Vol.2013-DPS-154 No.55
Vol.2013-CSEC-60 No.55
2013/3/15

GPGPU を用いた近似 kNN グラフによる 大規模高次元ベクトルに対する高速な近似最近傍探索法の検討

松村 聖司[†1]　赤間 浩樹[†1]　山室 雅司[†2]

大規模な画像や音声等の各認識処理において，高次元ベクトルの近似最近傍探索の処理が行われている．近似最近傍探索では，探索精度と探索時間の関係がトレードオフの関係にある．そして近似最近傍探索法の 1 つに，kNN グラフを用いた探索がある．kNN グラフは，グラフの枝数を増やすことで，高精度かつ高速な探索が可能になる．しかし，kNN グラフの構築コストは大きく，構築コスト削減の為，近似 kNN グラフが考案されている．近似 kNN グラフの 1 つに NN-Decent という手法があり，枝数が少ないと，構築コストが小さくなるといった特徴がある．その為，本検討では，近似 kNN グラフ及び GPGPU を用いた高精度かつ高速な近似最近傍探索法について検討を行った．その結果，枝数の多い場合の kNN グラフと同等の精度及び処理時間で探索が可能であることを確認した．

Approximate Nearest Neighbor Search using GPGPU for Massive High-dimensional Vectors with kNN Graph

SEIJI MATSUMURA[†1]　HIROKI AKAMA[†1]　MASASHI YAMAMURO[†2]

An approximate nearest neighbor search for high-dimensional vectors are used for massive object, image and sound recognition. In an approximate nearest neighbor search, there is trade-off between searching accuracy and speed. One of approximate nearest neighbor search, there is kNN graph. kNN gprah is able to search with high speed and high accuracy when that graph has many branches. However, because it needs too much time to build kNN graph, approximate kNN graph is invented to reduce time building kNN graph. There is NN-Decent in one of approximate kNN graph. When there are few branches, NN-Decent doesn't spend much time to build. Therefore, we studied searching method using GPGPU with high accuracy and high speed when NN-Decent has a few branches. As a result, we confirmed that it was possible to search with high accuracy and high speed compared with kNN graph ,which has many branches.

1. はじめに

1.1 背景

物体や画像，音声等の多くの認識処理の中では，高次元ベクトルに対する最近傍探索処理 [1]が使われている．例えば物体認識処理では，SIFT [2]や SURF [3]等のアルゴリズムによって抽出された数十から数百次元の高次元の局所特徴量ベクトルに対して, 最近傍探索処理が行われている.

物体認識処理を用いて，身の回りの物を撮影し，それを Web 上の画像とマッチングさせるサービスを想定した場合, 距離計算対象となる局所特徴量ベクトルの数は莫大な量となり, 最近傍探索処理の高速化が求められている. 例えば, 画像共有サイト Flickr[4]では, 約 60 億枚もの画像が存在しており, 1 画像から 1000 個の特徴量ベクトルが抽出されたとすると，約 6 兆ものベクトル数になる．この場合，１秒以内に処理を達成する為には，最近傍探索処理を高速化する工夫が必要になる．この最近傍探索処理時間を短縮する為，木構造を用いた kd-tree[5]や，kd-tree に GPGPU を適用した手法[6], ハッシュ構造を用いた LSH[7], グラフ構造を用いた kNN グラフ[8]などの近似最近傍探索が使われている．近似最近傍探索では，探索精度と探索時間はトレードオフの関係にあり，一般的に探索精度を高く維持しようとすると，探索時間が長くなる．このベクトル数が多くなると，高い精度を維持した高速な探索が難しくなる．本検討では，これらの近似最近傍探索法の中で，高精度な探索が可能な手法の一つである kNN グラフに着目して, 近似最近傍探索の高速化について検討を行う．しかし，kNN グラフは厳密に作成すると，ノード数の二乗に比例したグラフ構築コストが生じる．そこで本検討では，グラフ構築コストの小さい近似 kNN グラフを用いることで、構築コストを抑えつつ，高精度かつ高速な近似近傍探索法について検討を行う.

