compact implementation of ngram language models, it fails to fully capitalize on quantization techniques for values of model parameters such as probabilities of ngrams, because of a structual limitation: it stores values and indexes in the common array. In this study, we developed some variants of DALM which have separate arrays for values and indexes and can exploit benefits of quantization. We investigated basic characteristics of DALM empirically and propose "partly transposed double-array" which is a key technique to educe the ability of DALMs with separate arrays.

研究分野：　情報工学

キーワード：　言語モデル　ダブル配列　部分転置ダブル配列　ランダム配置

１．研究開始当初の背景

近年、データからの学習に基づく機械翻訳技術の発展によって、多くの言語間の翻訳性能が飛躍的に高まった。特に近年の発展は、大量のトレーニングデータによる統計的モデルの精密化によるところが大きい。しかし、トレーニングデータの大規模化は統計的機械翻訳システムが必要とする資源の大規模化に直結しており、コンパクトかつ高速な実装技術が必要不可欠である。我々は、トライの実装技術の一つであり辞書探索用のデータ構造として優れた Double Array データ構造[1]を用いた確率的言語モデルの圧縮実装技術を開発し、コンパクト性と高速性のバランスで優れた性能を達成した。しかし、データ量の増加はますます進んでいるためさらなる高性能化が必要であった。

２．研究の目的

本研究では、統計的機械翻訳システムの主要な構成要素である言語モデルと翻訳モデルに対して、実行時の高速検索、高圧縮率、高精度を達成し、かつ圧縮したデータ構造を作成する点でも高速なモデル実装技術開発を目的とした。特に、これまで我々が開発してきた言語モデル実装手法の最大の欠点であった量子化圧縮手法が適用できない点の改良を主要な目的とした。

３．研究の方法

研究目的を達成するために、言語モデルと翻訳モデルについてそれぞれ以下のような手法で研究開発を行った。

（1）DALM 言語モデルの改良：我々が開発してきたダブル配列言語モデル（Double-Array Language Model: DALM）をベースに、圧縮性能を保持したまま量子化ができるような構造を検討した。まず、量子化ができるように言語モデルのパラメータ配列を分離する構造を検討した。しかし、従来の DALM がダブル配列インデックスの隙間にパラメータを詰め込む手法であったため単純に分離すると圧縮性能が悪くなる。このため、パラメータで埋めていた隙間を少なくするような手法を開発した（この部分が「研究成果」の節で述べる「部分転置ダブル配列」である）。さらにダブル配列の構築高速化のための「ランダム配置法」を検討・開発した。最終的に統計的機械翻訳システムに組み込んで性能を評価した。

（2）翻訳モデルの圧縮手法の開発：当初、翻訳モデルの索引構造を圧縮する予定であったが、検討の結果索引構造の圧縮よりも翻訳モデルのパラメータの圧縮（量子化）の方が効果が大きそうであることが判明した。このため、パラメータ圧縮手法を検討した。言語モデルの１次元のパラメータで効果があった bining 法を多次元（翻訳モデルのパラメータは多次元）に拡張するベクトル binning 法を検討・開発した。翻訳性能と圧縮率の変化で提案手法の性能を従来法と比較評価した。

４．研究成果

（1）DALM 言語モデルの改良

① ダブル配列[1]

図1に示すように、ダブル配列はTRIEの(疎)行列表現を2本の配列で表現したデータ構造であり、(疎) 行列表現の高速性を保ったままコンパクトにできる[1]。(疎) 行列表現は TRIE のノードを行に割り当て、ノードから子ノードへの遷移先(子ノードの行番号) を行で表現したものである。しかし、多くのノードは少数の遷移記号に対応する子ノードしか持たないため行列は疎な行列となり、言語モデルに応用した場合、非現実的なサイズとなってしまう。この無駄をなくすために、列方向にぶつからないようにすべての行をずらして1 本にまとめ（base配列と呼ぶ）、さらに誤遷移をなくすために親ノードの情報を別の1本に格納した配列（check 配列と呼ぶ）を加えたものがダブル配列である。

