ZooBP: Belief Propagation for Heterogeneous Networks

Dhivya Eswaran
Carnegie Mellon University
deswaran@cs.cmu.edu

Stephan Günnemann
Technical University of Munich
guennemann@in.tum.de

Christos Faloutsos
Carnegie Mellon University
christos@cs.cmu.edu

Disha Makhija
Flipkart, India
disha.makhiji@flipkart.com

Mohit Kumar
Flipkart, India
k.mohit@flipkart.com

ABSTRACT

Given a heterogeneous network, with nodes of different types – e.g., products, users and sellers from an online recommendation site like Amazon – and labels for a few nodes ('honest', 'suspicious', etc), can we find a closed formula for Belief Propagation (BP), exact or approximate? Can we say whether it will converge?

BP, traditionally an inference algorithm for graphical models, exploits so-called "network effects" to perform graph classification tasks when labels for a subset of nodes are provided; and it has been successful in numerous settings like fraudulent entity detection in online retailers and classification in social networks. However, it does not have a closed-form nor does it provide convergence guarantees in general. We propose ZooBP, a method to perform fast BP on undirected heterogeneous graphs with provable convergence guarantees. ZooBP has the following advantages: (1) *Generality*: It works on heterogeneous graphs with multiple types of nodes and edges; (2) *Closed-form solution*: ZooBP gives a closed-form solution as well as convergence guarantees; (3) *Scalability*: ZooBP is linear on the graph size and is up to 600× *faster* than BP, running on graphs with *3.3 million edges* in a few seconds. (4) *Effectiveness*: Applied on real data (a Flipkart e-commerce network with users, products and sellers), ZooBP identifies fraudulent users with a near-perfect precision of *92.3 %* over the top 300 results.

1. INTRODUCTION

Suppose we are given users, software products, reviews ('likes') and manufacturers; and that we know there are two types of users (honest, dishonest), three types of products (high-quality-safe, low-quality-safe, malware), and two types of sellers (malware, non-malware). Suppose that we also know that user 'Smith' is 'honest', while seller 'evil-dev' sells malware. Given this information, the BP algorithm allows us to infer the types of all other nodes – but will it converge? Can we have a closed formula for the beliefs of all nodes in the above setting?

Proceedings of the VLDB Endowment, Vol. 10, No. 5
Copyright 2017 VLDB Endowment 2150-8097/17/01.

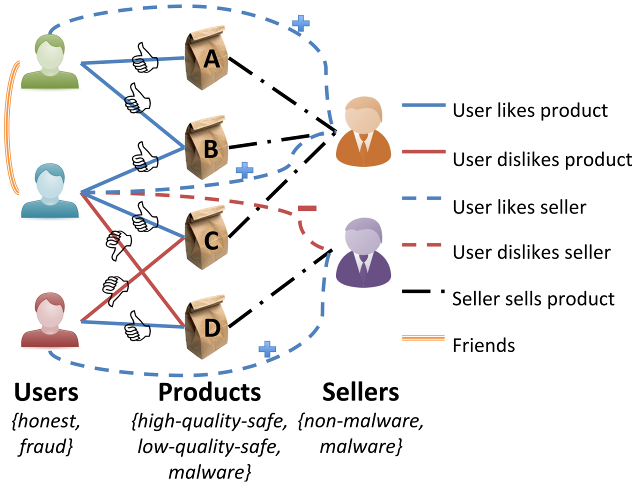


Figure 1: ZooBP is very *general* and can handle any undirected, weighted, heterogeneous multi graph

The generic problem for BP is informally given by:

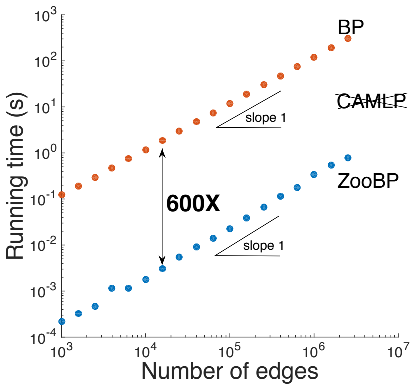
Informal Problem 1 (General BP)**. Given**
- *a large heterogeneous graph (as, e.g., in Figure 1),*
- *for each node type, a set of classes (labels) (e.g. honest/dishonest for user nodes)*
- *the compatibility matrices for each edge-type, indicating the affinity between the nodes' classes (labels)*
- *initial beliefs about a node's class (label) for a few nodes in the graph*

Find *the most probable class (label) for each node.*

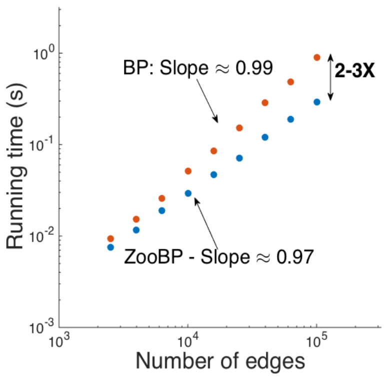
This problem is found in many other scenarios besides the above mentioned one: in a health-insurance fraud setting, for example, we could have patients (honest or accomplices), doctors (honest or corrupt) and insurance claims (low or expensive or bogus). The textbook solution to this problem is loopy Belief Propagation (BP, in short) – an iterative message-passing algorithm, which, in general, offers no convergence guarantees.

Informal Problem 2 (BP- Closed-form solution)**. Given** *a setting like the general BP.* **Find** *an accurate, closed-form solution for the final beliefs.*

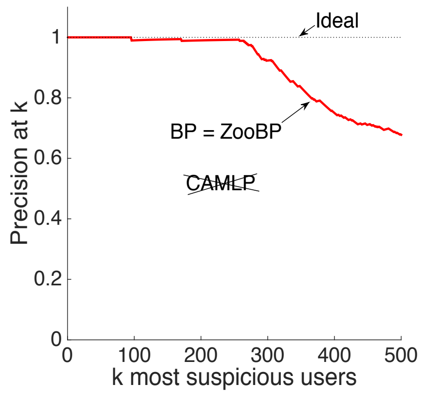
Here, we show how to derive a closed-form solution (Theorem 1) that almost perfectly approximates the result of



(a) Scalability & speed (MATLAB)



(b) Scalability & speed (C++)



(c) Effectiveness

Figure 2: ZooBP is (a), (b) *fast* up to 600 times depending on platform; (c) *effective* for fraud detection on FLIPKART-(3,5) data with 3 node types and 5 edge types. Competitors such as CAMLP are not applicable to this scenario.

loopy BP, using well-understood highly optimized matrix operations and with provable convergence properties. The contributions of our work are:

- **Generality**: ZooBP works on any undirected weighted heterogeneous graph with multiple edge types. Moreover, it trivially includes previous results – FaBP [15] and LinBP [12] – as special cases.
- **Closed-form solution**: Thanks to our closed-form solution (Theorem 1), we know when our method will converge (Theorem 2).
- **Scalability**: ZooBP is linear on the input size and it matches or outperforms BP with up to 600× speed-up (Figure 2a), requiring a few seconds on a stock machine, for million-scale graphs.
- **Effectiveness**: On real data (product and seller reviews from FLIPKART), ZooBP achieved precision of *92.3 %* in the top 300 most suspicious users (Figure 2c).

We want to emphasize that the dramatic 600× savings are largely thanks to our closed formula: MATLAB is very inefficient in handling loops, but there is no other choice with the traditional BP equations (Eq. 3). With ZooBP, however, we can replace the loops with a matrix equation (Theorem 1) and this allows the use of all the highly optimized matrix algorithms resulting in dramatic speed-ups. Comparisons of C++ implementations (Figure 2b) show that ZooBP never loses to BP; and it usually wins by a factor of 2× to 3×, depending on the relative speed of additions, multiplications, and function calls (logarithms) for the given machine.

Reproducibility: Our code is open-sourced at `http://www.cs.cmu.edu/ deswaran/code/zoobp.zip` .

2. RELATED WORK

In this section, we review related works on belief propagation and summarize prior attempts on linearization.

Propagation in Networks. Exploiting network effects improves accuracy in numerous classification tasks [14, 18]. Such methods include random walk with restarts [24], semi-supervised learning [5], label propagation [27] and belief propagation [20]. Unlike BP, most of the proposed techniques operate on simple unipartite networks only (even though more complex graphs are omnipresent [3]) or they do not extend to scenarios of heterophily; hence we mainly focus on BP in this work.

Belief Propagation. Belief Propagation [20] is an efficient inference algorithm in graphical models, which works by iteratively propagating network effects. However, there is no closed formula for its solution and it is not guaranteed to converge unless the graph has no loops [21] or on a few other special cases [16]. Nevertheless, loopy BP works well in practice [17] and it has been successfully applied to numerous settings such as error-correcting codes [10, 9], stereo matching in computer vision [23, 8], fraud detection [19, 1] and interactive graph exploration [6]. The success of BP has increased the interest to approximate BP and to find closed-form solutions in specialized settings.