1.2 課題

本検討の目的は，インデックス構築時間の小さな近似最近傍探索法を用いて，高精度かつ高速な探索法を実現することである．その為に，高い精度を維持した高速探索が可能な kNN グラフを用いた探索法に着目した．kNN グラフの課題は，グラフの構築時間が長いこと，枝数 k が小さい場合に，高い探索精度を維持した高速な探索が難しい点である. 構築時間を短縮する為に近似 kNN グラフが考案されている．この近似 kNN グラフの 1 つに NN-Decent [9]と呼ばれる手法がある. NN-Decent は, 枝数 k が小さい場合に, グラフの構築コストを小さくすることができる．従って本

†1 NTT ソフトウェアイノベーションセンタ
NTT Software Innovation Center
†2 エヌ・ティ・ティ・ソフトウェア株式会社
NTT Software Corporation

Vol.2013-DPS-154 No.55
Vol.2013-CSEC-60 No.55
2013/3/15

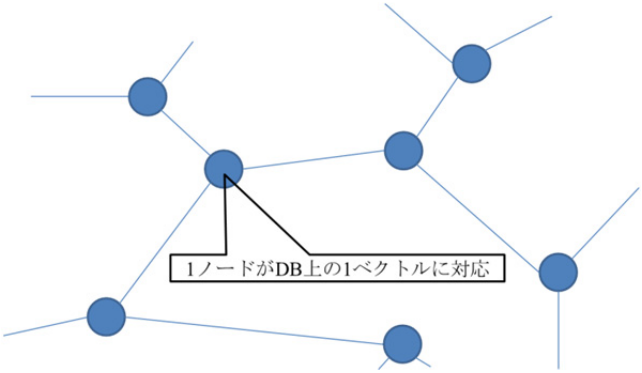

図 1　kNN グラフの例（k=3 の場合）
距離が近い 3 つのベクトルと枝を張る

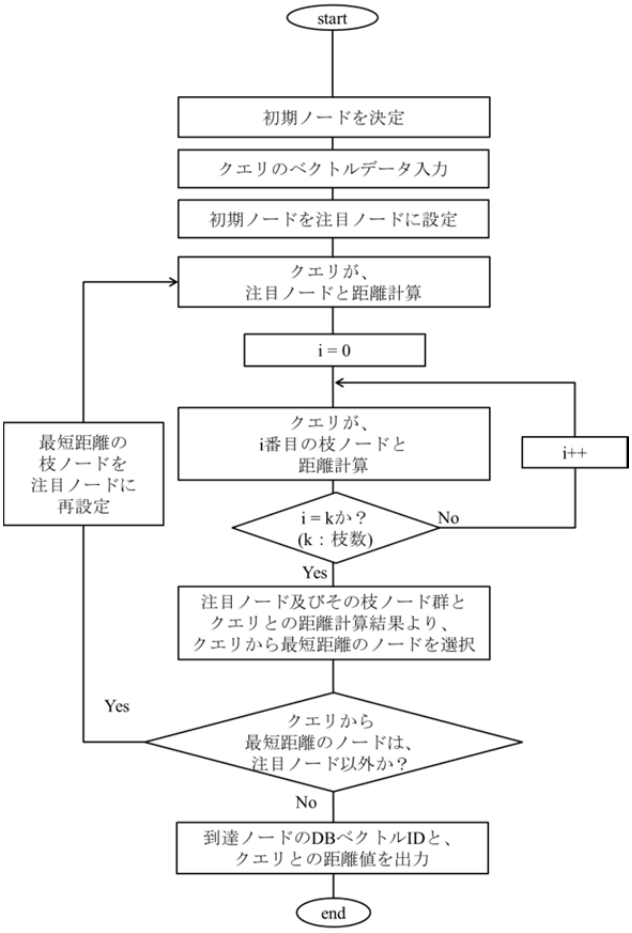

図 2　kNN グラフを用いた探索フロー

検討では，グラフ構築コストが小さい NN-Decent を用いて，近似最近傍探索の高速化手法について検討する．

2. 従来技術

2.1 kNN グラフ

kNN グラフは，探索の対象となるデータベース（以下，DB）内の近傍ベクトルどうしが，互いに枝を張ることで，グラフを構築している（図 1）．この時，k はあるベクトルから見たときの近傍ベクトル数であり，kNN グラフの枝の数に相当する．

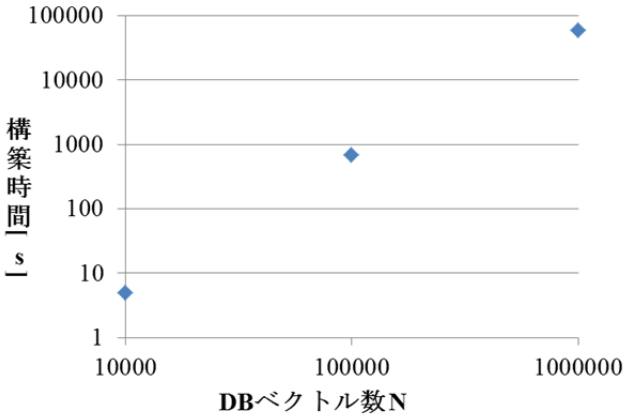

図 3　Brute Force による厳密な kNN グラフの構築時間
枝数 k によらず一定，100 万件で約 10 時間（右端）
（Intel® Xeon® X5690 @3.47GHz ×2　12cores）

