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

令和　３ 年　 ６ 月　 ４ 日現在

機関番号：１２１０２
研究種目：基盤研究(C)（一般）
研究期間：2018～2020
課題番号：１８Ｋ１１４２３

研究課題名（和文）部分転置ダブル配列ngram言語モデルの構築高速化と深化

研究課題名（英文）Construction speedup and deepening of partially transpose double array ngram language models

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）　　3,300,000円

研究成果の概要（和文）：部分転置ダブル配列を用いたngram言語モデルの実装は、アクセス速度とモデルサイズの両面で優れているが、モデル(データ構造)の構築に非常に時間がかかるという欠点がある。本質的な困難性は数億から数十億にもなる子ノード配列(隙間がある)をお互いにぶつからないように1本の配列に配置する点にあり、相互依存が大きいため単純な並列化等の高速化が困難である。本研究では、部分転置ダブル配列の性質を深く検討し、複数の高速化手法によってモデル構築時間について高速化を実現すると同時により高い圧縮率を達成した。

研究成果の学術的意義や社会的意義
ngram言語モデルは音声認識や統計的機械翻訳技術の基盤技術であるため、本研究の成果によって高速かつコンパクトなngram言語モデルを短時間で作成できるようになった点に意義がある。また、より広い観点からは、ダブル配列はトライと呼ばれる一般的な辞書データ構造の実現方法の一つであり、本研究は巨大なデータに対するトライを高速かつコンパクトに実現できるという意味で巨大な辞書を必要とする広いアプリケーションに対しても有効である。

研究成果の概要（英文）：The implementation of ngram language models using the partially transposed double array is excellent in terms of both access speed and model size, but has the disadvantage that it takes a very long time to build the model (data structure). The essential difficulty lies in arranging hundreds of millions to billions of child node arrays (with gaps) in a single array so that they do not collide with each other. Due to the large interdependence, it is difficult to increase the speed by techniques such as simple parallelization. In this study, we deeply examined the properties of the partially transposed double array, realized a faster model construction by multiple acceleration methods, and at the same time achieved a higher compression rate.

研究分野：情報工学

キーワード：ngram言語モデル　ダブル配列　双方向配置　文字列マッチング　細粒度並列化

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９（共通）

１．研究開始当初の背景

近年の統計的な手法に基づく音声認識や機械翻訳技術の発展は訓練データの大規模化によって、高性能な統計的モデルを構成できるようになった点に大きく依存している。しかし、訓練データの大規模化は、統計モデルの大規模化に直結しており、最先端の研究・開発を行うためには大規模な計算環境が必要である。また、システムの研究開発コストだけでなく運用コストの増大にも直接影響しており、統計的手法の発展と普及に対する壁となっている。

このような状況で、2010 年代に入ってから、統計的手法の主要モデルの一つである ngram 言語モデルの圧縮技術の研究開発が世界的に活発化し、急速に実装技術が進化している。統計的機械翻訳の高精度化のための研究が主であるが、ngram 言語モデルは自然言語文の言語らしさを確率で評価する自然言語処理の基盤技術の一つであり応用範囲は広い。ngram 言語モデルは単語連鎖の条件付き確率を保持し、単語連鎖の組み合わせで文全体の確率を評価する。このため、ngram 言語モデルの実装は、基本的には単語連鎖をキーとして確率値を返す辞書・探索データ構造となる。その圧縮実装技術は、基本的な辞書・探索データ構造を ngram 言語モデルの特性に合わせて特化した技術である。実装技術の評価軸は圧縮率、探索速度、モデル構築速度の 3 つであり、トレードオフの関係にある。望まれる実装技術は、モデル構築速度が現実的な一定の時間以下で、かつ残り 2 つの評価軸(圧縮率と探索速度)に対してバランスよく高性能であることである。

２．研究の目的

ダブル配列と呼ばれる辞書データ構造を ngram 言語モデルに応用した実装技術がダブル配列言語モデルである。ダブル配列中の子ノードの配置は指数関数的な組み合わせが存在し、ダブル配列を最小とする配置を決定する問題は NP 困難である。このため、実際の構築では１つ１つの子ノード配列をできるだけ近くに配置していく貪欲法に基づくヒューリスティックスが使われる。この手法でも最悪計算量はトライのノード数の 2 乗のオーダとなるが、ダブル配列は主に単語辞書などのような数十万から数百万エントリという比較的小さなタスクに応用されてきたため大きな問題とはならなかった。しかし、言語モデルのようにノード数が数億から数十億に達することがある場合はその速度的な問題が健在化する。