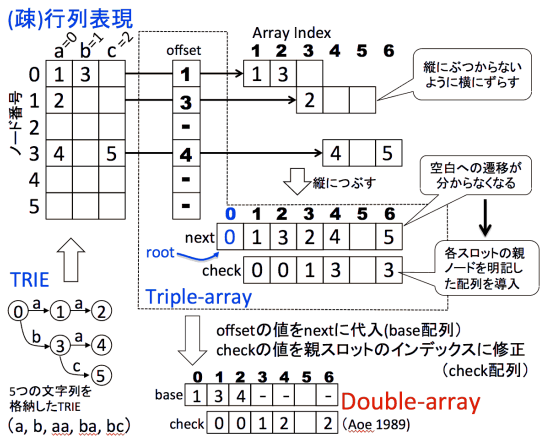

図1 ダブル配列

② ダブル配列を用いた言語モデルの実装

ngram言語モデルは、keyとしての単語(ID)の列を受け取り、その確率を返すことが基本であるが、key が存在しない場合は条件付確率の履歴部分のバックオフ係数と呼ばれる近似のための補正値を返す必要がある。このように、単語列によって1つあるいは2つのパラ

メータを格納する場合がある。ダブル配列でkeyを表現した場合、モデルパラメータをどこに格納するかが設計上の大きなポイントとなる。

DALMはダブル配列中の使われていない部分にパラメータを格納する。各ngramエントリに対して最大2つのパラメータがあり、ダブル配列はちょうど2本の配列があるため言語モデルとの相性がよい。この手法は、結果的にパラメータも含めて考えた場合の充填率が上がりコンパクトなモデルとなる。しかし、一方、ダブル配列は任意の場所を定数時間でアクセスできる配列の特性を活用しているため、各スロットのサイズを変更できない。このため、たとえ、モデルパラメータを量子化によってサイズを減らしても格納しているスロットを小さくできない。

モデルパラメータの量子化に対応するためには、2つの値を格納する配列をダブル配列と独立に確保すればよい。これを、従来のDALMと区別するためvsDALM(value-separated DALM)と呼ぶ。しかし、残念ながら、vsDALM の場合、ダブル配列の未使用領域が未使用のまま残り、かつ別配列が必要なため、トータルな充填率が下がってしまうという問題が生じる。図2にこれら2つの実装方法の長所と短所を図解する。

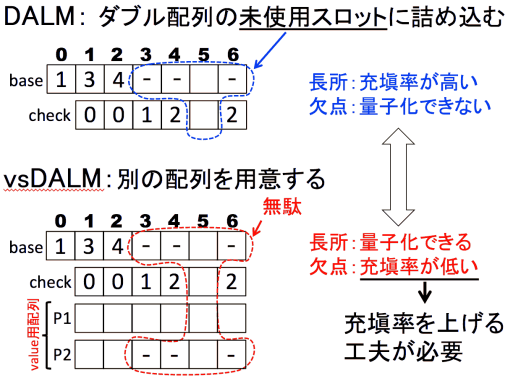

図2 ダブル配列を用いた言語モデル実装

③ Leaf-last 手法

vsDALM の充填率を上げるために、配列のあるindex 以降に値を保持する必要がない場合はその部分を削除できることに注目する。子ノードを持たないノードはbase値を保持する必要がないので、子ノードを持たないノードをダブル配列の後半に連続するように並べればbase 配列の後半部分を削除できる。また、ngram モデルの最高次数ngram は子ノードを持たずかつバックオフ係数を持つ必要がないので、ngramノードを配列の後半に配置すれば、バックオフ係数格納用の配列も後半を削除できる。この方法をleaf-last法と呼ぶ。配列の先頭から順に配置可能な場所を探すアルゴリズムを仮定すれば、子ノード群がすべて子を持たないノード(すなわち、孫ノードを持たないノード) を後から配置するようにすれば子ノードを持たないノードが最後に連続する可能性が高くなる。次のような順序でノード集合を配置していけばよい(それぞれの集合は子ノード数順で配置)。1. 孫ノードを持つノード集合、2.孫ノードを持たずかつ少なくとも1 つの子ノードがバックオフ係数を持つノード集合、3.残りのノード、の順である。

base 配列のある位置にbase値が格納されると、そこまでは削除できない。上記のアルゴリズムを素直に適用しても、残念ながらbase 値を持っているノードがかなり後半に出現してしまい削除できる割合は少ないことが実験的に明らかになった。この点を次節でやや詳細に分析し、部分転置ダブル配列の必要性を説明する。