Approximation Attempts. Koutra et. al. [15] provide a linearized approximation of BP for *unipartite graphs with two classes*. and Gatterbauer et. al. [12] extended it to multiple classes. Gatterbauer [11] attempted to extend this even further to $|T|$*-partite networks*. None of the above can handle a general heterogeneous graph with multiple types of nodes and edges. Even the most general formulation above [11] is limited to single edge type between two node types and hence cannot handle real world scenarios where edges naturally have a polarity (e.g., product-rating networks). In addition, edges between nodes of the same type cannot be handled(e.g., *friendship* edges). Furthermore, [11] neither provides a scalable implementation[1] nor easy-to-compute convergence conditions. Independently, Yamaguchi et al [25] used the degree of a node as a measure the confidence of belief to linearize BP in a completely new way. However, this also assumes a case of unipartite graphs with a single edge type.

Finally, we note that our notion of the term *residual* differs from that of Residual Belief Propagation (RBP) [7]. RBP calculates residuals based on the difference in messages in the successive iterations while our residual beliefs and messages are deviations from their centered values (as we will demonstrate shortly). Our goals are also different: RBP uses residuals to derive an efficient asynchronous BP schedule whereas our interest is in linearizing BP in a heterogeneous setting and providing precise convergence guarantees.

[1]This is non-trivial – naively solving for beliefs directly from Theorem 1 leads to an algorithm quadratic in graph size as $(\mathbf{I} - \mathbf{P} + \mathbf{Q})^{-1}$ is a dense matrix.

In summary, as shown in Table 1, none of competitors satisfy all properties that ZooBP satisfies.

Table 1: Contrasting ZooBP against previous methods

Property	BP [26], RBP [7]	FaBP [15]	LinBP [12]	CAMLP [25]	[11]	ZooBP
> 2 classes	✓		✓	✓	✓	✓
Node heterogeneity	✓				✓	✓
Unrestricted edge types	✓					✓
Closed-form solution		✓	✓	✓	✓	✓
Convergence guarantees		✓	✓	✓	✓	✓
Scalable implementation	✓	✓	✓	✓		✓

3. PRELIMINARIES

We will first provide mathematical definitions and results that we will use in our derivation and then introduce the basic framework of Belief Propagation. We will follow the notation given in Table 2.

Definition 1 (Constant-margin matrix)**.** *A $p \times q$ matrix is said to be constant-margin of scale α if each row sums to $q\alpha$ and each column sums to $p\alpha$.*

Definition 2 (Matrix Vectorization [13])**.** *Vectorization of an $m \times n$ matrix converts it into a $mn \times 1$ vector given by:*

$$vec(\mathbf{X}) = [\mathbf{x}_1^T \dots \mathbf{x}_n^T]^T$$

where $\mathbf{x}_i$ denotes the i^{th} column vector of matrix $\mathbf{X}$.

Definition 3 (Kronecker product [13])**.** *The Kronecker product of two matrices $\mathbf{X}_{m\times n}$ and $\mathbf{Y}_{p\times q}$ is the $mp \times nq$ matrix given by:*

$$\mathbf{X} \otimes \mathbf{Y} = \begin{bmatrix} X(1,1)\mathbf{Y} & X(1,2)\mathbf{Y} & \dots & X(1,n)\mathbf{Y} \\ X(2,1)\mathbf{Y} & X(2,2)\mathbf{Y} & \dots & X(2,n)\mathbf{Y} \\ \vdots & \vdots & \ddots & \vdots \\ X(m,1)\mathbf{Y} & X(m,2)\mathbf{Y} & \dots & X(m,n)\mathbf{Y} \end{bmatrix}$$

Definition 4 (Centered Matrix/Vector)**.** *A matrix or a vector is said to be c-centered if the average of all its elements is c and the maximal deviation from c is small in magnitude when compared to c.*

Definition 5 (Residual vector/matrix)**.** *A 0-centered vector/matrix is termed as residual. The residual of a c-centered vector or matrix is obtained by subtracting c from each of its elements.*

Table 2: Notation

Entity/Operator	Notation
Scalar	small or capital, italics; e.g., S, k_s
Vector	bold, small; e.g., $\mathbf{b}_u, \mathbf{m}_{uv}$
Matrix	bold, capital; e.g., $\mathbf{B}_s, \mathbf{Q}$
Vectorization	$vec(.)$
Set/Multiset	calligraphic, capital; e.g., $\mathcal{S}$
Kronecker product	$\otimes$
Direct sum of matrices	$\bigoplus$
Vector/matrix entry	Not bold; e.g., $b_u(i), H(i,j)$
Spectral radius	$\rho(.)$

Example 1. $\mathbf{X} = \begin{bmatrix} 2.8 & 3 & 3.2 \\ 3.2 & 3 & 2.8 \end{bmatrix}$ *is a constant-margin matrix of scale 3. It is also a 3-centered matrix, as the maximal deviation 0.2 is small compared to the overall matrix average 3. Its residual is* $\begin{bmatrix} -0.2 & 0 & 0.2 \\ 0.2 & 0 & -0.2 \end{bmatrix}$.

Lemma 1 (Roth's column lemma [13])**.** *For any three matrices $\mathbf{X}, \mathbf{Y}$ and $\mathbf{Z}$,*

$$vec(\mathbf{XYZ}) \quad = \quad (\mathbf{Z}^T \otimes \mathbf{X}) vec(\mathbf{Y}) \qquad (1)$$

Definition 6 (Matrix Direct Sum [2])**.** *The matrix direct sum of n square matrices $\mathbf{A}_1, \dots, \mathbf{A}_n$ is the block diagonal matrix given by $\bigoplus_{i=1}^n \mathbf{A}_i = diag(\mathbf{A}_1, \dots, \mathbf{A}_n)$.*

3.1 Belief Propagation

Belief propagation, also known as sum-product message passing, is a technique to perform approximate inference in graphical models. The algorithm starts with prior beliefs for a certain subset of nodes in a graph (e.g., $\mathring{e}_u$; prior knowledge about u's class) and then sequentially propagates from one node (say, u) to another (v) a message ($\mathring{m}_{uv}$) which represents u's belief about v's class. This process is carried out until a steady state is reached (assuming convergence). After the sequential update process, the final beliefs about a node's class (e.g., $\mathring{b}_u$; the inferred information about the class of u) are recovered from the messages that a node receives.

Eq. (2) and Eq. (3) give the precise updates of the BP algorithm as given by Yedidia [26].

$$\mathring{b}_u(i) \quad \leftarrow \quad \frac{1}{Z_u} \mathring{e}_u(i) \prod_{v \in \mathcal{N}(u)} \mathring{m}_{vu}(i) \qquad (2)$$

$$\mathring{m}_{vu}(i) \quad \leftarrow \quad \frac{1}{Z_{vu}} \sum_j \phi(i,j)\ \mathring{e}_v(j) \prod_{w \in \mathcal{N}(v) \backslash u} \mathring{m}_{wv}(j) \qquad (3)$$

In every step of BP, the message that a node sends to another (Eq. (3)) is computed as the product of the messages it has received from all its neighbors except the recipient itself (echo-cancellation[2]), modulated by the *discrete potential function* $\phi(i,j)$. We let $\phi(i,j)$ be the conditional probability $\mathbb{P}(i|j)$ of class i on node u given class j on node v to facilitate probabilistic interpretation. This value is computed using an *edge compatibility matrix* or *edge-potential matrix* $\mathring{\mathbf{H}}$ as: $\mathbb{P}(i|j) = \mathring{H}(i,j)/\sum_g \mathring{H}(g,j)$. The edge compatibility matrix captures the affinity of classes, i.e., the higher or more positive the value of $\mathring{H}(i,j)$ relative to other entries, the more probable that a node with class i influences its neighbor to have class j, and vice versa. A numerical example to understand compatibility matrix follows.

Example 2. *We have a graph on readers and news articles with edges indicating "who reads what". The edge compatibility matrix is given by Table 3. Due to the higher value of $\mathring{H}$(Republican, Conservative), a Republican reader is likely to pass the message that the news articles he reads are conservative. Observe that our compatibility matrix is neither square nor doubly stochastic but is constant-margin. This is*

[2]This term prevents sending the same message that was received in the previous iteration along the same edge. It helps prevent two (or more) nodes mutually reinforcing each others' beliefs.

Table 3: Edge compatibility matrix $\overset{\circ}{\mathbf{H}}$ for Example 2.

↓ People/ Articles →	Conservative	Progressive	Neutral
Republican	0.367	0.300	0.333
Democrat	0.300	0.367	0.333

intentional – we would only be dealing with constant-margin compatibility matrices in our work.

Finally, the normalization constants Z_{vu} and Z_u in Eq. (2) and (3) respectively ensure that the beliefs and messages sum up to a constant (typically 1) at any iteration.