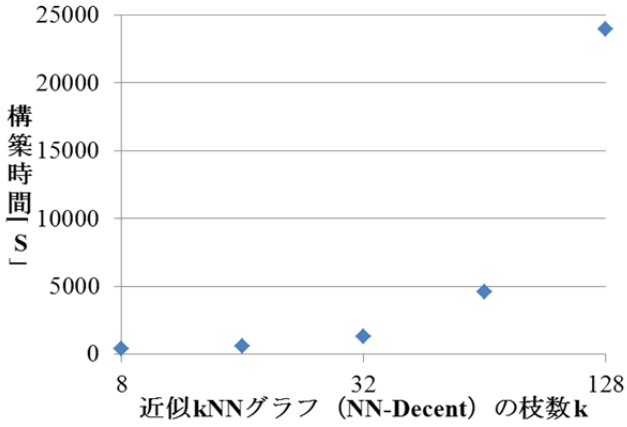

図 4　NN-Decnet による kNN グラフ構築時間
枝数 k を変数とし，100 万個の 128 次元ベクトルで構築
（Intel® Xeon® X5690 @3.47GHz ×1　1core）

次に，kNN グラフを用いた探索の一例を示す（図 2）．まず探索を開始するノード（以下，初期ノード）を決定し，注目ノードとする．クエリとなるベクトルデータ（以下，クエリベクトル）が入力されると，注目ノード及び注目ノードの枝ノード群とクエリベクトルとの距離計算を行う．もし，注目ノードとの距離が最も短ければ，注目ノードを最近傍ベクトルとして出力する．計算対象となったベクトルの中で，最短距離のベクトルが枝ノードにあった場合は，その枝ノードを注目ノードとして再設定し，注目ノードが最短距離となるまで距離計算を繰り返す．

kNN グラフでは，グラフの枝数を増やすことで，探索精度を向上させることができる．しかし，枝数を増やすことで探索時間は遅くなってしまう．その為，少ない枝数で高精度な探索法の確立が求められている．1 つのアプローチとして，グラフの枝数を増やす方法があるが，枝数を増や

Vol.2013-DPS-154 No.55
Vol.2013-CSEC-60 No.55
2013/3/15

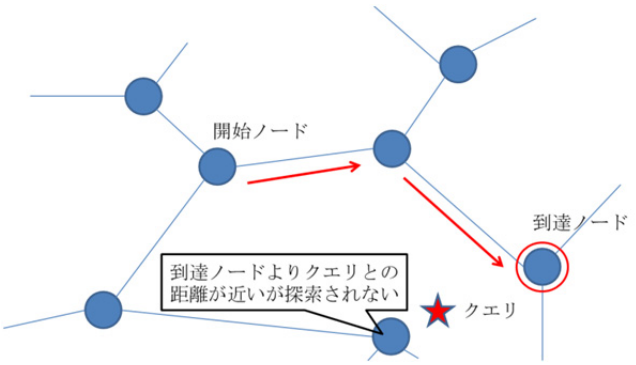

図 5　到達ノード以外が最近傍になるケース

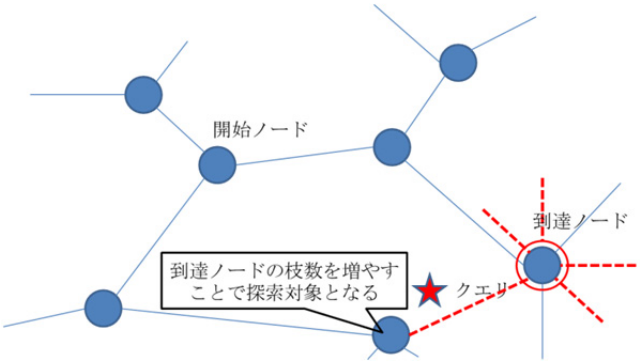

図 6　提案手法
（到達ノードに達した後、枝数を増やして周囲を再探索）

すことで，計算コストが向上する為，単純に枝数を増やすだけでは，高精度かつ高速な探索を実現するのは難しい．

また，kNN グラフのもう 1 つの課題として，グラフの構築時間が挙げられる．Brute Force にて，厳密なグラフを構築すると，ベクトル数を N とした場合に，$O(N^2)$のオーダで構築コストがかかる．事前実験において，次元数 128 のベクトルデータ 100 万件に対して，kNN グラフを構築すると，3.47GHz の CPU を 12 コア用いて並列処理した場合でも，約 10 時間もの時間を要した（図 3）．