④ Leaf-last法を用いたvsDALMの性質

前節で述べたleaf-last 法を用いても配列を削減できる割合が少ない原因を明らかにするための実験を行った。ノードを順に配置しているときにダブル配列は徐々に伸びていく。あるノードを配置するとき、それ以前に必要となった配列の範囲(先頭から最も後ろに配置された子ノードまで) 内に今配置しようとしているノードが配置された場合、配列は伸びない。しかし、その範囲よりもより後ろに配置する場合は、ダブル配列の範囲を広げる必要がある。この様子を可視化するために、横軸にノードの配置順番、縦軸にそのノードを配置した直後の配列の最も後ろのindex 値をプロットした。図3-(a)は1000万のngram エントリを持つ言語モデルに対してこのプロットを行った図である。配置順序はleaf-last法である。図3-(a)より、子ノード数の大きさの上位1000 個(0.02%)のノードによって、ダブル配列の大きさがほぼ決定されてしまい、その他の99.9%以上のノードは、上記1000個で決定された範囲の隙間に配置されていることが分かる。孫ノードを持つノード(約40%) 以降に伸びた部分が削除可能な部分であり、割合が非常に低いことが分かる。

このようなことが生じる理由は、子ノード数順に配置した場合の上位のノードは極端に子ノード数が多いため、大きなずらし幅が必要なためである。図3-(a)で用いたモデルデータは、ngramエントリ数が約1000万、子ノード

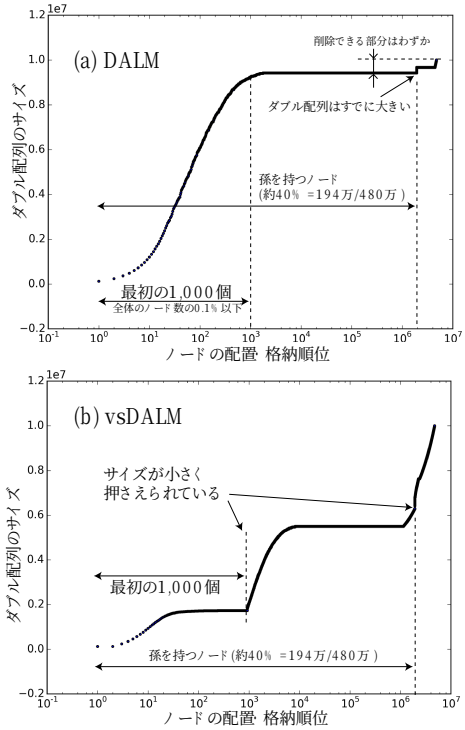

図3 ダブル配列の成長の様子

を持つノード数が約480万である。子ノード数上位1000個のノードが持つ子ノードの数は総計約100万となり、上位の0.02%(1000/480万)のノードの子ノードがノード全体の約1割(100万/1000万)を占め、上位のノードは平均で1000 個(100万/1000)の子ノードを持つ。語彙サイズは約10万であるため行列表現の 行は10万列からなり、この中に平均1000 個の子ノードの飛び先が格納されている。約1%(1000/10 万)の密度は小さく感じるかもしれないが、長さ10万の2つの配列をぶつからないように重ねて配置するには大きなずらし幅が必要となり、結果的に約100万の子ノードを配置するために約900万の長さの配列を必要としている。この約900万の中で使用されていないスロットは800万あり、残りの約9割の子ノードはこの隙間に配置されている。

