2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

Compression for Very Sparse Big Social Data

Carson K. Leung[1, (✉)], Yibin Zhang[1, 2], Fan Jiang[3]
[1] *Department of Computer Science, University of Manitoba*
Winnipeg, MB, Canada
[2] *Department of Computer Science, University of Toronto*
Toronto, ON, Canada
[3] *Department of Computer Science, University of Northern British Columbia* (*UNBC*)
Prince George, BC, Canada
(✉) Email: kleung@cs.umanitoba.ca

***Abstract*—Technological advancements in the current era of big data have led to rapid generation and collection of very large amounts of valuable data from a wide variety of rich data sources. As rich data sources, social networks consist of social entities that are linked by some social relationships (e.g., kinship, colleagueship, co-authorship, friendship, followship). Usually, these networks are very big but also very sparse. Embedded in the very sparse but very big networks are implicit, previously unknown and potentially useful information and knowledge that can be discovered by social network analysis and mining. In this paper, we aim to discover interesting social relationships from very sparse but very big social network data. Due to the sparsity of the data, we effectively compress bitmaps representing social entities in the data, from which useful information can be mined and interesting knowledge can be discovered. Evaluation results show the effectiveness of our compression scheme for very sparse but very big social network data.**

Keywords—social network analysis, social network mining, data mining, big data science, big data analytics, compression

I. INTRODUCTION

Technological advancements in the current era of big data [1-6] have led to rapid generation and collection of very large amounts of valuable data from a wide variety of rich data sources. These big data can be of different level veracity, with some precise data and some imprecise and uncertain data [7-9]. Examples include data generated or collected from:

- bio-engineering, bio-informatics, and bio-medical applications (e.g., omic data like genomic data [10-13]);
- e-commerce activities [14, 15],
- entertainment (e.g., movies) [16], games [17, 18], and music [19-21];
- financial and stock markets [22-24];
- healthcare sector [25] (e.g., disease reports [26, 27], epidemiological data and statistics [28-33]);
- traffic and road conditions [34-39].

In addition, as one of the aforementioned rich data sources, social networks [40-43] (e.g., co-authorship networks [44, 45]) consist of social entities that are linked by some social relationships. For instance, a social entity can be the next-of-kin, colleague, co-author, mutual friend, follower, and/or followee of another social entity in a social network.

To elaborate, in social networking sites like Facebook, users can create a personal profile and add other users as friends. For instance, a Facebook user X can add another Facebook user Y as a friend by sending Y a friend request. Upon Y's acceptance of X's friend request, X and Y can become *mutual friends*. In addition to exchanging messages among mutual friends, Facebook users can also join common-interest user groups and categorize their friends into different customized lists (e.g., classmates, co-workers). The number of (mutual) friends may vary from one Facebook user to another.

Besides mutual friendship, another common linkage between users in social networks is *followship* (also known as *follower-followee relationship* or *"following" pattern*) [46], which captures the linkage that a social network user X follows another user Y. Let us elaborate by continuing with the aforementioned example on Facebook users. Although many of the Facebook users are linked to some other Facebook users via the mutual friendship (i.e., if a user X is a friend of another user Y, then user Y is also a friend of user X), there are also situations in which such a relationship is no longer mutual. To handle these situations, Facebook added the functionality of 'subscribe' in 2011, which was relabelled as 'follow' in 2012. Specifically, a user can subscribe or follow public postings of some other Facebook users—usually, famous celebrities, public institutions, product and services, news media, and well-known bloggers—without the need of adding them as friends. A user X may follow other users who do not know user X. In this situation, the link between these social entities is no longer mutual (i.e., undirectional) but a directional "following" pattern from followers to followees. Note that this follower-followee relationship is common in many social networking sites such as Instagram and Twitter, in which a user X can 'follow' the Instagram and Twitter accounts of another user Y, but it is not necessary that user Y follows back the corresponding accounts of user X. Similarly, in YouTube, a user X can 'subscribe' to YouTube channels of another user Y, but again it is not necessary that user Y subscribes/follows back the corresponding channels of user X.

To recap, mutual friendship—e.g., as captured by Facebook, where two social entities are mutual friends of each other—is

IEEE/ACM ASONAM 2020, December 7-10, 2020
978-1-7281-1056-1/20/$31.00 ©2020 IEEE

undirectional or bidirectional. In contrast, *followship* or *follower-followee relationship*—e.g., as captured by Instagram and/or Twitter, where a user (i.e., follower) follows another user (i.e., followee)—is directional from the follower to the followee. Note that these networks are usually very big but also very sparse. For example, as of July 2020[1], although there were 1.08 billion monthly active users (MAU) in Instagram, the average number of followers in a personal Instagram account[2] is about 150.

In general, data science [47, 48]—which applies data mining [49-53], machine learning [54], mathematical and statistical modelling [55], etc.—can discover implicit, previously unknown and potentially useful information and knowledge that are embedded in the big data. Specifically, *social network analysis and mining* can discovered discover useful information and knowledge from the aforementioned very big but very sparse social networks.

With social network analysis and mining for the follower-followee relationships, various recommendations can be made. For instance, when many friends of a user X follow some individual users (or groups of famous users), it is likely that user X may also be interested in following these individual users (or groups of famous users). This leads to a collection of most-followed users, which include Instagram accounts of some sports players, popular performers, public figures, and politicians. For instance, as of December 2020, the most-followed Instagram accounts[3] include those of (a) Portuguese soccer player Cristiano Ronaldo; (b) American musician & actress Ariana Grande; (c) American-Canadian actor & professional wrestler Dwayne Johnson (aka The Rock); (d) American TV personality, model & cosmetic business-woman Kylie Jenner; and (e) American singer, actress & producer Selena Gomez. Then, upon the discovery of *frequently followed groups* (i.e., groups of famous users or social entities, who are followed by a significant number of common users), if any user X in the social network follows some members of these groups, then we could recommend other members of these groups to user X.