2.2 近似 kNN グラフ（NN-Decent）

近似 kNN グラフ構築手法として，NN-Decent が注目されている．この手法は，まずランダムにグラフを構築し，注目ベクトルの近傍の近傍を見て，その中に，近傍ベクトルより距離が近いベクトルがあれば，枝を張り替えるといった手法である．NN-Decnet は，$O(k^{1.8} \cdot N^{1.14})$のオーダでグラフを構築することができ，枝数 k が小さいと Brute Force での厳密なグラフ構築に比べて大幅にグラフ構築時間を短縮できる．実際に，枝数 k の値を変化させながら，100 万件の 128 次元のベクトルデータに対して，構築時間を測定した（図 4）．その結果，並列化していないにもかかわらず，厳密なグラフ構築時間と比較しても，枝数 k が小さいと大幅な高速化が可能であることを確認した．従って，NN-Decent のような近似 kNN グラフを用いることで，高速にグラフを構築することができる．

3. 提案手法

3.1 GPGPU を用いた探索範囲拡大手法

本提案手法では，2 つの点に着目し，高精度かつ高速な探索を実現している．

1 つ目は，グラフ探索後に再度探索範囲を拡大させることで探索精度の向上を図っている．kNN グラフでは，グラフ探索後に最近傍ノードとされたノード以外に，真の最近傍が存在する可能性があり（図 5），枝数 k が小さいグラフでは，この真の最近傍を逃す可能性が高くなる．この解決策としては，枝数 k を増やすことが考えられるが，単純に枝数を増やすと，グラフ構築時間と探索時間の長大を招く．そこで，グラフの探索は，少ない枝数で行い，探索後に再度，広範囲をみることで真の最近傍に出会える確率の向上を図った（図 6）．

2 つ目は，GPGPU の利用である．GPGPU は，SIMD（Single Instruction Multiple Data）命令が得意である．そして，個々のベクトル距離計算は独立している為，複数のベクトル計算を GPGPU により並列化することで高速化することが期待できる．その為，グラフ探索後に再度広範囲を探索する場合に，探索対象となったベクトル群との距離計算に対して，GPGPU を用いて並列化することで，探索処理の高速化を図った．

3.2 処理の流れ

提案手法の近似最近傍探索法の処理フローを図 7 に示す．事前処理として，ホスト側のメモリ上に近似 kNN グラフが構築されており，探索対象となる DB 側のベクトルデータが，ホスト，GPGPU 両方のメモリ上に保持されているものとする．

まず，クエリベクトルが入力されると，初期ノードを注目ノードとし，注目ノード及びその枝ノードとクエリベクトルとの距離計算を行う．もし，注目ノードが最短距離ならば，注目ノードを到達ノードとして，グラフ探索を終える．計算対象となったノードの中で，最短距離となったノードが枝ノードならば，その枝ノードを注目ノードとして再設定する．そして，注目ノードとの距離が，枝ノードより短くなるまでグラフ上での探索を繰り返す．ここまでは，kNN グラフを用いた従来の探索方法と同じである．

到達ノードに達した後，到達ノードの多数の近傍ベクトルとクエリベクトルとの距離計算を再度行う．この時，再度距離計算を行う多数の近傍ベクトル数を k’とおく．k’は枝数 k よりも大きい為，この再計算は計算コストが大きくなる．そこで，GPGPU を用いて並列化することで，距離計算の高速化を測る．従って，グラフ探索中は CPU で処理を行い，到達ノードに達した後の k’個のベクトルとの距離計算は GPGPU を用いて処理を実施している．