⑤ 部分転置ダブル配列

前節の検討から、1%の密度の長さ約10万の行をぶつからないように配置するためには大きなずらし幅が必要であることが分かった。これを改善するために、「長さ10万」の方に着目する。例えば、密度1%は同じでも、行を短くすれば小さなずらし幅で重ねることができる。例えば、1%の密度の長さ1000の行(使用されているスロットは10個)は高い確率でずらさなくてもそのまま重ねることができるこ

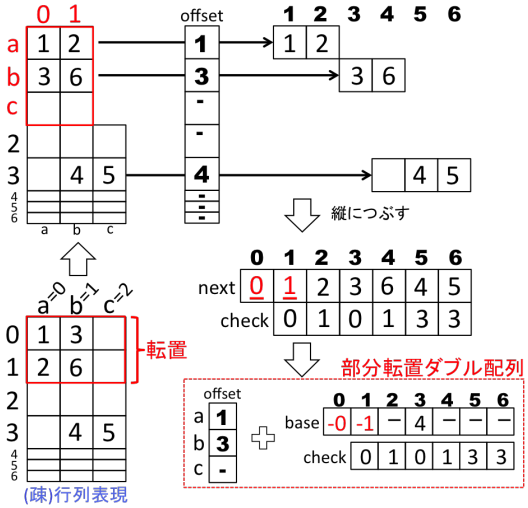

図4 部分転置ダブル配列の作成例

とは容易に想像がつく。このアイデアを用いて、ノード数の多い上位0.1%のノード(の子ノード)がダブル配列全域に分布しないようにできれば、leaf-last法が効く可能性がある。

上記のアイデアを次のように実現する。図1の(疎)行列表現を上から下に子ノード数が多い順にならべ替え、上位の1000行を転置してbase 配列を作るのである。元々1000×10万であった上位部分の行列が10万×1000の行列となる。数は増えてしまうが行の列は1000 となりずらし幅を小さくして重ねていくことができる。この手法を「部分転置ダブル配列」と呼ぶ。

部分転置ダブル配列を作成例を図4に示す。左下のTRIEの(疎) 行列表現の上2行を転置して、左上の部分転置表現とする。転置された部分は行が単語種類、列がノード番号と逆になっている。各行を横にずらしてぶつからないようにnext配列を決めるところは変化ないが、転置部分に関して単語種類毎にoffsetが決まっている点が異なる(offset値はすべて異なる必要がある)。通常のトリプル配列の遷移はnext 配列の値(遷移先のノード番号) からoffset 配列の値を引き出し(base 値)、その値に単語ID を加えることによって遷移先のindex が決定される。これに対して、転置部分の遷移は、next配列のノード番号に単語IDから決まるoffset値を加えて遷移先のindex が決定される。offsetの参照についてノード番号と単語ID がちょうど逆になっている関係である。ダブル配列への修正は次のようにする。next配列の転置部分に対応するノード番号は変更せず、転置部分ではないノードはoffset配列の値を代入してbase 配列を作る。この際、転置部分かそうでないかを

区別するために転置部分のbase値は-1を掛けて負の値とする。すなわち、base 配列値が負の場合はその値の絶対値に遷移単語IDで引けるoffset値とを加えた先を遷移先とする。check 配列についてはオリジナルのダブル配列と変わらず遷移元のindex 値を代入する。

図4の右下の部分転置ダブル配列の例では、ルートから'bb'で遷移する場合は次のようになる。まず、ルートのbase[0]=-0であるため、部分転置されたノードであることが分かる(-0の表現については便宜上である)。よって、遷移先は0+offset['b']=0+3=3となり、base[3]に遷移する。check[3]=0であるためチェックも大丈夫である。次にbase[3]=4で部分転置ノードでないことが分かるための遷移先はbase[3]+1('b'の単語ID)=5となる。check[5] = 3であるため、遷移に成功し、'bb'という単語列は存在していることが分かる。

部分転置ダブル配列は単語種類毎のoffset配列が付加的に必要となる。しかし、単語種類数は全体のノード数に比べるとわずかであり(1000万エントリの5gramで1%程度)、かつモデルが大規模になる毎に比率は低くなるため、深刻な問題とはならない。