To find these frequently followed groups, an effective way to represent very big but very sparse social network is needed. A compressed bitmap is a logical way as it has been applied to various application areas including compression of data [56, 57], image and video compression [58, 59], as well as sequence compressions (e.g., DNA sequences) [60]. Compressed data in these application areas help speed up the information retrieval of data in the areas. However, as they were not designed for social network analysis and mining, most of them cannot be easily adapted to compressing social networking sites.

Regarding related works that focus on compressing social networks for frequent pattern mining and analysis, we [61] presented a social network mining strategy in the IEEE/ACM ASONAM 2016. The strategy applies the word-aligned hybrid (WAH) compression model to take advantage of the sparsity of "following" data. The idea behind this compression model is to divide the long bitmap into groups of 31 bits, then encode long-run of consecutive zero groups (i.e., groups without any "1"-bit) into a compressed word. If a "1"-bit appears in a group, then the group is stored without compression.

Observed that a few "1"-bits are commonly following a long-run of consecutive zero groups of "0" for very sparse data set, we [62] presented in the IEEE/ACM ASONAM 2017 a solution to deal with these commonly observed situations. Specifically, our solution—namely, the improved position list word-aligned hybrid (IPLWAH) compression model—encodes both the long run of consecutive zero groups of "0" and its (at most k) "1"-bits in the succeeding group of 31 bits.

Observed that there are situations in which a few "1"-bits appear in multiple consecutive groups (instead of a single group) succeeding a long run consecutive groups of "0", we [63] presented in the IEEE/ACM ASONAM 2019 a solution with a flexible compression model. Specifically, our solution—namely, the multi-line improved position list word-aligned hybrid (M-IPLWAH) compression model—encodes both the long run of consecutive zero groups of "0" and its (at most k) "1"-bits in multiple succeeding groups of 31 bits. Here, these succeeding groups must contain at least one "1"-bit.

However, we also observed that there are situations in which the few "1"-bits appear in multiple groups succeeding a long run consecutive groups of "0", but not all these succeeding groups contain at least one "1"-bit. For example, "1"-bits may appear in the first and the third groups—but not the second group—succeeding a long run consecutive groups of "0". Both IPLWAH and M-IPLWAH were not designed to handle the situations (with a gap between groups containing "1"-bits). Here, in the current IEEE/ACM ASONAM 2020 paper, we come up a solution to deal with the situations. Our key contributions of this paper include our design and development of this solution—namely, a *gapped-line improved position list word-aligned hybrid* (*G-IPLWAH*) *compressed bitwise representation* of social networks.

The remainder of this paper is organized as follows. The next section provides background and related works. Section III presents our compression model G-IPLWAH. Evaluation and conclusions are given in Sections IV and V, respectively.

II. Background and Related Works

A social network can be represented as a collection of bit vectors (i.e., bitmaps). Each vector corresponds to a follower, and represents the list of its followees. Specifically, a "1" in the i-th bit of the vector indicates that the follower follows the i-th followee in the social network. Recall from Section I that, as the social network can be very big, the vector can be long. Moreover, as the network can be very sparse, the number of "1"-bits in each vector can be small.

A. Word-Aligned Hybrid (WAH) Compression

When applying the word-aligned hybrid (WAH) compression model [61] for social network analysis and mining,

[1] https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/
[2] https://www.hashtagsforlikes.co/blog/instagram-followers-how-many-does-the-average-person-have/
[3] https://www.statista.com/statistics/421169/most-followers-instagram/

the bit vectors for followers are compressed as follows. Bits in each vector are divided into groups of 31 bits. Then,

- If 31 bits in a group are all zeros, then the group is categorized as a zero-fill group. Consecutive zero-fill groups (of 31 zeros) are then combined and compressed into a *zero-fill word*, which is represented as a 32-bit word. Here,
    - the zero-fill word is prefixed with "10", where the first bit "1" indicates that it is a fill word, and the second bit "0" indicates that it is a 0-fill word; and
    - the suffix 30 bits indicate the number of consecutive groups of 31 zeros.
- If 31 bits in a group contain a mixture of "0"- and "1"-bits, then these mixture bits are put into a *literal word*. Specifically, a literal word is also represented as a 32-bit word, which is
    - prefixed with "0" to indicate that its identity (i.e., word prefixed with "0" is a literal word); and
    - the suffix 31 bits capture the mixture of "0"- and "1"-bits.

For illustrative purposes, let us consider a social network, in which Kees follows Aart (user #31728), Brechtje (user #63344), Cas (user #63349), Danique (user #63354), Evert (user #94611), Famke (user #94632), Gerrit (user #94653), Hannie (user #126231), Ignaas (user #126272).

Example 1. The original uncompressed bit vector for Kees contains "1"-bits at positions 31728 (for Aart), 63344 (for Brechtje), 63349 (for Cas), 63354 (for Danique), 94611 (for Evert), 94632 (for Famke), 94653 (for Gerrit), 126231 (for Hannie) and 126272 (for Ignaas). With at least 125532 bits, this vector requires 3,946 words (of 32 bits), for a total of 126,272 bits to capture all followees of a single follower Kees. This number is then multiplied by the number of followers to capture followees of all followers in the social network.

Example 2. Continue with Example 1. When applying the WAH compressed model, the vector for Kees can be compressed into a bitmap consisting of a sequence of 12 (zero-fill and literal) 32-bit words:

- The first 32-bit word in the sequence is a zero-fill word 10 00 0000 0000 0000 0000 0011 1111 1111, where (a) the prefix "10" indicates that it is a zero-fill word and (b) the suffix 11 1111 1111 $_{(2)}$ = 1023 $_{(10)}$ indicates 1023 groups of 31 consecutive zeros.
- The second 32-bit word in the sequence is a literal word 0 00000 00000 00001 00000 00000 00000 0, where (a) the prefix "0" indicates that it is a literal word and (b) the "1"-bit is in the 15th position of this word to represent the followee Aart (i.e., user#31728) because of the presence of "1"-bit in the 1023x31 + 15 = 31728th position in the original bit vector.
- The third 32-bit word in the sequence is a zero-fill word 10 00 0000 0000 0000 0000 0011 1111 1011, where the suffix indicates 11 1111 1011 $_{(2)}$ = 1019 $_{(10)}$ groups of 31 consecutive zeros.
- The fourth 32-bit word in the sequence is a literal word 0 00000 00000 10000 10000 10000 00000 0, where the three "1"-bits are in the 11th, 16th and 21st positions of this word. The "1"-bit in the 11th position represents the followee Brechtje (user #63344) because of the presence of "1"-bit in the (1023+1+1019)x31 + 11 = 63344th, position in the original bit vector. Similarly, the "1"-bits in the 16th and 21st positions of this word represent the followees Cas (user #63349) and Danique (user #63354) because of the presence of "1"-bits in the 63349th and 63354th positions in the original bit vector.
- The fifth 32-bit word in the sequence is a zero-fill word 10 00 0000 0000 0000 0000 0011 1110 1111, where the suffix indicates 11 1110 1111 $_{(2)}$ = 1007 $_{(10)}$ groups of 31 consecutive zeros.
- The sixth 32-bit word in the sequence is a literal word 0 00000 00000 00000 00000 00000 00001 0, where the "1"-bit is in the 30th position of this word to represent the followee Evert (user #94611) because of the presence of "1"-bits in the (1023+1+1019+1+1007)x31 + 30 = 94611th positions in the original bit vector.
- The seventh 32-bit word in the sequence is a literal word 0 00000 00000 00000 00001 00000 00000 0, where the "1"-bit is in the 20th position of this word to represent the followee Famke (user #94632) because of the presence of "1"-bits in the (1023+1+1019+1+1007+1) x 31 + 20 = 94632nd position in the original bit vector.
- The eighth 32-bit word in the sequence is a literal word 0 00000 00001 00000 00000 00000 00000 0, where the "1"-bit is in the 10th position of this word to represent the followee Gerrit (user #94653) because of the presence of "1"-bits in the (1023+1+1019+1+1007+2) x 31 + 10 = 94653rd position in the original bit vector.
- The ninth 32-bit word in the sequence is a zero-fill word 10 00 0000 0000 0000 0000 0011 1111 1001, where the suffix indicates 11 1111 1001 $_{(2)}$ = 1017 $_{(10)}$ groups of 31 consecutive zeros.
- The 10th 32-bit word in the sequence is a literal word 0 00000 00000 00000 00000 00001 00000 0, where the "1"-bit is in the 25th position of this word to represent the followee Hannie (user #126231) because of the presence of "1"-bits in the (1023+1+1019+1+1007+3 +1017)x31 + 30 = 126231st positions in the original bit vector.
- The 11th 32-bit word in the sequence is a zero-fill word 10 00 0000 0000 0000 0000 0000 0000 0001, where the suffix indicates 1 $_{(2)}$ = 1 $_{(10)}$ group of 31 consecutive zeros.
- Finally, the 12th 32-bit word in the sequence is a literal word 0 00000 00000 00000 00000 00001 00000 0,

where the "1"-bit is in the 9th position of this word to represent the followee Ignaas (user #126272) because of the presence of "1"-bits in the (1023+1+1019+1+1007+3 +1017+1+1)x31 + 9 = 126272nd positions in the original bit vector.

In other words, this WAH compressed bitmap only requires 12x32 = 384 bits (cf. 126,272 bits for the original uncompressed bit vector) to capture all followees of the follower Kees.

B. Improved Position List Word-Aligned Hybrid (IPLWAH) Compression

Our improved position word-aligned hybrid IPLWAH(k) compression model [62] improves WAH for social network analysis and mining by combining:

- a zero-fill word (i.e., long run of consecutive zero groups of "0"), with
- its succeeding literal word containing at most k "1"-bits.

The combined word is still (a) represented as a 32-bit word and (b) having prefix "10" to indicate its identity (i.e., a compressed zero-fill word). However, it uses at most k collections of 5 bits (e.g., 3rd-7th bits, 8th-12th bits, ...) to indicate the positions of at most k "1"-bits in a single group of 31 bits succeeding the long run of consecutive groups of 31 zeros. The number of these consecutive groups is indicated by the suffix 30–5k bits. Practically, $1 \leq k \leq 5$ for our IPLWAH(k) compression model. To summarize, a zero-fill word in IPLWAH(k)—for $1 \leq k \leq 5$ practically—is represented as a 32 bit word with:

- prefix "10" to indicate its identity (i.e., a compressed zero-fill word);
- next k collections of 5 bits—in the 3rd bit to the (5k+2)-th bit (e.g., 3rd-7th bits, 8th-12th bits, 13th-17th bits, ...)—indicate the positions of at most k "1"-bits in a single group of 31 bits succeeding the long run of consecutive groups of 31 zeros; and
- suffix 30–5k bits (e.g., suffix 25 bits, 20 bits, 15 bits, ...) to indicate the number of these of consecutive groups of 31 zeros.

Example 3. Continue with Example 2. With our IPLWAH(1) compressed model, the vector for Kees can be further compressed into a bitmap consisting of a sequence of eight (zero-fill and literal) 32-bit words. Most of the words in this IPLWAH(1) compressed bitmap are identical to those in the WAH compressed bitmap in Example 2, except the following:

- The first two 32-bit words in the WAH bitmap are compressed to become the first 32-bit word in this IPLWAH(1) sequence, which is a compressed zero-fill word 10 01111 0 0000 0000 0000 0011 1111 1111, where (a) the prefix "10" indicates that it is a zero-fill word, (b) the suffix 11 1111 1111 $_{(2)}$ = 1023 $_{(10)}$ indicates 1023 groups of 31 consecutive zeros are followed by (c) a "1"-bit at position 1111 $_{(2)}$ = 15 $_{(10)}$ of the succeeding word representing Aart.
- The fifth and sixth 32-bit words in the WAH bitmap are compressed to become the fourth 32-bit word in this IPLWAH(1) sequence, which is a compressed zero-fill word 10 11110 0 0000 0000 0000 0011 1110 1111, where (a) the suffix indicates 11 1110 1111 $_{(2)}$ = 1007 $_{(10)}$ groups of 31 consecutive zeros are followed by (b) a "1"-bit at position 11110 $_{(2)}$ = 30 $_{(10)}$ of the succeeding word representing Evert.
- The ninth and tenth 32-bit words in the WAH bitmap are compressed to become the seventh 32-bit word in this IPLWAH(1) sequence, which is a compressed zero-fill word 10 11001 0 0000 0000 0000 0011 1111 1001, where (a) the suffix indicates 11 1111 1001 $_{(2)}$ = 1017 $_{(10)}$ groups of 31 consecutive zeros are followed by (b) a "1"-bit at position 11001 $_{(2)}$ = 25 $_{(10)}$ of the succeeding word representing Hannie.
- Finally, the 11th and 12th 32-bit words in the WAH bitmap are compressed to become the eighth 32-bit word in this IPLWAH(1) sequence, which is a compressed word 10 01001 0 0000 0000 0000 0000 0000 0001, where (a) the suffix indicates 1 $_{(2)}$ = 1 $_{(10)}$ group of 31 consecutive zeros is followed by (b) a "1"-bit at position 1001 $_{(2)}$ = 9 $_{(10)}$ of the succeeding word representing Ignaas.

In other words, this IPLWAH(1) compressed bitmap only requires 8x32 = 256 bits (cf. 384 bits for the WAH compressed bitmap, and 126,272 bits for the original uncompressed bit vector) to capture all followees of the follower Kees.

Example 4. Continue with Example 3. Further compression is possible for higher values for k in our IPLWAH(k) compressed model. As an example, when k=3, the vector for Kees can be compressed into a bitmap consisting of a sequence of seven (zero-fill and literal) 32-bit words. Most of the words in this IPLWAH(3) compressed bitmap are identical to those in the IPLWAH(1) compressed bitmap in Example 3, except that:

- the second and third 32-bit words in the IPLWAH(1) bitmap are compressed to become the second 32-bit word in this IPLWAH(3), which is a compressed zero-fill word 10 01011 10000 10101 000 0011 1111 1111, where (a) the prefix "10" indicates that it is a zero-fill word, (b) the suffix 11 1111 1111 $_{(2)}$ = 1023 $_{(10)}$ indicates 1023 groups of 31 consecutive zeros are followed by (c) three "1"-bits at positions 1011 $_{(2)}$ = 11 $_{(10)}$, 10000 $_{(2)}$ = 16 $_{(10)}$ and 10101 $_{(2)}$ = 21 $_{(10)}$ of the single succeeding word representing Brechtje, Cas and Danique.

In other words, this IPLWAH(3) compressed bitmap only requires 7x32 = 224 bits (cf. 256 bits for the IPLWAH(1) compressed bitmap, 384 bits for the WAH compressed bitmap, and 126,272 bits for the original uncompressed bit vector) to capture all followees of the follower Kees.

C. Multi-line Improved Position List Word-Aligned Hybrid (M-IPLWAH) Compression

Our multi-line improved position word-aligned hybrid M-IPLWAH(k) compression model [63] further improves IPLWAH(k) for social network analysis and mining by combining:

- a zero-fill word (i.e., long run of consecutive zero groups of "0"), with
- its multiple consecutive succeeding literal words containing at most k "1"-bits (with at least one "1"-bit in each literal word).

A key difference between IPLWAH(k) and this M-IPLWAH(k) compression models is that, the former only combines a single literal word (containing at most k "1"-bits) succeeding the zero-fill word, whereas the latter can combine multiple literal words (with at most k '1"-bits distributed among these consecutive literal words with at least one "1"-bit in each literal word) succeeding the zero-fill word.

With our M-IPLWAH(k) compression model, the combined word is still (a) represented as a 32-bit word and (b) having prefix "10" to indicate its identity (i.e., a compressed zero-fill word). It uses at most k collections of 5 bits (e.g., 3rd-7th bits, 8th-12th bits, ...) to indicate the positions of at most k "1"-bits, which can be distributed among multiple consecutive groups of 31 bits succeeding the long run of consecutive groups of 31 zeros. The number of these consecutive groups is indicated by the suffix 31–6k bits. Moreover, (k–1) bits are used for indicating whether the current collection of 5 bits is on the same "line" (i.e., is in the same group) as the previous ones. Practically, $1 \leq k \leq 4$ for our M-IPLWAH(k) compression model. To summarize, a zero-fill word in M-IPLWAH(k)—for $1 \leq k \leq 4$ practically—is represented as a 32 bit word with:

- prefix "10" to indicate its identity (i.e., a compressed zero-fill word);
- next (k–1) bits—from the 3rd bit to the (k+1)-th bit (e.g., 3rd, 4th, 5th bits)—serve as flags to indicate whether the next "1"-bit is on the same "line" (i.e., the same literal word) as the current one (e.g., whether the second "1"-bit is on the same "line" as the first one, whether the third "1"-bit is on the same "line" as the second one, ...). Specifically:
  - a flag with a value of "1" indicates the next "1"-bit is on the next "line" succeeding the current one, whereas
  - a flag with a value of "0" indicates the next "1"-bit is on the same "line" as the current one;
- next k collections of 5 bits—in the (k+2)-th bit to the (6k+1)-th bit (e.g., 3rd-7th bits for k=1; 4th-8th & 9th-13th bits for k=2; 5th-9th, 10th-14th & 15th-19th bits for k=3; ...)—indicate the positions of at most k "1"-bits succeeding the long run of consecutive groups of 31 zeros; and
- suffix 31–6k bits (e.g., suffix 25 bits, 19 bits, 13 bits, ...) to indicate the number of these of consecutive groups of 31 zeros.

Observation 1. Observed from the above specification, when k=1, both IPLWAH(1) and M-PLWAH(1) produce the same compressed bitmap. However, further compression is observed for higher values of k (i.e., when $k > 1$).