これまでに我々はダブル配列言語モデルの欠点の一つである量子化対応ができないという問題点を解決した部分転置ダブル配列言語モデルを提案しているが、従来のダブル配列言語モデルと同様にモデル構築に時間がかかるという問題があった。本研究では、これまで我々が開発してきた部分転置ダブル配列の性質をより深く探求し、(I)複数の高速化のアイデアを組み合わせてモデル構築の高速化を達成するとともに、(II)モデル自体の圧縮率をより高めることを目的とした。

３．研究の方法

研究目的で述べた目的を達成するために、以下のようなアイデアを実装・評価するとともにその組み合わせによってより高速にモデル構築を行え、かつ圧縮率も高まる方法を検討した。

(1) 子ノード配置順のランダム化

最適な（サイズが小さい）モデルを構築するには子ノード数の多い順にダブル配列中の配置を決定することがよいが、構築速度が非常に遅くなってしまうことが知られている。逆に子ノード数によらずランダムな順で 1 本の配列に配置すると構築速度を高速化できるが、ダブル配列が長くなってしまう。しかし、我々の提案している部分転置と組み合わせることにより、特に子ノード数が多い数千のノードを小さく分解できるため、ランダム順配置による配列長の増加を抑えながら実行できる可能性がある。

(2) 細粒度の並列アルゴリズム

子ノード配列の配置位置は相互に依存するため細粒度の並列化(1つ1つの子ノード毎の並列化)は難しいが、次のような考えで並列アルゴリズムを構築できると考えた。子ノード列の配置手順は、(a)子ノードがぶつからない位置を探す手続きと、(b)見つかった位置を確保する手続きの 2 つに別れるが、ぶつからない位置を探す手続き(a)がほとんどの時間を使っている(10000:1 程度)。(a)について並列化すると投機的となるが(他のプロセスが同じ場所を探している可能性)、(b)の手続きで再確認することにより速度低下を避けながら細粒度の並列処理ができる。

(3) 双方向配置法

ダブル配列中でトライのノードの遷移先を決定するために次単語の単語 ID の数値に応じたオフセットの箇所をチェックする。これまで、オフセットは配列の後方方向のみ（順方向）の可能性しか考慮していなかったが、配列の前方方向（逆方向）へのオフセットの可能性を考慮することにより、より密な配列配置を決定できる。また、配置できる可能性が高まるため配置の失敗が少なくなりモデル構築の高速化も同時に期待できる。

(4) 文字列マッチング法の利用

子ノード配列をダブル配列に配置する際に無駄な候補をチェックしないことにより高速化できる。子ノード配列の配置は子ノード配列を文字列とみなした場合の文字列マッチングと似た処理と考えることができる。BM(Boyer-Moore)法のように suffix ベースの文字列マッチングアルゴリズムを用いれば配置に失敗したときの次候補へのスキップ幅を大きくでき、無駄なチェックを減らせる可能性がある。

４．研究成果

前節で述べたアイデアの実装・評価を行ったところ、いずれも構築高速化の効果を示すことができた。最初の 2 つのアイデア（配置順ランダム化と細粒度並列化）は、実用的な巨大なモデルを構築・評価する場合に必須の技術となったため、後半 2 つのアイデアの評価では最初の 2 つの技術を前提とした（高速化した手法を用いないと大規模データに対する実験が時間的制約で実施できなくなるため）。各アイデアの具体的な実装手法と評価結果は次の通りである。

(1) 子ノード配置順のランダム化

子ノード配列をダブル配列に配置する順番を(a)子ノードの数が多い順、(b)トライ木の深さ優先で探索した順、(c)ランダム順の 3 つで比較した。ngram 数は 1 億のエントリー数のデータを用いた。残念ながら(a)については数日では計算が終わらないため測定をあきらめた。(b)と(c)についての実験結果を図 1 に示す。横軸は構築にかかった時間(秒)、縦軸はモデルのサイズである(相対的な大きさ)。この結果から、部分転置ダブル配列についてランダム順で配置を行えば、サイズの悪化なしに約 1 時間でモデル構築を終えていることが分かる。ナイーブな手法では数日以上の時間がかかることから、部分転置法とランダム化を組み合わせることにより(a)に対して数十倍の高速化が可能であることが分かった。

(2) 細粒度の並列アルゴリズム

最近ではマルチコア CPU が安価に利用できるため高速化のための並列化は必然である。ダブル配列の構築では、子ノード配列をダブル配列に配置する時間がほとんどの構築時間を占めるため、単純には子ノード配列を各コアに割り当てて配置場所を探すことが考えられる。しかし、子ノード配列の配置はお互いに排他的であり、相互依存が強いため単純なアルゴリズムでは動作しない。我々は実際の配置の様子をシミュレートすることにより、配置場所を決定するために 1 つの子ノード配列の配置決定のために非常に多くの配置箇所のチェックをしており、その時間が支配的であることを明らかにした。また、実際のダブル配列の構築では高々数十万から数百万の子ノード列を長さ数億の配列上に配置場所を探すので競合はそれほど多くは発生しない。これらの事実に基づいて、それぞれのコアが独立にダブル配列に対する配置箇所を探索し、実際に配置する瞬間に競合をチェックするアルゴリズムを開発した。