⑥ 評価実験

提案手法を評価するために、実際の統計的機械翻訳を行った際の翻訳速度とメモリサイズ従来法と比較した。比較対象としては、従来のDALMの他、現在最も普及している言語モデル実装であるKenLM[3]を用いた。提案手法の転置ノード数は1万とした。また、KenLM trieと提案手法はそれぞれパラメータを確率値7bit、バックオフ係数を8bitに量子化したものも使用した。結果を図5に示す。この図より提案法(DALMpt) は従来のDALM とほぼ同等の速度、メモリ使用量で翻訳を行えていることがわかる。また、量子化によりさらにメモリ使用量を抑えることができている。

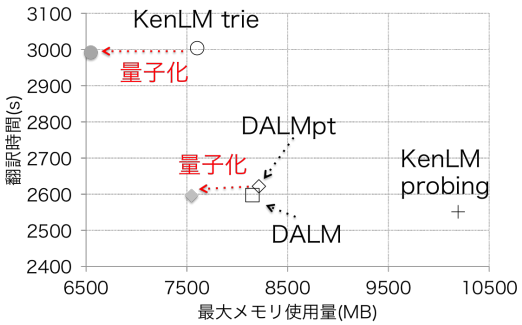

図5 速度と翻訳時間による提案手法の評価

⑦ 構築の高速化

DALM 構築において最も時間がかかる処理は、ノードの子ノード列が他の子ノード列と重ならないようなずらし幅を決める部分である。一般的にはノードをトライの深さ優先順に配置するがトライのサイズが大きくなると配列の隙間が増えてしまう。さらに、配置順序をランダム順にすることで高速化することができるが充填率をさらに下げてしまう。そこでランダム配置順と本研究で提案した部分転置ダブル配列手法を組み合わせることでコンパクトさを保ったまま構築の高速化ができる手法を開発した。

実験結果を図6に示す。グラフは横軸に構築時間(秒)、縦軸にngram数に対するcheck配列長の比率をとった(隙間なく詰まれば比率は1.0となり、隙間があればそれより大きくなる)。またleaf-last 法によりbase 配列がどれほど削れたかを評価するため各点横にngram数に対するbase配列長の比率を記した(base 配列は1.0 より小さくなり得る)。実験結果より深さ優先順配置と部分転置を組み合わせることにより2 つの配列を短くすることができた上に構築時間も短くなった。さらにランダム順配置と部分転置の組み合わせでさらなる高速化に成功しサイズも小さくなった。

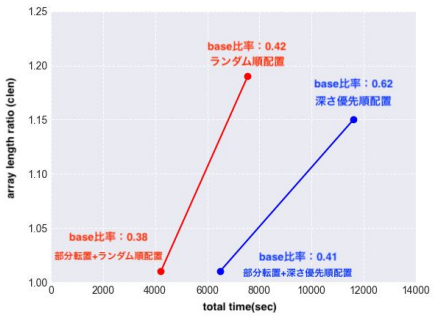

図6 部分転置とランダム配置による高速化

(2) 翻訳モデルのパラメータ圧縮

翻訳モデルは数千万から数十億の翻訳フレーズペアエントリごとに4つのパラメータを持っている。各パラメータは4bytes(32bits)のfoating point データ型で表現されることが多く、1つの翻訳フレーズ毎に16bytes のパラメータ用メモリが必要である。例えば、1億エントリを持つ翻訳モデルの場合16億bytes(1.6Gbytes) のパラメータ用のメモリが必要となる。

量子化はパラメータの値空間である実数空間を有限個のクラスに分け、それぞれのクラスの代表値で置き換えることにより必要メモリを圧縮する。Bertoldi ら[2] は、Lloyd 法