Vol.2013-DPS-154 No.55
Vol.2013-CSEC-60 No.55
2013/3/15

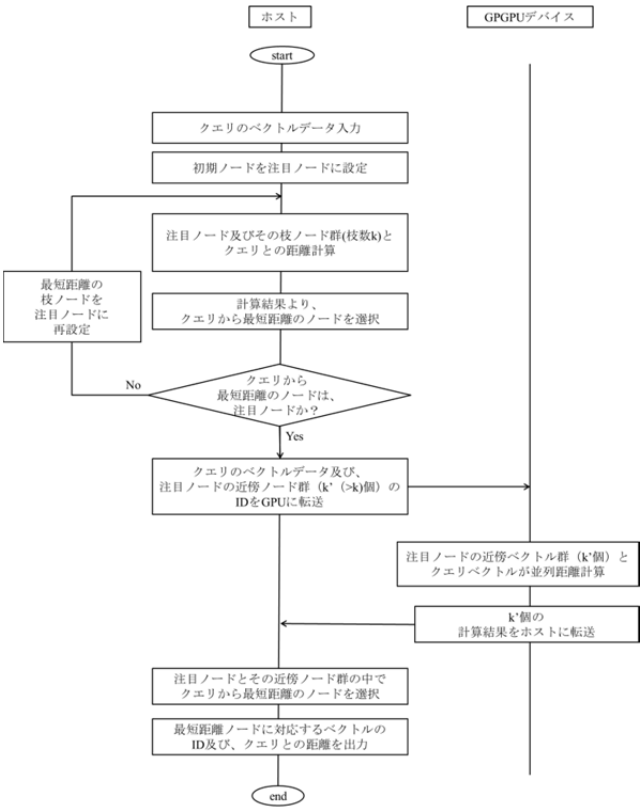

図 7　提案手法の処理フローの例

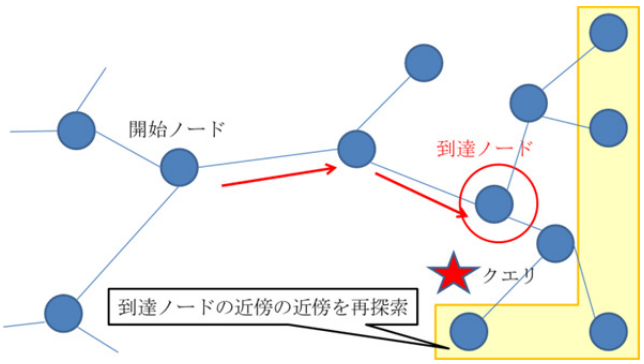

図 8　到達ノードの近傍の近傍を再探索する例

グラフ探索後に，再度距離計算を行う k’個の近傍ベクトルの選定方法は，単純に枝数 k’（>k）でグラフを構築する方法も考えられるが，大きな枝数でグラフを構築しようとすると厳密な kNN グラフではもちろんのこと，近似 kNN グラフの NN-Decent でも構築コストが大きくなる為，現実的ではない．そこで本手法では，近傍の近傍，さらには近傍の近傍の近傍とみていく範囲を広げていくことで，少ない枝数 k でグラフを構築するが，k’の値を可変かつ大きく設定できることも可能にした（図 8）．それによって，k’の値を変えることでも，探索精度と探索時間を調節することを可能にした．

本提案手法によって，少ない枝数で近似 kNN グラフを構築しても，グラフ探索後にグラフ探索中よりも多くの近傍ノードと再度距離計算すること精度向上を図る．その時，多数のベクトルとの距離計算を，GPGPU の利用によって並列度を上げて高速化することで，高い精度を維持した高速な近似最近傍探索の実現が期待できる．

4. 実験・考察

4.1 実装

本検討では，128 次元の画像特徴量ベクトルを用いて実験を行った．ベクトル数は，1 画像から約 1000 特徴ベクトルが抽出されると想定し，クエリベクトル数を 1000 ベクトルとした．DB 側のベクトル数は 100 万ベクトルとした．

実験環境としては，CPU（Intel® Xeon® X5690 @3.47GHz）とメインメモリがあるホストと，GPGPU デバイス（NVIDIA® TESLA® C2070）を搭載したマシン 1 台を用いて測定を行った．

従来手法は，注目ノード及びその近傍ノード群とクエリベクトルまでの距離を計算し，注目ノードが最短距離になるまで注目ノードを再設定する処理を行った．
一方，提案手法はグラフ探索後に，到達ノードから距離が近い k/4 個の近傍ノード群を選択し，さらにその近傍ノードの近傍ノード群から距離の近い k/4 個の近傍ノード群，つまり到達ノードの近傍の近傍のノード$k^2/16$個を対象とした．そして，それらのベクトル群との距離計算を，GPGPU を用いて並列化した．

4.2 実験

提案手法の効果を確認する為，2 つの実験を実施した．まず，厳密な kNN グラフを用いて従来手法で探索した場合と比較し，近似 kNN グラフでも同程度の探索精度と探索時間を維持できるかを検証した．

次に，本提案手法を厳密な kNN グラフと近似 kNN グラフに対して適用し，探索処理を実施した場合に，探索精度にどの程度の差が生じるかを検証する為の実験を行った．