Example 5. Continue with Example 4. With our M-IPLWAH(3) compressed model, the vector for Kees can be further compressed into a bitmap consisting of a sequence of five (zero-fill and literal) 32-bit words. Most of the words in this M-IPLWAH(3) compressed bitmap are identical or similar to those in the IPLWAH(3) compressed bitmap in Example 4. Specifically:

- the first, fourth and fifth words in this M-IPLWAH(3) are *identical* to the corresponding words in IPLWAH:
  - the first, seventh and eighth words in the IPLWAH(1) compressed bitmap; or,
  - the first, sixth and seventh words in the IPLWAH(3) compressed bitmap.

  It is because these words capture a long run of consecutive groups of 31 zeros followed by only a single "1"-bits (so that no flag bits need to be added):
- The second 32-bit word is *similar—but not identical*—to that of IPLWAH(3) due to the addition of the flag bits: 10 0 0 01011 10000 10101 0 0011 1111 1111, where (a) the suffix indicates 11 1111 1111 $_{(2)}$ = 1023 $_{(10)}$ groups of 31 consecutive zeros are followed by (b) a "1"-bit at position 1011 $_{(2)}$ = 11 $_{(10)}$ of a succeeding word representing Brechtje. Then, (c) a "0"-bit flag in the 3rd position indicates that (d) the next "1"-bit at position 10000 $_{(2)}$ = 16 $_{(10)}$ representing Cas is in the same word representing Brechtje. Furthermore, (e) another "0"-bit flag in the 4th position indicates that (f) the next "1"-bit at position 10101 $_{(2)}$ = 21 $_{(10)}$ representing Danique is in the same word representing Cas. In other words, all three followees are in the same word.
- The third 32-bit word in this M-IPLWAH(3) compressed bitmap is *different* because it is a compression of the third, fourth and fifth words in the IPLWAH(3): 10 1 1 11110 10100 01010 0 0011 1111 1111, where (a) the prefix "10" indicates that it is a zero-fill word, (b) the suffix 11 1110 1111 $_{(2)}$ = 1007 $_{(10)}$ indicates 1007 groups of 31 consecutive zeros are followed by (c) a "1"-bit at position 11110 $_{(2)}$ = 30 $_{(10)}$ of a succeeding word representing Evert. Then, (d) a "1"-bit flag in the 3rd position indicates that (e) the next "1"-bit at position 10100 $_{(2)}$ = 20 $_{(10)}$ representing Famke is in a word succeeding the word representing Evert. Furthermore, (f) another "1"-bit flag in the 4th position indicates that (g) the next "1"-bit at position 1010 $_{(2)}$ = 10 $_{(10)}$ representing Gerrit is in a word succeeding the word representing Famke.

In other words, this M-IPLWAH(3) compressed bitmap only requires 5x32 = 160 bits (cf. 224 bits for the IPLWAH(3) compressed bitmap, 256 bits for the IPLWAH(1) compressed bitmap, 384 bits for the WAH compressed bitmap, and 126,272 bits for the original uncompressed bit vector) to capture all followees of the follower Kees.

III. Our Gapped-Line Improved Position List Word-Aligned Hybrid (G-IPLWAH) Compression

Observed from our illustrative social network that the situation for three groups of followees—namely, (a) {Brechtje, Cas, Danique}, (b) {Evert, Famke, Gerrit} and (c) {Hannie, Ignaas}—are similar but not identical. Specifically, observed from Example 2 that the first group of followees are happened to be on the same literal word in the WAH compressed bitmap due to the very close proximity of their user ID numbers. Because of that, these three followees can be combined with their preceding run of consecutive groups of 31 zeros in the IPLWAH(3) compressed bitmap, and thus subsequent M-IPLWAH(k) compressed bitmap. In contrast, the second group of followees are not on the same literal word but on three consecutive literal words (with each word containing one followee). As such, although they are not compressed into a single word in the IPLWAH(k) compressed bitmap, they can be compressed into a single word. More precisely, they are compressed into a single word with their preceding run of consecutive groups of 31 zeros in the M-IPLWAH(3) compressed bitmap. As for the third group of followees, they are not even on two consecutive literal words. There is a gap of a single zero-fill word with only one group of 31 zeros in between. Consequently, they are not compressed into a words in the M-IPLWAH(k) compressed bitmap. So, a logical question is: Can the third group of followees be compressed?

Here, we response with a "yes" answer by designing a *gapped-line improved position list word-aligned hybrid* (*G-IPLWAH*) *compression model*. In this section, we describe our G-IPLWAH(k, g) compression model for social network analysis and mining. The idea is to combine:

- a zero-fill word (i.e., long run of consecutive zero groups of "0"), with
- its multiple consecutive succeeding (literal or zero-fill) words containing at most k "1"-bits (with at least one "1"-bit in each literal word) and may contain a small gap among these words.

A key difference between M-IPLWAH(k) and this G-IPLWAH(k, g) compression models is that, the former only combines consecutive multiple literal words (containing at most k "1"-bits) succeeding the zero-fill word, whereas the latter can combine consecutive multiple (literal or zero-fill) words (with at most k '1"-bits distributed among these literal words with at least one "1"-bit in each literal word)—and may contain gaps—succeeding the zero-fill word. For instance, G-IPLWAH(k, g) is expected to handle situations like that for followees Hannie and Ignaas, in which the user ID numbers between the two is more than 31, thus creating a gap with a zero-fill word consisting of only one group of consecutive zeros.