4. PROPOSED METHOD: ZOOBP

The goal of our work is to provide a closed-form solution to BP in arbitrary heterogeneous graphs using an intuitive principle for approximating the beliefs of nodes. The core idea is to derive a system of linear equations for beliefs which can be solved using matrix algebra to finally calculate all node beliefs in a single step of matrix operations. To do this, ZooBP borrows the basic framework of using residual compatibility matrix, beliefs and messages $(\mathbf{H}, \mathbf{b}_u, \mathbf{m}_{vu})$ instead of their non-residual counterparts $(\overset{\circ}{\mathbf{H}}, \overset{\circ}{\mathbf{b}}_u, \overset{\circ}{\mathbf{m}}_{vu})$ as in Eq. (2) and Eq. (3) from [15, 12].

We now describe our problem setting more formally before stating our main (and most generic) results.

4.1 Notation and Problem Description

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected heterogeneous graph on a collection of node types $\mathcal{S}$ and edge types $\mathcal{T}$ such that

$$\mathcal{V} = \bigcup_{s\in\mathcal{S}} \mathcal{V}_s \quad ; \quad \mathcal{E} = \bigcup_{t\in\mathcal{T}} \mathcal{E}_t$$

where $\mathcal{V}_s$ denotes the set of nodes of type s and $\mathcal{E}_t$ the multiset of edges of type t (i.e., parallel/multiple edges are allowed). Each node (edge) has a single node (edge) type. A node's type determines the set of classes it can belong to. Let us use k_s to denote the number of classes a type-s node can belong to.

Without loss of generality, we assume that an edge type can only connect a particular pair of node types (possibly a self pair, e.g., *friendship* edge). For example, in Figure 1, we have separate types of edges, one for "user likes product" and another for "user likes seller", instead of having a single edge type called "like" which connects users with both products and sellers. Observe that this is not a restrictive assumption, as we could always partition a complex edge type ("like") into simpler edge types obeying this condition.

Also, note the subtle generality in our notation – we have used a separate identifier $\mathcal{E}_t$ for edges of type t, instead of referring to it through the pair of node types it connects, $\mathcal{T}_{ss'}$. This allows us to have multiple types of edges connecting the same pair of node types – in Figure 1, we have both "user likes product" and "user dislikes product" edges connecting users and products; and possibly we can have multiple types of edges connecting the same pair of nodes as well (for example, a user initially *likes* a product but later *dislikes* it).

Let $\overset{\circ}{\mathbf{H}}_t$ and $\mathbf{A}_t$ denote the compatibility matrix and adjacency matrix for an edge type $t \in \mathcal{T}$. If t connects nodes of type s to type s', then $\mathbf{A}_t$ is a $n_s \times n_{s'}$ matrix where each row corresponds to a node of type s and each column corresponds to a node of type s'. Similarly, $\overset{\circ}{\mathbf{H}}_t$ is a $k_s \times k_{s'}$ matrix with rows denoting classes of type-s nodes and columns denoting classes of type-s' nodes.

Further, let $\mathcal{T}_{ss'}$ denote the set of edge types connecting node types s and s' and $\mathcal{T}_{uv}$ denote the multiset of edge types (to account for parallel/multiple edges of the same type) connecting nodes u and v. (see Table 5).

The problem then is to conduct *transductive inference* [4] on this graph, i.e., given initial beliefs for a subset of the nodes, to infer the most probably class for every node in the graph.

4.2 Key Insights

4.2.1 Residual compatibility matrices

In the case of undirected unipartite graphs, the compatibility matrix $\overset{\circ}{\mathbf{H}}_t$ turns out to be square and symmetrical; by further assuming that it was doubly stochastic, BP was successfully linearized [12]. However, in the general case, the compatibility matrices may not be square, let alone doubly stochastic. One core question is: *What kind of compatibility matrices allow a linearization of BP in this general setting?*

We first note that due to the normalization constants in Eq. (2) and Eq. (3), the overall scale of the compatibility matrices has no effect on the belief updates. Thus, w.l.o.g., we can fix the scale to be 1 – or using the notion from above, we can focus on 1-centered matrices. Thus, each compatibility matrix can be expressed as

$$\overset{\circ}{\mathbf{H}}_t = \mathbf{1} + \epsilon_t \mathbf{H}_t \tag{4}$$

where $\mathbf{1}$ is a matrix having the same dimension as $\overset{\circ}{\mathbf{H}}_t$ with all of its entries as 1 and $\mathbf{H}_t$ is the *residual compatibility matrix.* Here, ϵ_t can be viewed as the absolute strength of interaction through an edge of type t whereas $\mathbf{H}_t$ indicates the relative affinities of a pair of labels on either side of a type-t edge. Operating with the residual matrix allows us later on to derive an approximation of the BP equations.

The key insight for our result is to focus on compatibility matrices $\overset{\circ}{\mathbf{H}}_t$ that are *constant-margin.* Using this property, it follows that the residual $\mathbf{H}_t$ fulfills:

$$\sum_i \mathbf{H}_t(i,j) = \sum_j \mathbf{H}_t(i,j) = 0 \quad \forall \ \ t,i,j \tag{5}$$

Although the constant-margin constraint decreases the number of free parameters for a $p \times q$ compatibility matrix from $pq - 1$ (excluding scale) to $pq - p - q + 1$ (constraining row and column sums to be equal), we note that the set of constant-margin compatibility matrices is sufficiently expressive to model numerous real-world scenarios e.g., fraud detection in e-commerce networks [1], blog/citation networks and social networks [25].

To illustrate this, we derive the residual $\mathbf{H}$ for $\overset{\circ}{\mathbf{H}}$ (after re-scaling) given in Table 3.

$$\overset{\circ}{\mathbf{H}} \ = \ \begin{bmatrix} 1.1 & 0.9 & 1 \\ 0.9 & 1.1 & 1 \end{bmatrix} = \mathbf{1} + \overbrace{0.2}^{\epsilon} \overbrace{\begin{bmatrix} 0.5 & -0.5 & 0 \\ -0.5 & 0.5 & 0 \end{bmatrix}}^{\mathbf{H}}$$

In the above example (and in the rest of our work), we fix the scale of the residual $\mathbf{H}$ by holding its largest singular value at 1. This allows us to determine a unique $(\epsilon_t, \mathbf{H}_t)$ pair for a given constant-margin compatibility matrix.

4.2.2 *Residual beliefs and messages*

Similar to the original (non-residual) compatibility matrix $\mathring{\mathbf{H}}_t$ and its residual counterpart $\mathbf{H}_t$, we also introduce the notions for residuals of beliefs and messages. Let $\mathring{\mathbf{e}}_u, \mathring{\mathbf{b}}_u, \mathring{\mathbf{m}}_{vu}$ denote the original (non-residual) prior beliefs, final beliefs, and messages; we denote with $\mathbf{e}_u, \mathbf{b}_u, \mathbf{m}_{vu}$ their residual counterparts (see Table 5).

Note that the prior and final beliefs of a node sum to 1, i.e. each belief vector b_u is centered around $1/k_s$ where s denotes the type of node u. Similarly, w.l.o.g., messages can be assumed to be centered around 1 (by selecting an appropriate Z_{uv}). Therefore, the residual beliefs and messages are obtained by subtracting their respective *center value* ($1/k_s$ or 1) from their respective initial values. For a node u of type s, these would be:

$$e_u(i) = \mathring{e}_u(i) - \frac{1}{k_s};\ b_u(i) = \mathring{b}_u(i) - \frac{1}{k_s};\ m_{vu}(i) = \mathring{m}_{vu}(i) - 1$$

The normalization constraints on the original values $\mathring{e}_u, \mathring{b}_u$ and $\mathring{m}_{vu}$ thus translate to

$$\sum_i e_u(i) = \sum_i b_u(i) = \sum_i m_{vu}(i) = 0 \quad \forall \quad u, v$$

for the residual values.

The overall motivation behind this procedure is to rewrite BP updates in terms of the residuals. Approximating these residuals finally enables us to derive the linearized BP update equations.

4.3 Proposed ZooBP

Before we proceed to give our main theorem , we introduce some notation that would enable us to solve for the combined beliefs of all nodes via a single equation system. Following this, we provide an example illustrating the definitions.

Definition 7 (Persona)**.** *We use the term "persona" to denote a particular class label of a particular node. For example, if Smith is a node who can be a "democrat" or a "republican", we have the following personas: Smith-democrat, Smith-republican. In general, if there are n_s nodes of type s and each can belong to any of the k_s classes, we have $p_s = n_s k_s$ personas in total, for all type-s nodes.*

Let us denote the prior residual beliefs of all nodes of type s via the matrix $\mathbf{E}_s$ (see Table 5). Here, each row represents the prior residual information about each node, i.e. $\mathbf{e}_u$. If no prior information is given, the row is zero. Similarly, denote with $\mathbf{B}_s$ the final residual beliefs of all nodes of type s after the convergence of belief propagation.