本実験における探索精度とは，入力 1000 ベクトルに対して，近似最近傍探索を行った時に，入力全体からみて何割の入力ベクトルが，厳密な最近傍探索を行った時と同じ結果になったかを表している．また，探索時間は，入力 1000 ベクトルが入力されてから，全ての入力ベクトルの最近傍ベクトルが求められるまでの時間を測定している．

いずれの実験においても，初期ノードを増やす際は，1，3，6，12，25，50，100 とノード数を増加させ検証を行った．

4.2.1 実験 1：枝数固定での従来の探索手法との比較

近似 kNN グラフに対して，枝数 k を 8 から 64 までそれぞれ固定し，初期ノード数を 1 個から 100 個まで変化させた時の探索精度と探索時間を，従来手法と提案手法で比較した（図 9～12）．グラフ上で初期ノード数は，右にいくほど増加している．

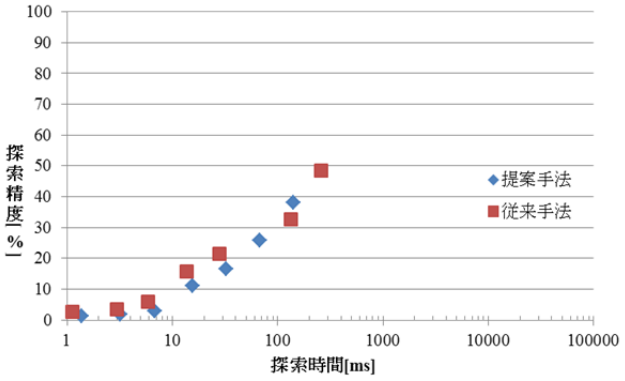

図 9　k=8 での従来手法と提案手法の比較

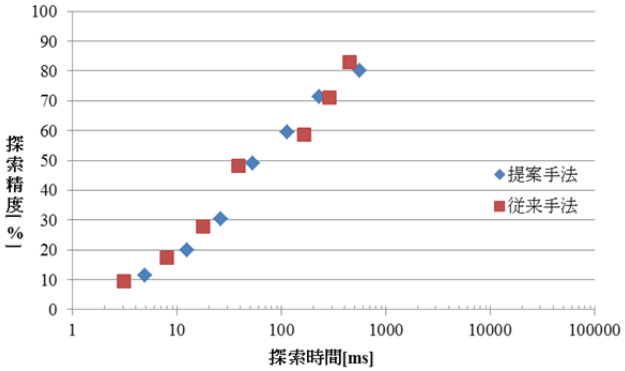

図 10　k=16 の時の従来手法と提案手法の比較

測定の結果，従来手法と提案手法共に初期ノードが増えるほど，探索精度は高くなり，探索時間は長くなる傾向を示した．枝数の違いでもみても，両手法共に，枝数の多い方が，同じ初期ノード数で比較した場合，探索精度は高くなり，探索時間は長くなる傾向を示した．同じ枝数にて，同等の探索時間で比較した場合，従来手法に対して，提案手法は同等の精度を有していることがわかった．

また，提案手法において，初期ノードを増やすことで，枝数が少なくても，枝数が多い場合と同等の高い探索精度での高速探索を実現している．例えば，図 13 において，枝数 k=64，初期ノード 3 個で，探索精度 78%，探索時間 107 msec という結果を示しているが，枝数 k=32，初期ノード 12 個でも，探索精度 77%，探索時間 139 msec と，同程度の探索精度と探索時間を達成することができた．従って提案手法は，少ない枝数で高い精度を維持した高速な探索が可能であることがわかった．

4.2.2 実験 2：厳密 kNN グラフと近似 kNN グラフでの比較

Brute Force で厳密に構築した kNN グラフと，NN-Decent で近似的に構築した kNN グラフそれぞれに本提案手法を適用し，探索精度と探索時間を比較した．枝数 k を 8 から 64 までそれぞれ固定した場合に，初期ノード数を変化させ，その時の探索精度と速度を比較した（図 14～17）．グラフ上で初期ノード数は，右にいくほど増加している．