With our G-IPLWAH(k, g) compression model, the combined zero-fill word is still (a) represented as a 32-bit word and (b) having prefix "10" to indicate its identity (i.e., a zero-fill word). It uses the first 5 bits to indicate the position of the first "1"-bits succeeding the long run of consecutive groups of 31 zeros. It uses ($5+g$) bits to indicate the positions of the subsequent "1"-bits (for a total of at most k "1"-bits), and g indicates the additional span of these (literal or zero-fill) words to be combined into a single compressed zero-fill word. To elaborate, as we use ($5+g$) bits to represent the positions of "1"-bits, they can beyond the usual positions 1 to 31 into positions 1 to $2^{(5+g)}-1$. This allows us to combine multiple "lines" even with small gaps in between. Here, the flag bits from M-IPLWAH(k) is no longer needed, and thus saving (k–1) bits. As such, te number of the consecutive groups of 31 zeros is indicated by the suffix 25–(k–1)($5+g$) bits. Practically, ($1 \le k \le 3$) and ($0 \le g \le 2$) for our G-IPLWAH(k, g) compression model. To summarize, a zero-fill word in G-IPLWAH(k, g)—for ($1 \le k \le 3$) and ($0 \le g \le 2$) practically—is represented as a 32 bit word with:

- prefix "10" to indicate its identity (i.e., a zero-fill word);
- next 5 bits (i.e., 3rd-7th bits) indicate the positions of the first "1"-bits in a single group of 31 bits succeeding the long run of consecutive groups of 31 zeros
- next (k–1) collections of ($5+g$) bits—in the (k+7)-th bit to the (7+(k–1)($5+g$))-th bit (e.g., 8th-21st bits when k=3 and g=2)—indicate the positions of at most (k–1) "1"-bits succeeding the long run of consecutive groups of 31 zeros; and
- suffix 25–(k–1)($5+g$) bits (e.g., suffix 11 bits when k=3 and g=2) to indicate the number of these of consecutive groups of 31 zeros.

Observation 2. Observed from the above specification, when g=0, both IPLWAH(k) and G-PLWAH(k, 0) produce the same compressed bitmap. However, further compression is observed for higher values of g (i.e., when $g > 0$).

Example 6. Continue with Example 4. With our G-IPLWAH(3, 2) compressed model, the vector for Kees can be further compressed into a bitmap consisting of a sequence of four (zero-fill and literal) 32-bit words. Most of the words in this G-IPLWAH(3, 2) compressed bitmap are identical or similar to those in the IPLWAH(3) compressed bitmap in Example 4. Specifically:

- The first word in this G-IPLWAH(3, 2) is *identical* to the first word in IPLWAH(k).
- The second 32-bit word in this G-IPLWAH(3, 2) is *similar—but not identical*—to that of IPLWAH(3) due to bit size changes for indicating the positions of "1"-bits: 10 01011 0010000 0010101 011 1111 1111, where (a) the suffix (i.e., 11 bits from the 22nd-32nd bits) indicates $11\ 1111\ 1111_{(2)} = 1023_{(10)}$ groups of 31 consecutive zeros are followed by (b) three "1"-bits at positions $1011_{(2)} = 11_{(10)}$, $10000_{(2)} = 16_{(10)}$ and $10101_{(2)} = 21_{(10)}$ of a succeeding word representing Brechtje, Cas and Danique. Here, the position of the first "1"-bit is captured by 5 bits (i.e., 3rd-7th bits), whereas those of the second and third "1"-bits are captured by $5+g = 7$ bits (i.e., 8th-14th and 15th-21st bits).
- The third 32-bit word in this G-IPLWAH(3, 2) is *different* because it is a compression of the third, fourth and fifth words in the IPLWAH(3) compressed bitmap: 10 11110 0110011 1001000 011 1111 1111, where (a) the suffix indicates $11\ 1110\ 1111_{(2)} = 1007_{(10)}$ groups of 31 consecutive zeros are followed by (b) three "1"-

bits at positions $11110_{(2)} = 30_{(10)}$, $110011_{(2)} = 51_{(10)}$ and $1001000_{(2)} = 72_{(10)}$ of an "extended" succeeding word representing Evert, Famke and Gerrit. Here, the position of the first "1"-bit is captured by 5 bits (i.e., 3rd-7th bits). Those of the second and third "1"-bits are captured by $5+g = 7$ bits (i.e., 8th-14th and 15th-21st bits), which allow us to have positions beyond the usual position 31 (up to position 127), and thus enable us to capture these three followees spanning over multiple (precisely, up to 4) "lines" (cf. Brechtje, Cas and Danique who span on the same "line" or the same literal word).

- The fourth 32-bit word in this G-IPLWAH(3, 2) sequence is also *different* because it is a compression of the sixth and seventh words in the IPLWAH(3): 10 11001 0 0000 0000 0000 0011 1111 1001, where (a) the suffix indicates $11\ 1111\ 1001_{(2)} = 1017_{(10)}$ groups of 31 consecutive zeros are followed by (b) two "1"-bits at positions $11001_{(2)} = 25_{(10)}$ and $1101011_{(2)} = 107_{(10)}$ of an "extended" succeeding word representing Hannie and Ignaas. Here, the position of the first "1"-bit is captured by 5 bits (i.e., 3rd-7th bits). That of the second "1"-bit is captured by $5+g = 7$ bits (i.e., 8th-14th bits), which enable us to capture these two followees spanning over three "lines" with gaps (i.e., the "1"-bit for Hannie is in one "line", that for Ignaas is in the third "line", and no "1"-bit in the second "line" between them. This is quite different from the "1"-bits for Evert, Famke and Gerrit who span on three consecutive "line" or literal word with at least one "1"-bit in each "line").

In other words, this G-IPLWAH(3, 2) compressed bitmap only requires 4x32 = 128 bits (cf. 160 bits for M-IPLWAH(3) compressed bitmap, 224 bits for the IPLWAH(3) compressed bitmap, 256 bits for the IPLWAH(1) compressed bitmap, 384 bits for the WAH compressed bitmap, and 126,272 bits for the original uncompressed bit vector) to capture all followees of the follower Kees.

IV. Evaluation

In terms of functionality, the WAH compression model aims to compress consecutive groups of 31 zeros. The IPLWAH(k) model combines a few (say, up to k=5 practically) "1"-bits appear on the same "line" (i.e., literal word) succeeding run of consecutive groups of 31 zeros. The M-PLWAH(k) model combines a few (say, up to k=4 practically) "1"-bits appear on consecutive "lines" (without gaps) succeeding run of consecutive groups of 31 zeros. The G-PLWAH(k, g) model further combines a few (say, up to k=3 practically) "1"-bits appear on multiple "lines" (with may have gaps) succeeding run of consecutive groups of 31 zeros. See Table I.