Further, instead of representing the beliefs for each node type individually, we use a joint representation based on the following definition:

Definition 8 (Vectorized residual beliefs $\mathbf{e}, \mathbf{b}$)**.** *Based on the type-s residual prior and final belief matrices $\mathbf{E}_s$ and $\mathbf{B}_s$ ($s \in \mathcal{S}$) the vectorized residual prior and final beliefs are constructed as:*

$$\mathbf{e} = \left[vec(\mathbf{E}_1)^T \quad \dots \quad vec(\mathbf{E}_S)^T\right]^T \tag{6}$$

$$\mathbf{b} = \left[vec(\mathbf{B}_1)^T \quad \dots \quad vec(\mathbf{B}_S)^T\right]^T \tag{7}$$

We now ask: How can we describe the net influence that personas of type s exert on personas of type s'? We define a matrix $\mathbf{P}_{ss'}$ which captures exactly this (Eq. (8)). By concatenating these matrices suitably, we also define the persona-influence matrix $\mathbf{P}$, which consolidates information about how each of the P personas in our graph affects another. Here, we provide equations to derive $\mathbf{P}$ from the graph structure $\mathbf{A}_t$ and the network effects $\mathbf{H}_t, \epsilon_t$ for each $t \in \mathcal{T}$; and later in Section 4.4, we will see how this term naturally emerges from our derivation.

Definition 9 (Persona-influence matrix $\mathbf{P}$)**.** *From the type-t interaction strength, adjacency and residual compatibility matrices ϵ_t, $\mathbf{A}_t$ and $\mathbf{H}_t$, the persona-influence matrix, which summarizes the net effect a class (label) on a node (i.e., a persona) has on another, is constructed as:*

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{11} & \dots & \mathbf{P}_{1S} \\ \vdots & \ddots & \vdots \\ \mathbf{P}_{S1} & \dots & \mathbf{P}_{SS} \end{bmatrix};\ \mathbf{P}_{ss'} = \sum_{t \in \mathcal{T}_{ss'}} \frac{\epsilon_t}{k_s} (\mathbf{H}_t \otimes \mathbf{A}_t) \tag{8}$$

where $\mathcal{T}_{ss'}$ is the set of edge types that connect node types s and s'.

Let us use the term *type-t degree* to denote the count of type-t edges incident on a node. If $t \in \mathcal{T}_{ss'}$ and $u \notin \mathcal{V}_s \cup \mathcal{V}_{s'}$, then its type-$t$ degree is defined as zero. Stacking type-t degrees of all s-type nodes diagonally in a $n_s \times n_s$ matrix, we obtain the type-t degree matrix of s-type nodes, $\mathbf{D}_{st}$.

Analogous to the question we asked before we constructed $\mathbf{P}$, we now ask: what is the net influence that a persona exerts on itself through its neighbors? It is important to account for this echo-influence and deduct it from the persona-influence matrix before solving for node beliefs. We calculate this quantity, called echo-cancellation matrix from the degree matrices $\mathbf{D}_{st}$ and the network effects $\mathbf{A}_t$ and ϵ_t for $t \in \mathcal{T}$. Again, we provide the equations here and postpone the derivation until Section 4.4.

Definition 10 (Echo-cancellation matrix $\mathbf{Q}$)**.** *From the diagonal degree matrices $\mathbf{D}_{st}$ and residual compatibility matrices $\mathbf{H}_t$ and interaction strengths ϵ_t for $t \in \mathcal{T}$, the echo-cancellation matrix may be constructed as:*

$$\mathbf{Q} = \bigoplus_{i=1}^{S} \mathbf{Q}_s \quad \textit{where,} \quad \mathbf{Q}_s = \sum_{s' \in \mathcal{S}} \sum_{t \in \mathcal{T}_{ss'}} \frac{\epsilon_t^2}{k_s k_{s'}} (\mathbf{H}_t \mathbf{H}_t^T \otimes \mathbf{D}_{st}) \tag{9}$$

for the usual meaning of $\mathcal{T}_{ss'}$. Here, $\bigoplus$ denotes the direct sum (Definition 6).

Observe that our persona-influence and echo-cancellation matrices are extremely sparse due to the Kronecker product with the adjacency and diagonal degree matrices, respectively; and hence can be efficiently stored in 4GB main memory for even million scale graphs!

The following example illustrates the above definitions.

Example 3. *Continuing our previous example of a bipartite graph on readers and news articles with "who reads what" edges, let our graph now have 2 readers - R1,R2 and 2 news articles - A, B with adjacency matrix and hence, diagonal degree matrices given by:*

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix};\ \mathbf{D}_R = \mathbf{D}_N = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Suppose also, the nodes' residual prior belief matrices are initialized as

$$\mathbf{E}_R = \begin{bmatrix} -0.03 & 0.03 \\ 0 & 0 \end{bmatrix} \mathbf{E}_N = \begin{bmatrix} 0 & 0 & 0 \\ -0.03 & 0.02 & 0.01 \end{bmatrix}$$

stating that R1 is likely to be a Republican and article B is likely to be about democracy. Then, the vectorized residual prior belief vector would be:

$$\mathbf{e} = \begin{bmatrix} -3 & 3 & 0 & 0 & 0 & 0 & 0 & -3 & 2 & 1 \end{bmatrix}^T \times 10^{-2}$$

Thus, using ϵ and $\mathbf{H}$ calculated earlier, the persona-influence and the echo-cancellation matrices can be derived as:

$$\mathbf{P} = \begin{bmatrix} \mathbf{0} & \frac{\epsilon}{2}\mathbf{H} \otimes \mathbf{A} \\ \frac{\epsilon}{3}\mathbf{H}^T \otimes \mathbf{A}^T & \mathbf{0} \end{bmatrix}; \quad \mathbf{Q} = \begin{bmatrix} \frac{\epsilon^2}{6}\mathbf{H}\mathbf{H}^T \otimes \mathbf{D}_R & \mathbf{0} \\ \mathbf{0} & \frac{\epsilon^2}{6}\mathbf{H}^T\mathbf{H} \otimes \mathbf{D}_N \end{bmatrix}$$

Now, we are ready to state our main theorem.

Theorem 1 (ZooBP)**.** *If $\mathbf{b}, \mathbf{e}, \mathbf{P}, \mathbf{Q}$ are constructed as described above, the linear equation system approximating the final node beliefs given by BP is:*

$$\mathbf{b} = \mathbf{e} + (\mathbf{P} - \mathbf{Q})\mathbf{b} \qquad (\text{ZooBP}) \tag{10}$$

Proof. See Appendix. ■

Lemma 2 (ZooBP *)**.** *Further, if echo cancellation can be ignored, the linear equation system simplifies to:*

$$\mathbf{b} = \mathbf{e} + \mathbf{P}\mathbf{b} \qquad (\text{ZooBP *}) \tag{11}$$

Note that our method can also easily handle weighted edges, by appropriately modifying the adjacency matrices to reflect the weights on the edges. Furthermore, ZooBP contains two existing works as special cases:

Lemma 3 (LinBP and FaBP are special cases of ZooBP)**.** *For a single node-type connected by a single edge type, our formulation reduces to that of* LinBP*. In addition, if we constrain the nodes to belong to only two classes, our formulation reduces to that of* FaBP*.*

Proof. Assuming a single node-type and a single edge-type, the persona-influence and echo-cancellation matrices reduce to:

$$\mathbf{P} = \frac{\epsilon}{k}\mathbf{H} \otimes \mathbf{A} \quad \text{and,} \quad \mathbf{Q} = \frac{\epsilon^2}{k^2}\mathbf{H}^2 \otimes \mathbf{D}$$

Using this together with the relationship between the residual compatibility matrix in both methods ($\hat{\mathbf{H}} = \frac{\epsilon}{k}\mathbf{H}$), our theorem becomes

$$\mathbf{b} = \mathbf{e} + (\mathbf{I} + \hat{\mathbf{H}} \otimes \mathbf{A} + \hat{\mathbf{H}}^2 \otimes \mathbf{D})\mathbf{b}$$

which is exactly LinBP. Thus, LinBP is a special case of ZooBP. As FaBP is a special case of LinBP, it follows that ZooBP subsumes FaBP as well. ■

4.4 Derivation of ZooBP (proofs in appendix)

Lemma 4 (Residual BP)**.** *For a pair of nodes $u \in \mathcal{V}_s$ and $v \in \mathcal{V}_{s'}$, BP update assignments can be approximated in terms of residual messages and beliefs as:*

$$b_u(i) \quad \leftarrow \quad e_u(i) + \frac{1}{k_s} \sum_{v \in \mathcal{N}_u} \sum_{t \in \mathcal{T}_{uv}} m_{vu}^{(t)}(i)$$

$$m_{vu}^{(t)}(i) \quad \leftarrow \quad \frac{\epsilon_t}{k_{s'}} \sum_j H_t(i,j) \Big(k_{s'} b_v(j) - m_{uv}^{(t)}(j) \Big)$$

where $\mathbf{m}_{vu}^{(t)}$ indicates the message vector that v passes to u through an edge of type t, $\mathcal{N}_u$ is the set of neighbors of u and $\mathcal{T}_{uv}$ is the multiset of edge types corresponding to the edges connecting u and v.