測定の結果，いずれの枝数においても，同程度の探索時

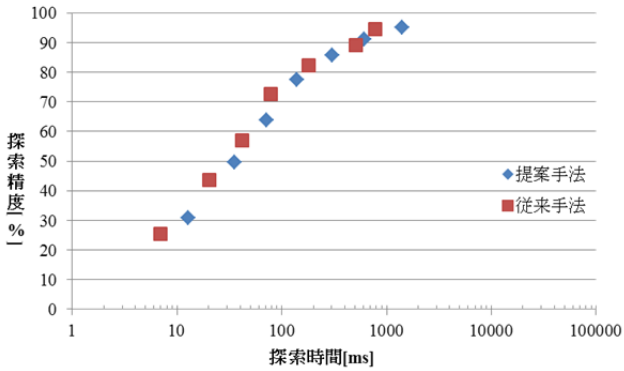

図 11　k=32 の時の従来手法と提案手法の比較

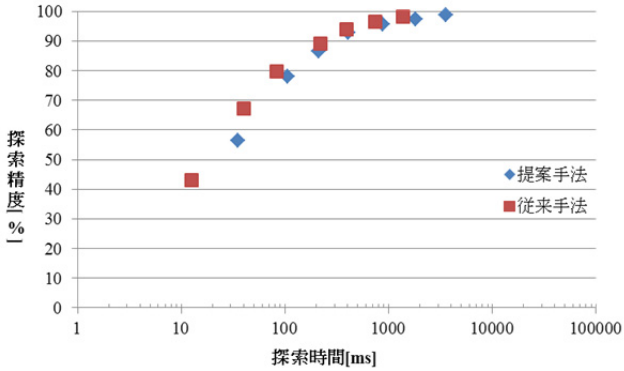

図 12　k=64 の時の従来手法と提案手法の比較

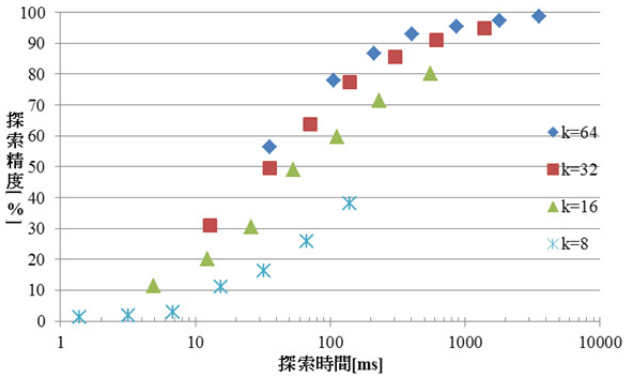

図 13　k 可変時の提案手法の探索精度と時間

間で比較した場合，厳密な kNN グラフの方が高精度に探索できる結果となった．

4.3 考察

実験 1 の結果から，近似 kNN グラフと本提案手法の組み合わせによる探索が，厳密な kNN グラフに従来手法を用いた探索法と同程度の探索精度と探索時間を維持していることが確認できた．また，枝数 k が少ない場合でも，高い精度を維持した高速な探索が可能であった．

実験 2 の結果から，本提案手法を厳密な kNN グラフと近似 kNN グラフに適用した場合，厳密な kNN グラフの方が，同等の探索速度で比較した場合，高い精度を維持していることがわかった．