In terms of memory and runtime, evaluation on datasets from SNAP Stanford Large Network Collection (e.g., ego-Gplus, ego-Twitter) show that G-PLWAH(k, g) further compresses the data, and thus requires less space, than related works. Consequently, evaluation results also show a reduction in runtime for social network analysis and mining (e.g., mining groups of frequently followed social entities/followees).

TABLE I. Functionality of Compression Models

Handle "1"-bits following 0s	WAH	IPLWAH	M-IPLWAH	G-IPLWAH
on same "line"	x	✓	✓	✓
over multi-lines w/o gaps	x	x	✓	✓
over multi-lines w/ gaps	x	x	x	✓

Conclusions

In this paper, we presented a gapped gapped-line improved position list word-aligned hybrid (G-IPLWAH) compression for social network analysis and mining for very sparse but big social data. The G-IPLWAH(k, g) extends the number of bits to represent positions of "1"-bits beyond the usual 31 bits. This allows us to capture "1"-bits spanning over multiple "lines" (i.e., literal words) even with gaps. As a logical continuation along with the directions on compression for sparse social networks over the past few IEEE/ACM ASONAM, this compression model is more flexible and further compresses social data than previous ones. As ongoing and future work, we explore further compression for effective social network analysis and mining..

Acknowledgment

This work is partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), as well as the University of Manitoba.

References

[1] M. Day, et al., "AI robo-advisor with big data analytics for financial services," in IEEE/ACM ASONAM 2018, pp. 1027-1031.

[2] M. Kaya, et al. (eds.), Social Network Based Big Data Analysis and Applications, 2018.

[3] A. Kobusinska, et al., "Emerging trends, issues and challenges in Internet of Things, big data and cloud computing," FGCS 87, 2018, pp. 416-419.

[4] C.K. Leung, "Big data analysis and mining," in Encyclopedia of Information Science and Technology, 4e, 2018, pp. 338-348.

[5] C.K. Leung, "Big data computing and mining in a smart world," in Big Data Analyses, Services, and Smart Data, 2021, pp. 15-27.

[6] K.F. Xylogiannopoulos, et al., "Frequent and non-frequent pattern detection in big data streams: an experimental simulation in 1 trillion data points," in IEEE/ACM ASONAM 2016, pp. 931-938.

[7] F. Jiang, C.K. Leung, "A data analytic algorithm for managing, querying, and processing uncertain big data in cloud environments," Algorithms 8(4), 2015, pp. 1175-1194.

[8] C.K. Leung, "Uncertain frequent pattern mining," in Frequent Pattern Mining, 2014, pp. 417-453.

[9] C. Zhang, O.R. Zaïane, "Detecting local communities in networks with edge uncertainty," in IEEE/ACM ASONAM 2018, pp. 9-16.

[10] J. de Guia, et al., "DeepGx: deep learning using gene expression for cancer classification," in IEEE/ACM ASONAM 2019, pp. 913-920.

[11] C.K. Leung, et al., "Predictive analytics on genomic data with high-performance computing," in IEEE BIBM 2020, pp. 2187-2194.

[12] T. Pawliszak, et al, "Operon-based approach for the inference of rRNA and tRNA evolutionary histories in bacteria," BMC Genomics 21 (Supplement 2), 2020, pp. 252:1-252:14.

[13] O.A. Sarumi, C.K. Leung, "Exploiting anti-monotonic constraints for mining palindromic motifs from big genomic data," in IEEE BigData 2019, pp. 4864-4873.

[14] S.D. Bernhard, et al., "Clickstream prediction using sequential stream mining techniques with Markov chains," in IDEAS 2016, pp. 24-33.

[15] K.F. Xylogiannopoulos, et al., "Clickstream analytics: an experimental analysis of the amazon users' simulated monthly traffic," in IEEE/ACM ASONAM 2018, pp. 841-848.

[16] D. Choudhery, C.K. Leung, "Social media mining: prediction of box office revenue," in IDEAS 2017, pp. 20-29.

[17] J.A. Brown, et al., "A machine learning system for supporting advanced knowledge discovery from chess game data," in IEEE ICMLA 2017, pp. 649-654.

[18] V. Cedeno-Mieles, et al., "Mechanistic and data-driven agent-based models to explain human behavior in online networked group anagram games," in IEEE/ACM ASONAM 2019, pp. 357-364.

[19] K.E. Barkwell, et al., "Big data visualisation and visual analytics for music data mining," in IV 2018, pp. 235-240.

[20] C. Dhahri, et al., "Mood-aware music recommendation via adaptive song embedding," in IEEE/ACM ASONAM 2018, pp. 135-138.

[21] C. Fan, et al., "Social network mining for recommendation of friends based on music interests," in IEEE/ACM ASONAM 2018, pp. 833-840.

[22] M. Ahmed, et al., "Anomaly detection on big data in financial markets," in IEEE/ACM ASONAM 2017, pp. 998-1001.

[23] C.K. Leung, et al., "A machine learning approach for stock price prediction," in IDEAS 2014, pp. 274-277.

[24] K.J. Morris, et al, "Token-based adaptive time-series prediction by ensembling linear and non-linear estimators: a machine learning approach for predictive analytics on big stock data," in IEEE ICMLA 2018, pp. 1486-1491.

[25] C.K. Leung, et al., "Data science for healthcare predictive analytics," in IDEAS 2020, pp. 8:1-8:10.

[26] J. Souza, et al., "An innovative big data predictive analytics framework over hybrid big data sources with an application for disease analytics," in AINA 2020, pp. 669-680.

[27] H. Vural, et al., "A model based on random walk with restart to predict circRNA-disease associations on heterogeneous network," in IEEE/ACM ASONAM 2019, pp. 929-932.