The proof makes use of the following two approximations for small residuals:

$$\ln(1 + b_u(i)) \quad \approx \quad b_u(i)$$

$$\frac{\frac{1}{k_{s'}} + b_v(j)}{1 + m_{uv}^{(t)}(j)} \quad \approx \quad \frac{1}{k_{s'}} + b_v(j) - \frac{m_{uv}^{(t)}(j)}{k_{s'}}$$

The assumption of "small residuals" is reasonable because the magnitude of residual beliefs has a linear dependence on ϵ and for a given nature of network effects (homophily/heterophily/mixed), decreasing the interaction strength does not affect the accuracy of ZooBP compared to BP as we demonstrate empirically.

Lemma 5 (Steady State Messages)**.** *For small residuals and after convergence of belief propagation, message propagation from a node $v \in \mathcal{V}_{s'}$ to node $u \in \mathcal{V}_s$ through an edge of type $t \in \mathcal{T}$ can be expressed in terms of the residual compatibility matrices and steady beliefs approximately as:*

$$\mathbf{m}_{vu}^{(t)} \quad = \quad \epsilon_t \mathbf{H}_t \mathbf{b}_v - \frac{\epsilon_t^2}{k_{s'}} \mathbf{H}_t \mathbf{H}_t^T \mathbf{b}_u \tag{12}$$

Lemma 6 (Type-s ZooBP)**.** *Using type-s prior and final residual beliefs, $\mathbf{E}_s$ and $\mathbf{B}_s$, type-t adjacency and residual compatibility matrices $\mathbf{A}_t$ and $\mathbf{H}_t$ and diagonal degree matrices $\mathbf{D}_{st}$, the final belief assignment of type-s nodes from belief propagation can be approximated by the equation system:*

$$\mathbf{B}_s = \mathbf{E}_s + \sum_{s' \in \mathcal{S}} \sum_{t \in \mathcal{T}_{ss'}} \frac{\epsilon_t}{k_s} \mathbf{A}_t \mathbf{B}_{s'} \mathbf{H}_t^T - \frac{\epsilon_t^2}{k_s k_{s'}} \mathbf{D}_{st} \mathbf{B}_s \mathbf{H}_t \mathbf{H}_t^T$$

4.5 Iterative Updates and Convergence

Using Theorem 1, the closed form solution for node beliefs is:

$$\mathbf{b} \quad = \quad (\mathbf{I} + \mathbf{Q} - \mathbf{P})^{-1} \mathbf{e} \tag{13}$$

However, in practice, computation of the inverse of a large matrix such as $(\mathbf{I} + \mathbf{Q} - \mathbf{P})$ is very expensive and is done iteratively. Hence, we propose to do iterative updates of the form:

$$\mathbf{b} \quad \leftarrow \quad \mathbf{e} + (\mathbf{P} - \mathbf{Q})\mathbf{b} \tag{14}$$

Theorem 2 gives precise theoretical guarantees for the convergence of these iterative updates.

Theorem 2 (Exact convergence of ZooBP)**.** *The necessary and sufficient condition for convergence of iterative updates in Eq. (14) in terms of the persona-influence matrix $\mathbf{P}$ and echo-cancellation matrix $\mathbf{Q}$ is:*

$$\text{ZooBP } \textit{converges} \iff \rho(\mathbf{P} - \mathbf{Q}) < 1 \tag{15}$$

Proof. From the Jacobi method for solving linear equations [22], we know that the update in Eq. (14) converges for any arbitrary initialization of $\mathbf{b}$ if and only if the spectral radius of $\mathbf{P} - \mathbf{Q}$ is strictly less than 1. ■

The implicit convergence criterion poses difficulties in choosing appropriate ϵ_t to a practitioner. Thus, for practitioners' benefit, we tie all interaction strengths as $\epsilon_t = \epsilon \ \ \forall t$, use the fact that the spectral norm of a matrix is bounded above by any matrix norm $||\cdot||$ to provide an easier-to-use sufficient condition for convergence. This is stated in Theorem 3.

Theorem 3 (Sufficient convergence of ZooBP)**.** *Let $\mathbf{P}'$ and $\mathbf{Q}'$ be the persona-influence and echo-cancellation matrices obtained from Eq. (8) and Eq. (9) by temporarily setting $\epsilon_t = 1 \quad \forall t$. If the overall interaction strength ϵ is chosen such that,*

$$\epsilon \leq \frac{-\,||\mathbf{P}'|| + \sqrt{||\mathbf{P}'||^2 + 4\,||\mathbf{Q}'||^2}}{2\,||\mathbf{Q}'||}$$

then, ZooBP is guaranteed to converge. Here, $||\mathbf{P}'||$ and $||\mathbf{Q}'||$ can be chosen as any (possibly different) matrix norms.

Proof. By tying interaction strengths across all edges, the exact convergence criterion can be restated in terms of $\mathbf{P}'$ and $\mathbf{Q}'$ as:

$$\rho(\epsilon\mathbf{P}' - \epsilon^2\mathbf{Q}') < 1$$

Using triangle inequality, $\rho(\epsilon\mathbf{P}' - \epsilon^2\mathbf{Q}') \leq \epsilon\rho(\mathbf{P}') + \epsilon^2\rho(\mathbf{Q}')$. Also, as any matrix norm is larger than the spectral norm, this quantity is further bounded above by $\epsilon\,||\mathbf{P}'|| + \epsilon^2\,||\mathbf{Q}'||$. Thus, to ensure convergence, it is sufficient to solve for ϵ using:

$$\epsilon\,||\mathbf{P}'|| + \epsilon^2\,||\mathbf{Q}'|| - 1 < 0$$

This completes the proof. We are free to choose the matrix norms (possibly different norms for $\mathbf{P}'$ and $\mathbf{Q}'$) that would give the tightest bound on the spectral radii of these matrices. ∎

4.6 ZooBP: Time and Space Complexity

Lemma 7 (Time and Space Complexity)**.** *The space and per-iteration time complexity of ZooBP is linear in the total number of nodes and edges, i.e., $\mathcal{O}(|\mathcal{V}| + |\mathcal{E}|)$ and is given by*

$$\mathcal{O}\left(\sum_{s\in\mathcal{S}} k_s n_s + \sum_{s'\in\mathcal{S}} \sum_{t\in\mathcal{T}_{ss'}} k_s(k_s + k_{s'})NNZ(\mathbf{A}_t)\right) \quad (16)$$

where $n_s = |\mathcal{V}_s|$ is the number of s-type nodes, k_s is the corresponding number of classes and $NNZ(\mathbf{A}_t) = |\mathcal{E}_t|$ is the number of non-zero elements in $\mathbf{A}_t$ (i.e., edges of type-t).

Proof. We begin by computing an upper limit on the number of non-zeros of $\mathbf{P}$ and $\mathbf{Q}$:

$$\begin{aligned}
\text{NNZ}(\mathbf{P}) &= \sum_{s\in\mathcal{S}}\sum_{s'\in\mathcal{S}} \text{NNZ}(\mathbf{P}_{ss'}) \\
&\leq \sum_{s\in\mathcal{S}}\sum_{s'\in\mathcal{S}}\sum_{t\in\mathcal{T}_{ss'}} k_s k_{s'}\text{NNZ}(\mathbf{A}_t) \\
\text{NNZ}(\mathbf{Q}) &= \sum_{s\in\mathcal{S}} \text{NNZ}(\mathbf{Q}_s) \\
&\leq \sum_{s\in\mathcal{S}}\sum_{s'\in\mathcal{S}}\sum_{t\in\mathcal{T}_{ss'}} k_s^2\text{NNZ}(\mathbf{A}_t)
\end{aligned}$$

where we have used $\text{NNZ}(\mathbf{X}\otimes\mathbf{Y}) = \text{NNZ}(\mathbf{X})\cdot\text{NNZ}(\mathbf{Y})$. Using the above, we can bound the non-zeros of $\mathbf{P}-\mathbf{Q}$ as:

$$\text{NNZ}(\mathbf{P}-\mathbf{Q}) \leq \sum_{s\in\mathcal{S}}\sum_{s'\in\mathcal{S}}\sum_{t\in\mathcal{T}_{ss'}} k_s(k_s + k_{s'})\text{NNZ}(\mathbf{A}_t) \quad (17)$$

Space Complexity can be computed as the space required to store the sparse matrix $\mathbf{P}-\mathbf{Q}$ and the dense $\sum_{s\in\mathcal{S}} k_s n_s$-dimensional prior and belief vectors. **Per-iteration Time Complexity** (Eq. (16)) is estimated from the number of unit operations (addition/multiplication) involved in computing the LHS of the iterative update (Eq. (14)). The breakdown for each operation is given below.