実験結果を図 2 に示す。横軸は並列数（'m'の文字の右側の数字が並列数を示す)、縦軸は構築時間である。データは 2.4 億個の ngram、マシンは 8 コア(16 ハイパースレッド)の CPU を用いている。また、図の左半分は一つ前のアイデアであるランダム化を行わない場合、右半分はランダム化を行った場合である。ランダム化なしで 8 コアをフル(16 スレッド)に使った場合 9.6 倍、ランダム化した場合で 3.8 倍の高速化を達成している。ランダム化と並列化なしの場合から比べると、ランダム化と 8 コアフルの並列化を使った場合、65 倍の高速化を達成できている。

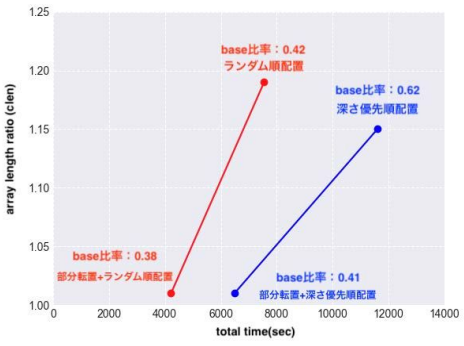

図 1 ランダム化と部分転置による高速化

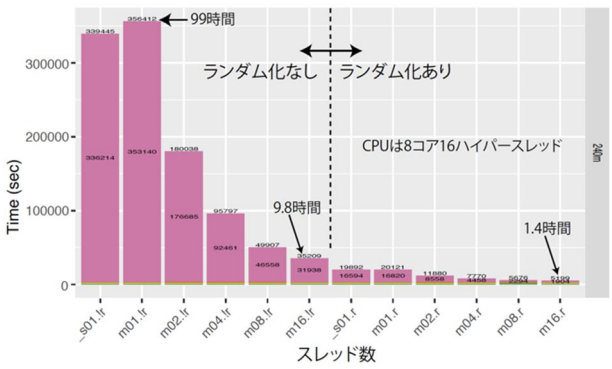

図 2 細粒度並列化とランダム化による高速化

(3) 双方向配置法

これまでのダブル配列の手法では、トライのあるノードから次の子ノードを見つけるために単語 ID の値だけ順方向(index が大きくなる方)のみにオフセットしていた。双方向配置法では、順方向側にターゲットの子ノードがない場合は逆方向も探索する。図 3 に従来の手法と双方向配置法の違いを示す。図 3 はすでにノードが配置されたダブル配列に遷移表のある行（子ノード配列）を配置する流れを示している。従来法の配置場所探索は各ノードの 1 つ目の子ノードをダブル配列上の空き要素がある場所まですらしその他の子ノードが全て順方向側に配置可能であ

るかを調べ、可能であれば配置し、すでに配置されたノードとぶつかってしまうのであれば 1 つ目の子ノードを次の空き要素までずらすという作業を配置できるまで繰り返す。双方向配置法では、各ノード毎に順方向側に配置できない場合は逆方向側のオフセットも試す。これにより、従来法では配置できなかった場所にも配置可能となり、配置場所を見つけやすくなる。これは、モデル構築速度の高速化とサイズ縮小に効果がある。

図 4 に実験結果を示す。横軸は構築時間、縦軸はダブル配列の相対的なサイズ(1.0 が最小の場合)を示す。図中の「分割」はダブル配列を分割して構築する場合の分割数である（これは従来からある高速化の手法の一つである)、「提案手法」は双方向配置法を意味する。データは 10 億の大規模 ngram データであり、いずれの実験もランダム化と細粒度並列化の手法（16 コアの CPU）を用いている。図 4 から分かる通り、双方向配置法により 3 割程度の高速化と同時にサイズの最適化が行えていることが分かる。