[28] Y. Chen, et al., "Temporal data analytics on COVID-19 data with ubiquitous computing," in IEEE ISPA-BDCloud-SocialCom-SustainCom 2020, pp. 958-965. doi: 10.1109/ISPA-BDCloud-SocialCom-SustainCom51426.2020.00146

[29] P. Gupta, et al., "Vertical data mining from relational data and its application to COVID-19 data," in Big Data Analyses, Services, and Smart Data, 2021, pp. 106-116.

[30] C.K. Leung, et al., "Big data science on COVID-19 data," in IEEE BigDataSE 2020, pp. 14-21. doi: 10.1109/BigDataSE50710.2020.00010

[31] C.K. Leung, et al., "Big data visualization and visual analytics of COVID-19 data," in IV 2020, pp. 415-420. doi: 10.1109/IV51561.2020.00073

[32] C.K. Leung, et al., "Machine learning and OLAP on big COVID-19 data," in IEEE BigData 2020, pp. 5118-5127.

[33] Q. Liu, et al., "A two-dimensional sparse matrix profile DenseNet for COVID-19 diagnosis using chest CT images," IEEE Access 8, 2020, pp. 213718-213728.

[34] A.A. Audu, et al., "An intelligent predictive analytics system for transportation analytics on open data towards the development of a smart city," in CISIS 2019, pp. 224-236.

[35] P.P.F. Balbin, et al., "Predictive analytics on open big data for supporting smart transportation services," Procedia Computer Science 176, 2020, pp. 3009-3018.

[36] C.K. Leung, et al., "An innovative fuzzy logic-based machine learning algorithm for supporting predictive analytics on big transportation data," in FUZZ-IEEE 2020. doi: 10.1109/FUZZ48607.2020.9177823

[37] C.K. Leung, et al., "Data mining on open public transit data for transportation analytics during pre-COVID-19 era and COVID-19 era," in INCoS 2020, pp. 133-144.

[38] C.K. Leung, et al., "Effective classification of ground transportation modes for urban data mining in smart cities," in DaWaK 2018, pp. 83-97.

[39] C.K. Leung, et al., "Urban analytics of big transportation data for supporting smart cities," in DaWaK 2019, pp. 24-33.

[40] L. Idan, J. Feigenbaum, "Show me your friends, and I will tell you whom you vote for: predicting voting behavior in social networks," in IEEE/ACM ASONAM 2019, pp. 816-824.

[41] F. Jiang, et al., "Finding popular friends in social networks," in CGC 2012, pp. 501-508.

[42] C.K. Leung, et al., "Big data analytics of social network data: who cares most about you on Facebook?" in Highlighting the Importance of Big Data Management and Analysis for Various Applications, 2018, pp. 1-15.

[43] M. Mai, et al., "Big data analytics of Twitter data and its application for physician assistants: who is talking about your profession in Twitter?" in Data Management and Analysis, 2020, pp. 17-32.

[44] C.K. Leung, et al., "Visual analytics of social networks: mining and visualizing co-authorship networks," in HCII-FAC 2011, pp. 335-345.

[45] R. Molontay, M. Nagy, "Two decades of network science: as seen through the co-authorship network of network scientists," in ," in IEEE/ACM ASONAM 2019, pp. 578-583.

[46] C.K. Leung, et al., "Mining 'following' patterns from big but sparsely distributed social network data," in IEEE/ACM ASONAM 2018, pp. 916-919.

[47] K.E. Dierckens, et al., "A data science and engineering solution for fast k-means clustering of big data," in IEEE TrustCom-BigDataSE-ICESS 2017, pp. 925-932.

[48] C.K. Leung, F. Jiang, "A data science solution for mining interesting patterns from uncertain big data," in IEEE BDCloud 2014, pp. 235-242.

[49] A.K. Chanda, et al., "A new framework for mining weighted periodic patterns in time series databases," ESWA 79, 2017, pp. 207-224.

[50] A. Fariha, et al., "Mining frequent patterns from human interactions in meetings using directed acyclic graphs," in PAKDD 2013, Part I, pp. 38-49.

[51] F. Jiang, et al., "Big data mining of social networks for friend recommendation," in IEEE/ACM ASONAM 2016, pp. 921-922.

[52] C.K. Leung, C.L. Carmichael, "FpVAT: A visual analytic tool for supporting frequent pattern mining," ACM SIGKDD Explorations 11(2), 2009, pp. 39-48.

[53] K.F. Xylogiannopoulos, et al., "Text mining for malware classification using multivariate all repeated patterns detection," in IEEE/ACM ASONAM 2019, pp. 887-894.

[54] S. Ahn, et al., "A fuzzy logic based machine learning tool for supporting big data business analytics in complex artificial intelligence environments," in FUZZ-IEEE 2019, pp. 1259-1264.

[55] C.K. Leung, "Mathematical model for propagation of influence in a social network," in Encyclopedia of Social Network Analysis and Mining, 2e, 2018, pp. 1261-1269.

[56] Y. Cao, et al., "Hybrid deep learning model assisted data compression and classification for efficient data delivery in mobile health applications," IEEE Access 8, 2020, pp. 94757-94766.

[57] H. Jiang, S. Lin, "A rolling hash algorithm and the implementation to LZ4 data compression," IEEE Access 8, 2020, pp. 35529-35534.

[58] H. Fu, et al., "An extended hybrid image compression based on soft-to-hard quantification," IEEE Access 8, 2020, pp. 95832-95842.

[59] K.S. Kumar, et al., "Efficient video compression and improving quality of video in communication for computer encoding applications," Comput. Commun. 153, 2020, pp. 152-158.

[60] S.M. Hossein, et al., "DNA sequences compression by GP^2 R and selective encryption using modified RSA technique," IEEE Access 8, 2020, pp. 76880-76895.

[61] C.K. Leung, et al., "Mining 'following' patterns from big sparse social networks," in IEEE/ACM ASONAM 2016, pp. 923-930.

[62] C.K. Leung, F. Jiang, "Efficient mining of 'following' patterns from very big but sparse social networks," in IEEE/ACM ASONAM 2017, pp. 1025-1032.

[63] C.K. Leung, et al., "Flexible compression of big data," in IEEE/ACM ASONAM 2019, pp. 741-748.