Computation	Unit ops.
Subtraction of $\mathbf{Q}$ from $\mathbf{P}$	$\mathcal{O}(\text{NNZ}(\mathbf{P}-\mathbf{Q}))$
Multiplication of $(\mathbf{P}-\mathbf{Q})$ and $\mathbf{b}$	$\mathcal{O}(\text{NNZ}(\mathbf{P}-\mathbf{Q}))$
Addition of $\mathbf{e}$ and $(\mathbf{P}-\mathbf{Q})\mathbf{b}$	$\mathcal{O}\left(\sum_{s\in\mathcal{S}} k_s n_s\right)$

∎

Table 4: Sample $\mathbf{H}_+, \mathbf{H}_-$ for product rating networks

$\mathbf{H}_+$	*Good*	*Bad*
Honest	0.5	-0.5
Fraud	-0.5	0.5

$\mathbf{H}_-$	*Good*	*Bad*
Honest	-0.5	0.5
Fraud	0.5	-0.5

4.7 Case Study - Product-Rating Network

In the following section, we introduce a case study of our ZooBP for product-rating networks (which are signed bipartite networks) – other complex scenarios can also be represented easily. The goal is to classify users and products as fraudulent or not.

Let $\mathcal{G} = (\mathcal{V}_u \cup \mathcal{V}_p, \mathcal{E}_+ \cup \mathcal{E}_-)$ be a product rating network where $\mathcal{V}_u$ is the set of users and $\mathcal{V}_p$ is the set of products. Let $n_p = |\mathcal{V}_p|$ be the number of products and $n_u = |\mathcal{V}_u|$ the number of users. The edge sets $\mathcal{E}_+$ and $\mathcal{E}_-$ represent the positive and negative ratings respectively.

Given the edge sets, we denote the corresponding $n_u \times n_p$ adjacency matrices as $\mathbf{A}_+$ and $\mathbf{A}_-$. Here, the rows correspond to users and columns correspond to products. Furthermore, let us use the term *positive degree* to denote the number of positive ratings given to a product or by a user (depending on the node type). Let $\mathbf{D}_{u+}, \mathbf{D}_{p+}$ be the $n_u \times n_u$ and $n_p \times n_p$ diagonal matrices of positive degree for users and products respectively. Similarly, we define diagonal degree matrices of *negative degree* for users and products - $\mathbf{D}_{u-}, \mathbf{D}_{p-}$.

Further, let the residual compatibility matrices for positive and negative edges be $\mathbf{H}_+$ and $\mathbf{H}_-$ and the corresponding edge interaction strengths be ϵ_+ and ϵ_-. Here, the rows correspond to user-classes and columns correspond to product-classes. In general, one would expect honest users to give positive ratings to good products, while positive ratings for fraudulent products are less likely; fraudsters in contrast might give positive ratings to fraudulent products. Thus, the matrices $\mathbf{H}_+$ and $\mathbf{H}_-$ might be instantiated as in Table 4 – of course, in our model, any other constant-margin instantiation can be picked as well.

In general, considering the setting of product rating networks, the persona-influence and echo-cancellation matrices are given by:

$$\mathbf{P} = \begin{bmatrix} \mathbf{0} & \frac{\epsilon_+}{2}\mathbf{H}_+\otimes\mathbf{A}_+ + \frac{\epsilon_-}{2}\mathbf{H}_-\otimes\mathbf{A}_- \\ (\frac{\epsilon_+}{2}\mathbf{H}_+\otimes\mathbf{A}_+ + \frac{\epsilon_-}{2}\mathbf{H}_-\otimes\mathbf{A}_-)^T & \mathbf{0} \end{bmatrix}$$

$$\mathbf{Q} = \begin{bmatrix} \frac{\epsilon_+^2}{4}\mathbf{H}_+\mathbf{H}_+^T\otimes\mathbf{D}_{u+} + \frac{\epsilon_-^2}{4}\mathbf{H}_-\mathbf{H}_-^T\otimes\mathbf{D}_{u-} & \mathbf{0} \\ \mathbf{0} & \frac{\epsilon_+^2}{4}\mathbf{H}_+^T\mathbf{H}_+\otimes\mathbf{D}_{p+} + \frac{\epsilon_-^2}{4}\mathbf{H}_-^T\mathbf{H}_-\otimes\mathbf{D}_{p-} \end{bmatrix}$$

These matrices can now be used to compute the final beliefs using Theorem 1 (ZooBP).

Besides this general solution, let us focus on the case where we set compatibility matrices as in Table 4 and tie

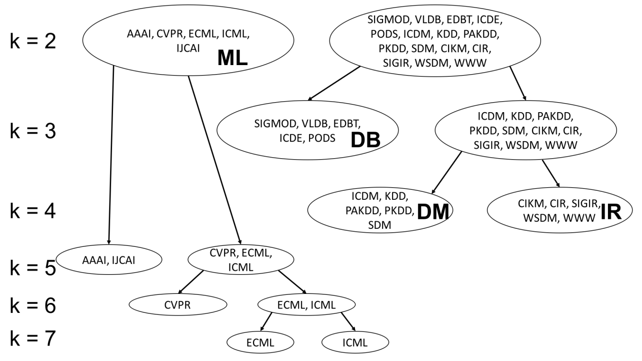

k = 2
k = 3
k = 4
k = 5
k = 6
k = 7
AAAI, CVPR, ECML, ICML, IJCAI
ML
SIGMOD, VLDB, EDBT, ICDE, PODS, ICDM, KDD, PAKDD, PKDD, SDM, CIKM, CIR, SIGIR, WSDM, WWW
SIGMOD, VLDB, EDBT, ICDE, PODS
DB
ICDM, KDD, PAKDD, PKDD, SDM, CIKM, CIR, SIGIR, WSDM, WWW
ICDM, KDD, PAKDD, PKDD, SDM
DM
CIKM, CIR, SIGIR, WSDM, WWW
IR
AAAI, IJCAI
CVPR, ECML, ICML
CVPR
ECML, ICML
ECML
ICML

Figure 3: DBLP 4-area (Databases, Data Mining, Machine Learning, Information Retrieval, resp.) dataset with class hierarchy ($k = [2, \dots, 7]$)

the interaction strengths across both types of edges $\epsilon_+ = \epsilon_- =: \epsilon$.

Let us now define the total adjacency matrix ($\mathbf{A}$) and total diagonal degree matrix ($\mathbf{D}$) as follows:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{A}_+ - \mathbf{A}_- \\ (\mathbf{A}_+ - \mathbf{A}_-)^T & \mathbf{0} \end{bmatrix}; \quad \mathbf{D} = \begin{bmatrix} \mathbf{D}_{u+} + \mathbf{D}_{u-} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{p+} + \mathbf{D}_{p-} \end{bmatrix}$$

Using these, Theorem 4 provides a compact closed-form solution for ZooBP (proof omitted for brevity).

Theorem 4 (ZooBP-2F). *For fraud detection in product-rating networks, if ϵ denotes the desired interaction strength and $\mathbf{A}$ and $\mathbf{D}$ are the total adjacency and diagonal degree matrices of users and products as defined above, the* ZooBP*-2F closed-form solution is:*

$$\mathbf{b} = \mathbf{e} + \left(\begin{bmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{bmatrix} \otimes \left(\frac{\epsilon}{2}\mathbf{A} - \frac{\epsilon^2}{4}\mathbf{D} \right) \right) \mathbf{b}$$

The exact and sufficient conditions for convergence of our ZooBP-2F can be derived as before:

Result 1 (Exact convergence of ZooBP-2F). *For fraud detection in product rating network, the* sufficient and necessary *condition for convergence, in terms of total adjacency matrix $\mathbf{A}$, the total diagonal degree matrix $\mathbf{D}$ and interaction strength ϵ is given by: $\rho\left(\frac{\epsilon}{2}\mathbf{A} - \frac{\epsilon^2}{4}\mathbf{D}\right) < 1$*

Result 2 (Sufficient convergence of ZooBP-2F). *For fraud detection in product rating network, with $\mathbf{A}$ and $\mathbf{D}$ denoting the total adjacency and diagonal degree matrices respectively, if the interaction strength (ϵ) is chosen to satisfy*

$$\epsilon < \frac{-||A|| + \sqrt{||A||^2 + 4d_{\max}}}{d_{\max}}$$

then, ZooBP $- 2F$ *is guaranteed to converge.*

5. EXPERIMENTS

To evaluate our proposed method, we are interested in finding answers to the following questions:

Q1. Accuracy: How well can ZooBP reconstruct the final beliefs given by BP? How accurate are its predictions on real-world data?

Q2. (In-)Sensitivity to interaction strength: How does the performance of ZooBP vary with interaction strength ϵ? What happens at the critical ϵ_* from Theorem 2? How sensitive is ZooBP to ϵ when $\epsilon < \epsilon_*$?

Q3. Speed & Scalability: How well does ZooBP scale with the network size? How fast is ZooBP compared to BP? Why? Does the speed-up generalize to networks with arbitrary number of node-/edge-types and classes?

We now describe the data we use for our experiments.

5.1 Data Description and Experimental Setup

We have used two real-world heterogeneous datasets in our experiments. These are explained below.