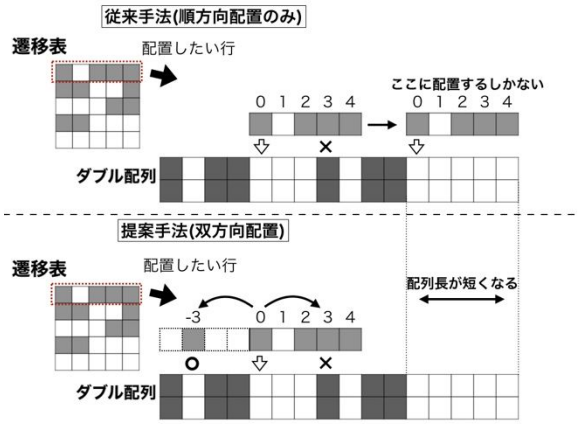

図 3 双方向配置法と従来法の違い

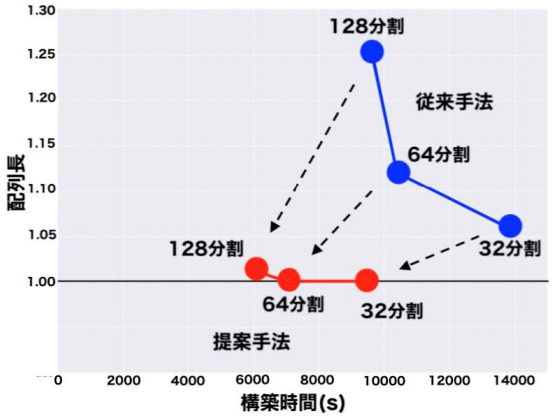

図 4 双方向配置法の効果

(4) 文字列マッチング法の利用

子ノード配列をダブル配列に配置する際、ある場所への配置に失敗すると次の配置場所へと移動しながら配置場所を探す。このとき、次の配置場所の候補として、子ノード配列のパターンから確実に配置できない候補を除く（スキップ）できると配置場所決定を高速化できる。これを実現するために、子ノード配列を文字列と考え、文字列マッチングのアルゴリズムを応用する。今回は文字列マッチングアルゴリズムの一つである Boyer-Moore 法からヒントを得た。図 5 に基本的なアイデアを図示する。配置しようとしている子ノード配列の競合が検知できた場所よりも左側が連続で埋まっている場合、その連続した長さは右にシフトしたとしても必ず競合が発生する。このため、連続している領域よりも多くシフトすることにすれば無駄なチェックを省略できる。また、できるだけシフトする量が多い場所から競合をチェックすることにすれば、大きなスキップが期待できる（「移動量順にチェック法」と呼ぶ)。

以上のアイデアを実装し、実験した結果が図 6 である。縦軸は構築時間の相対比（従来法を 1 とした場合)、横軸はデータ量（ngram 数）である。移動量順にチェック法を用いることによって、5 億の ngram に対して構築時間を 4 割短縮できていることが分かる。

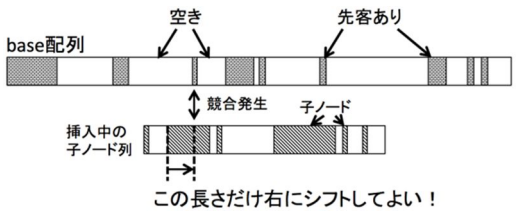

図 5 次の配置場所候補への安全なスキップ

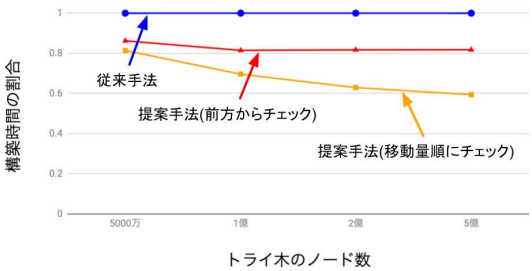

図 6 文字列マッチング法による構築高速化

以上 4 つの手法の組み合わせにより、数億～10 億のエントリーの ngram モデルデータであっても 1 ～2 時間程度という現実的な時間、かつより高い圧縮率でモデルを構築できるようになった。

５．主な発表論文等

〔雑誌論文〕　計0件

〔学会発表〕　計2件（うち招待講演　0件／うち国際学会　0件）

１．発表者名 仲村勇馬，山本幹雄
２．発表標題 文字列探索アルゴリズムを応用したダブル配列構築の高速化
３．学会等名 情報処理学会 第82回全国大会
４．発表年 2020年

１．発表者名 石井瑛彦，山本幹雄
２．発表標題 双方向配置によるコンパクトかつ高速なダブル配列言語モデル構築
３．学会等名 情報処理学会第81回全国大会
４．発表年 2019年

〔図書〕　計0件

〔産業財産権〕

〔その他〕

―

６．研究組織

	氏名 （ローマ字氏名） （研究者番号）	所属研究機関・部局・職 （機関番号）	備考

７．科研費を使用して開催した国際研究集会

〔国際研究集会〕　計0件

８．本研究に関連して実施した国際共同研究の実施状況

共同研究相手国	相手方研究機関