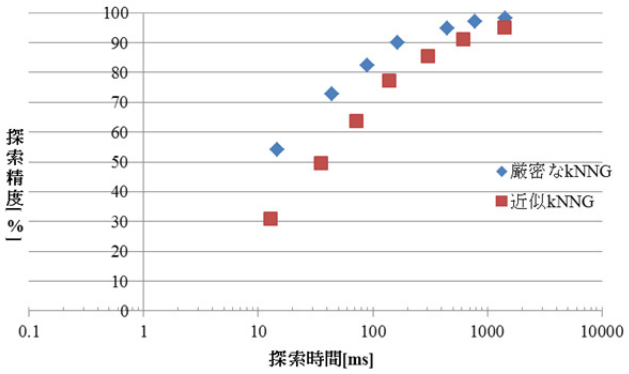

図 14　kNN グラフへの提案手法を適応結果（k=8）

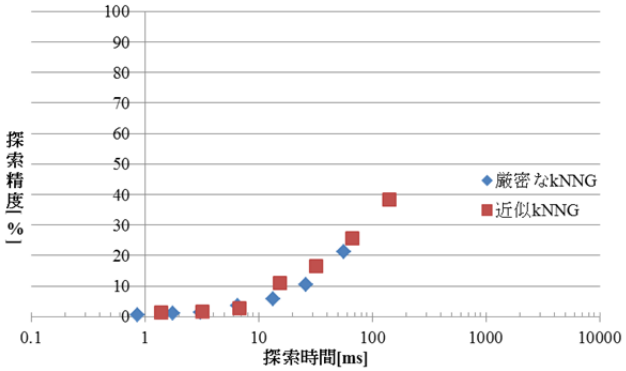

図 15　k= NN グラフへの提案手法を適応結果（k=16）

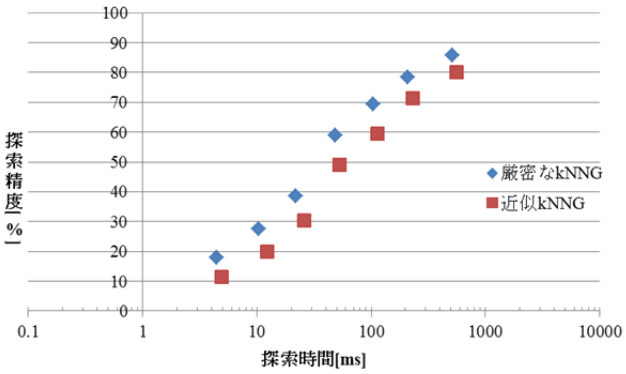

図 16　　NN グラフへの提案手法を適応結果（k=32）

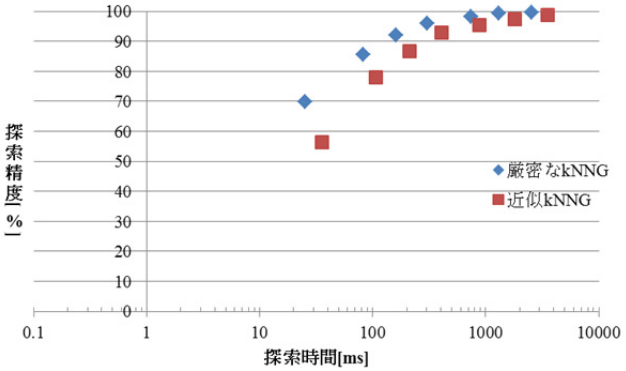

図 17　　NN グラフへの提案手法を適応結果（k=64）

従って，従来手法と比較した場合に，本提案手法は，グラフ構築コストの小さく抑えつつ，高精度かつ高速な探索が可能であることが確認できた．

5. まとめ

本検討では，大規模な高次元ベクトルに対して，構築時間の短いインデックスを用いた，高精度かつ高速な近似最近傍探索法を提案した．インデックスとして，グラフの枝数が少ない場合に，構築コストが小さい近似 kNN グラフである NN-Decent を用いた．kNN グラフは，枝数が少ないと精度が低下する．しかし本提案手法では，従来手法と同様の探索後に，GPGPU を用いて広範囲のベクトルと再計算することで高い探索精度を維持した高速な探索法を実現した．今後の課題としては，web 上の画像群を対象とする為に，検証環境を大規模なクラスタ環境に拡張し，数百億から数兆個のベクトル数に対して，本提案手法を適用し，効果を確認する必要がある．

謝辞　本検討の遂行にあたり，近似 kNN グラフの NN-Decent 構築において，ライブラリのご提供及び，ご助言を頂きました日本電信電話株式会社　コミュニケーション科学基礎研究所の青山 一生 様に心より感謝申し上げます．

参考文献

1)　和田俊和: 最近傍探索の理論とアルゴリズム，情報処理学会 CVIM 研究会, vol.2009-CVIM-169, no.12, pp.1-12, Nov.2009.
2)　D. Lowe, Disrinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision, vol.60, no.2, pp.91-110, Jan.2004.
3)　H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, SURF: Speeded Up Robust Features, Computer Vision and Image Understanding, vol.110, no.3, pp.346-359, June.2008.
4)　Flickr, http://www.flickr.com/
5)　J. L. Bentley, Multidimensional binary search trees used for associative searching, Communications of the ACM, vol.18, no.9, pp.509-517, Sept.1975.
6)　松村聖司，毛受崇，赤間浩樹，松尾嘉典，奥村昌和，山室雅司: kd-tree により構造化された大規模高次元ベクトル群に対する GPU を用いた高速最近傍探索法の検討，情報処理学会 DPS 研究会, Vol.2012-DPS-150, no.24, pp.1-8, Feb.2012.
7)　Mayur Datat, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni, Locality-sensitive hashing scheme based on p-stable distributions, SCG '04 Proceedings of the twentieth annual symposium on Computational geometry, pp.253-262, June.2004.
8)　Thomas B. Sebastian, and Benjamin B. Kimia, Metric-based Shape Retrieval in Large Databases, Pattern Recognition, 2002. Proceedings. 16th International Conference on, vol.3, pp.291-296, Aug.2002.
9)　W. Dong, M. Charikar, and K. Li, Efficient K-Nearest Neighbor Graph Construction for Generic Similarity Measures, WWW '11 Proceedings of the 20th international conference on World wide web, pp.577-586, March.2011.