5.1.1 DBLP

The DBLP 4-area dataset consists of authors and the papers published by them to 12 conferences. In the original dataset, these conferences were split into four areas (DB, DM, ML, IR). To perform a deeper analysis, we varied the number of classes, k, from 2 to 7 by merging or partitioning the above areas based on the conferences (Figure 3). The network is bipartite (node types $\mathcal{S} = \{$author, paper$\}$) with a single type of edges ($\mathcal{T} = \{$authorship$\}$). The goal is to assign a class to each author and paper.

The ground truth areas for papers and authors were obtained as follows. The area of a paper is the area of the conference it is published in. The area of an author is the area which most of her papers belong to, with ties broken randomly. As homophily captures the nature of network effects in this dataset, a $k \times k$ compatibility matrix with interaction strength ϵ and residual compatibility matrix $\mathbf{H}_{k\times k} = \mathbf{I}_{k\times k} - \frac{1}{k}\mathbf{1}_{k\times k}$ was used.

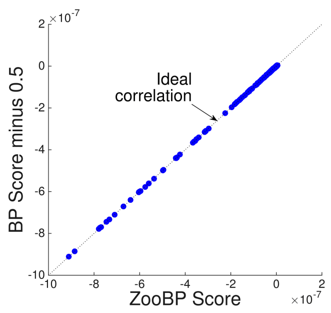
In our experiments, we seeded randomly chosen 30% of the authors and papers to their ground truth areas. The prior for the correct class was set to $+k \times 0.001$ and for the wrong classes was set to -0.001. $[0, \dots, 0]$ was used as the prior for unseeded nodes.

5.1.2 Flipkart

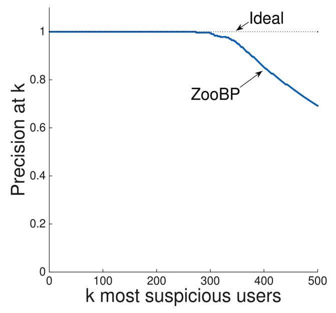
Flipkart is an e-commerce website that provides a platform for sellers to market their products to customers. The dataset consists of about 1M users, their $\sim$3.3M ratings to $\sim$512K products and $\sim$1.7M ratings to $\sim$4K sellers. In addition, we also have the connections between sellers and products. All ratings are on a scale of 1 to 5 – for simplicity, we treated 4 and 5 star ratings as positive edges, 1 and 2 star as negative edges and ignored the 3 star ratings.

We consider two versions of the data: (1) Flipkart-(2,2) (or Flipkart in short) containing only user-product rating information (node types $\mathcal{S} = \{$user, product$\}$ and edge types $\mathcal{T} = \{$positive rating for product, negative rating for product$\}$); and (2) Flipkart-(3,5) containing all 3 node types (user, product, seller) and 5 edge types (positive rating for product, negative rating for product, positive rating for seller, negative rating for seller, seller sells product).

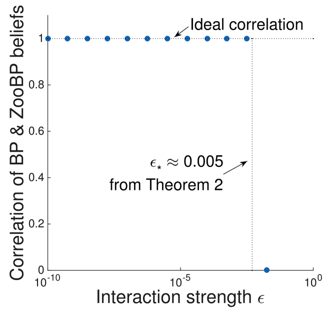
In both the Flipkart datasets, our goal is to classify users and products (and sellers) as fraudulent or not. $\mathbf{H}_+$ and $\mathbf{H}_-$ for both user-product and user-seller edges were chosen as in Table 4 and ϵ values were tied and set to 10^{-4}, unless mentioned otherwise. We used 50 manually labeled fraudsters as seed labels and initialized their prior to $[-0.05, +0.05]$ respectively for the honest and fraudulent classes. The prior for other users and all products (and all sellers) were set to $[0, 0]$.



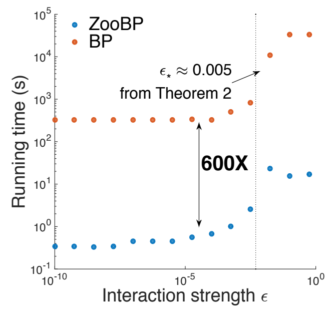
(a) ZooBP matches the beliefs of BP



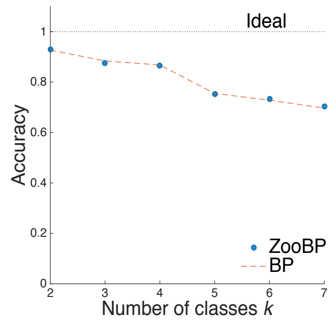
(b) ZooBP is effective in fraud detection



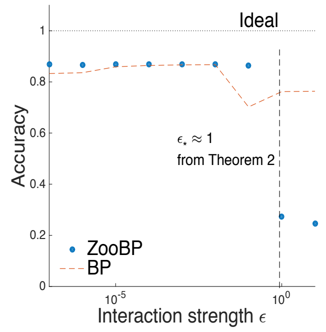
(c) ZooBP's approx. quality is independent of ϵ (when $\epsilon < \epsilon_*$)



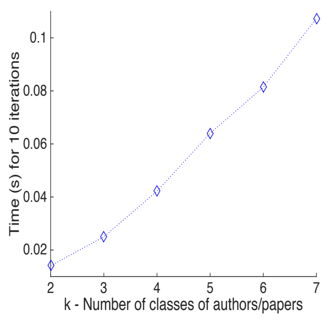
(d) ZooBP's speed gain is fairly robust to ϵ (for $\epsilon < \epsilon_*$)



(e) ZooBP is accurate on DBLP



(f) ZooBP's accuracy on DBLP ($k = 4$) is robust to ϵ (for $\epsilon < \epsilon_*$)



(g) ZooBP scales quadratically with #classes on DBLP

k	BP (s)	ZooBP (s)
2	24.0196	0.0141
3	28.8042	0.0250
4	33.4384	0.0422
5	37.6492	0.0640
6	42.1226	0.0814
7	46.5648	0.1073

(h) BP vs ZooBP: Matlab running time for 10 iterations on DBLP with k

Figure 4: Experimental results on Flipkart (a-d) and DBLP (e-h) data: ZooBP is accurate, robust, fast and scalable

We provide the full analysis on the DBLP and Flipkart datasets; for brevity, we only present the fraud detection precision results on Flipkart-(3,5). To compare running times on DBLP and Flipkart data with BP, we used an off-the-shelf Matlab implementation of BP for signed bipartite networks [1]. To enable a fair comparison, we implemented ZooBP also in Matlab.

Q1. Accuracy

A plot of the final beliefs returned by BP and ZooBP on Flipkart for $\epsilon = 10^{-4}$ is shown in Figure 4a. Here, we have subtracted 0.5 from the BP score (see y-axis) to match the scale of beliefs from both methods. We see that all points lie on the line of slope 1 passing through the origin, showing that ZooBP beliefs are highly correlated with BP beliefs. Such a trend was observed for all the ϵ values we tried (while ensuring that $\epsilon < \epsilon_*$, the limit given by Theorem 2).

Upon applying ZooBP to our data, we provided the list of 500 most fraudulent users (after sorting beliefs) to the domain experts at Flipkart, who verified our labels by studying various aspects of user behavior such as frequency and distribution of ratings and review text given by them. Figure 4b and Figure 2c depict how the precision at k changes with k over the top 500 results on Flipkart and Flipkart-(3,5) datasets. The high precision ($\tilde{1}00\%$ for top 250; $\tilde{7}0\%$ for top 500 users) confirms the effectiveness of ZooBP. Owing to difficulty in obtaining ground truth for all 1M users, studying recall was not possible.

Using the DBLP data, we study the performance for a graph from a different domain (citation network), with more than two classes. Figure 4e plots accuracy vs number of classes, k. The uniformly high accuracy across k suggest our performance can be expected to generalize well to networks from different domains with arbitrary classes.

In sum, our accuracy results show that (1) our assumption of constant-margin compatibility matrix is applicable in several realistic scenarios (2) our linear approximations do not lower the quality of prediction, thus making ZooBP extremely useful in practice, for solving several real world problems.

Q2. (In-)Sensitivity to interaction strength

Next, we study how the compatibility matrix (through interaction strength ϵ) influences the performance and speed of ZooBP. Figures 4c, 4d summarize the results on Flipkart.

The correlation (of BP and ZooBP beliefs) and the running time were found to be fairly constant with ϵ as long as $\epsilon < \epsilon_*$ (the limit from Theorem 2). As the spectral radius of $\mathbf{P} - \mathbf{Q}$ approaches 1, a slight increase in running time near ϵ_* is observed; but the correlation is still high. When $\epsilon > \epsilon_*$, the algorithm does not converge – ZooBP runs to a manually set maximum iteration count of 200. Hence, the running time suddenly increases past the dotted line, while the correlation drops to 0.0001. The resulting beliefs for high interaction strength ($\epsilon > 0.1$) were found to be unbounded (reaching $\pm\infty$) for some nodes – making the correlation coefficient indeterminate (these are omitted in Figure 4c).