とbinning 法と呼ばれる2つの量子化法による量子化の実験を行い、統計的機械翻訳のモデルについては、ある程度大きな量子化を行ってもそれほど性能は悪化しないこと、特にbinning 法による量子化の性能が高いことを示した。

binning法はパラメータをその大きさで整列した後に、等分割するアルゴリズムで非常に簡単に計算できる。しかし、ベクトルになるとその大きさの比較が単純ではない（ベクトルの長さでの比較は無意味である）。提案手法では、主成分への正射影の値による再帰的な分割によりbinning法を次のようにベクトルの場合に拡張した。

1. パラメータ集合の第1主成分を求め、各パラメータの第1主成分への正射影の値によってパラメータ手法を2つに分割する。2つのクラスに属するパラメータ数が同数になるように分割する。
2. 以上の2分割を分割されたクラス毎に再帰的に行う（再帰の度に第1主成分は計算し直す）。再帰の深さをk段にすれば、2^k個のクラスに分割され、各クラスに属するパラメータ数は等しい。

図 7 に各次元毎に binning 法を適用した場合と提案手法であるベクトル binning 法で量子化した場合を比較した結果を示す。横軸は各パラメータが必要とする bit 数、縦軸が翻訳性能である。この図より、ベクトル binning 法が高い圧縮性能と高い翻訳性能を持つことが分かる。

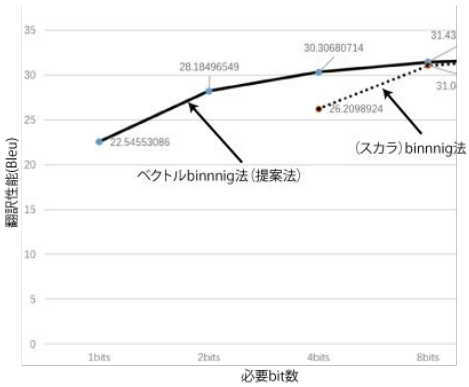

図 7 ベクトル binning 法の評価

＜引用文献＞

[1] Aoe, Jun-ichi. 1989. An efficient digital search algorithm by using a double-array structure, IEEE Transactionson Software Engineering, Vol. 15, No. 9, pp. 1066-1077.

[2] Federic,M.and N.Bertoldi. 2006. How many bits are needed to store probabilities for phrase- based translation? In StatMT06, pp.94-101.

[3] Heafield, Kenneth. 2011. KenLM: Faster and Smaller Language Model Queries. In Proceedings of the Sixth Workshop on SMT, pp.187-197.

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計 1 件）

① Jun-Ya Norimatsu, Makoto Yasuhara, Toru Tanaka and Mikio Yamamoto. 2016. A Fast and Compact Language Model Implementation Using Double-Array Structures. ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), Vol.15, Issue 4, 27 pages, June 2016.

〔学会発表〕（計 3 件）

① 石井瑛彦，芳賀駿平，竹中孝介，大隅賢二，山本幹雄．2018．細粒度並列処理によるダブル配列言語モデルの構築高速化．言語処理学会第24回年次大会，pp.216-219,March 2018.

② 竹中孝介，芳賀駿平，山本幹雄．2017．部分転置ダブル配列を用いたngram言語モデルの実装．言語処理学会第23回年次大会，P13-3．pp.663-666，March 2017.

③ 芳賀俊平，谷口正訓，山本幹雄．部分転置ダブルアレイを用いたngram言語モデルの検討．第30回人工知能学会全国大会．4 pages．June 2016.

〔その他〕

本研究で構築した言語モデル実装の公開：
https://github.com/nowlab/DALM/tree/ptDALM.v1.0.0

６．研究組織

(1)研究代表者
山本　幹雄（YAMAMOTO, Mikio）
筑波大学・システム情報系・教授
研究者番号：40210562

(2)研究分担者
乾　孝司（INUI, Takashi）
筑波大学・システム情報系・准教授
研究者番号： 60397031

(3)研究協力者
乗松　潤矢（NORIMATSU, Jun-ya）
谷口　正訓（TANIGUCHI, Masanori）
芳賀　俊平（HAGA, Shumpei）
大隅　賢二（OSUMI, Kenji）
竹中　孝介（TAKENAKA, Kousuke）
石井　瑛彦（ISHII, Akihiko）