On the DBLP data, we are not restricted to study correlation but are able to analyze the actual accuracy of BP and ZooBP with varying ϵ. Figure 4f depicts the classification accuracy vs ϵ for the DBLP dataset with $k = 4$. We see that both BP and ZooBP achieve a robust high accuracy on a range of ϵ values within the convergence limit. Not surprisingly, when ϵ was increased beyond $\epsilon_* = 1$, the performance of both methods deteriorated. This suggests our algorithm is *practically useful*, with robust approximation quality in BP's optimal range of interaction strength.

These results show that our method is fairly robust (except around and beyond ϵ_*) and not sensitive to the selection of interaction strength in general. Moreover, this value of ϵ_* is exactly as predicted by Theorem 2 (specifically, Result 1), thus validating its correctness.

Note to practitioner: Owing to the finite precision of machines, we recommend setting $\epsilon \in [0.01\epsilon^\dagger, 0.1\epsilon^\dagger]$, where $\epsilon^\dagger$ is calculated from Theorem 3.

Q3. Speed & Scalability

To examine the scalability of our method, we uniformly sampled 1K-3.3M edges from the FLIPKART data and timed BP and ZOOBP (with $\epsilon = 10^{-4}$) on the resulting subgraphs. We focus on the time taken for computations alone and ignore the time to load data and initialize matrices. The results are shown in Figure 2a (MATLAB) and 2b (C++).

We see that ZOOBP scales linearly with the number of edges in the graph (i.e., graph size), which is same as the scalability of BP. In addition, on MATLAB, ZOOBP also offers a 600× *speed-up* compared to BP, which is one of its most important practical advantages. On FLIPKART dataset with 3.3M edges, ZOOBP requires only 1 second to run!

What can this speed-up be attributed to? There are two primary contributing factors: **(F1)** ZOOBP replaces the expensive logarithms and exponentiation operations in BP by multiplication and addition; **(F2)** ZOOBP (via Theorem 1) converts the iterative BP algorithm into a matrix problem – it foregoes the redundant message computation and exploits optimized sparse-matrix operations.

To investigate the relative importance of the above factors, we implemented Lemma 4 in MATLAB. Lemma 4is similar to BP except in operating on residuals directly in the linear space through lighter-weight operations and hence serves as a clean break point between BP and ZOOBP to compare the speed-ups due to F1 and F2 individually. Our experimental observations are summarized below:

- **Savings A (∼ 2×)** BP → Lemma 4 (lighter operations)
  This speed-up is not tied to MATLAB as we demonstrate through identical experiments in C++ (Figure 2b). Savings A is platform-independent with the precise speed-up factor depending on the architecture-specific relative speed of elementary floating point operations (add, multiply) and function calls (exp, log).
- **Savings B (∼ 300×):** Lemma 4 → ZOOBP (optimized sparse matrix operations of MATLAB).
  We note that although the 300× savings from MATLAB implementation is largely due to its inefficient handling of loops, it may prove to be a critical factor of consideration for a number of data mining practitioners.

Can we explain the speed-up in terms of the architecture specifications? Our experiments used Intel i5 (Haswell) processor[3]. In this architecture, multiplication instructions (FMUL, FIMUL) issued up to two times more macro instructions (OPs) than addition or subtraction (FADD, FSUB, FIADD, FISUB). Further, function calls (i.e., control transfer instructions such as CALL) needed 2-3 times more clock cycles compared to arithmetic operations.

[3]`http://www.agner.org/optimize/instruction_tables.pdf` pages 189-191

This is exactly the speed-up (2-3×) that ZOOBP achieves over BP in C++, as shown in Figure 2b.

Do the speed gains persist as the number of classes grows? The answer is 'yes'. Figure 4g and Figure 4h show the results on the DBLP data. ZOOBP scales quadratically with number of class labels, as expected from Lemma 7; but the speed-up gains were consistently ≈ 600× even as k varied (Figure 4h.)

Comparison with the state-of-the-art: Table 1 gives the qualitative comparison of ZOOBP with top competitors. Only BP (and its asynchronous equivalent, RBP) can solve the general problem, but neither of them provides a closed-form solution or convergence guarantees. Still, we have provided comparison results against BP. None of the other methods (LINBP [12], FABP [15], [11]) can handle arbitrary heterogeneous graphs (e.g., FLIPKART-(3,5)) and are dropped from comparison. We use CAMLP as a baseline on the DBLP data, although it cannot handle multiple node-types. In our experiments on DBLP data with $k = 4$, ZOOBP practically tied CAMLP (86% vs. 87% accuracy).

In summary, our experiments show that ZOOBP obtains a very high prediction accuracy on real-world data, while at the same time, being highly scalable at handling million-scale graphs.

6. CONCLUSIONS

We presented ZOOBP, a novel framework which approximates BP with constant-margin compatibility matrices, in any undirected weighted heterogeneous graph. Our method has the following advantages:

- **Generality**: ZOOBP approximates BP in *any* kind of undirected weighted heterogeneous graph, with arbitrarily many node and edge types. Moreover, it includes existing techniques like FABP and LINBP as special cases.
- **Closed-form solution**: ZOOBP leads to a closed form solution (Theorem 1, Eq. 10), which results in exact convergence guarantees (Theorem 2).
- **Scalability**: ZOOBP scales linearly with the number of edges in the graph; moreover, it nevers loses, and it usually wins over traditional BP, with *up to* 600× *speed-up* for MATLAB implementation.
- **Effectiveness**: Applied on real data (FLIPKART), ZOOBP matches the accuracy of BP, achieving 92.3 % precision for the top 300 nodes.

Reproducibility: Our code is open-sourced at `http://www.cs.cmu.edu/ deswaran/code/zoobp.zip` and is available for public use.

7. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grants No. CNS-1314632 and IIS-1408924, by the German Research Foundation, Emmy Noether grant GU 1409/2-1, and by the Technical University of Munich - Institute for Advanced Study, funded by the German Excellence Initiative and the European Union Seventh Framework Programme under grant agreement no 291763, co-funded by the European Union. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, or other funding parties. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

8. REFERENCES

[1] L. Akoglu, R. Chandy, and C. Faloutsos. Opinion fraud detection in online reviews by network effects. In *ICWSM*, pages 2–11, 2013.

[2] F. Ayres. *Schaum's outline of theory and problems of matrices.* McGraw-Hill, 1962.

[3] B. Boden, S. Günnemann, H. Hoffmann, and T. Seidl. Mining coherent subgraphs in multi-layer graphs with edge labels. In *SIGKDD*, pages 1258–1266, 2012.

[4] O. Chapelle, B. Schölkopf, and A. Zien. *Transductive Inference and Semi-Supervised Learning.* MIT Press.

[5] O. Chapelle, B. Schölkopf, A. Zien, et al. Semi-supervised learning. 2006.

[6] D. H. Chau, A. Kittur, J. I. Hong, and C. Faloutsos. Apolo: making sense of large network data by combining rich user interaction and machine learning. In *ACM SIGCHI*, pages 167–176, 2011.

[7] G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. *UAI*, pages 165–173, 2006.

[8] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. *IJCV*, pages 41–54, 2006.

[9] M. P. Fossorier, M. Mihaljevic, and H. Imai. Reduced complexity iterative decoding of low-density parity check codes based on belief propagation. *IEEE Transactions on communications*, pages 673–680.

[10] B. J. Frey and F. R. Kschischang. Probability propagation and iterative decoding. In *Allerton Conference on Communications, Control and Computing*, pages 482–493, 1996.

[11] W. Gatterbauer. The linearization of pairwise markov networks. *arXiv preprint arXiv:1502.04956*, 2015.

[12] W. Gatterbauer, S. Günnemann, D. Koutra, and C. Faloutsos. Linearized and single-pass belief propagation. *PVLDB*, 8(5):581–592, 2015.

[13] H. V. Henderson and S. R. Searle. The vec-permutation matrix, the vec operator and kronecker products: A review. *Linear and multilinear algebra*, pages 271–288, 1981.

[14] D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In *KDD*, pages 593–598. ACM, 2004.

[15] D. Koutra, T.-Y. Ke, U. Kang, D. H. P. Chau, H.-K. K. Pao, and C. Faloutsos. Unifying guilt-by-association approaches: Theorems and fast algorithms. In *ECML PKDD*, pages 245–260, 2011.

[16] J. M. Mooij and H. J. Kappen. Sufficient conditions for convergence of the sum–product algorithm. *IEEE Transactions on Information Theory*, pages 4422–4437, 2007.

[17] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *UAI*, pages 467–475, 1999.

[18] J. Neville and D. Jensen. Iterative classification in relational data. In *AAAI Workshop on Learning Statistical Models from Relational Data*, pages 13–20, 2000.

[19] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *WWW*, pages 201–210, 2007.

[20] J. Pearl. Reverend bayes on inference engines: A distributed hierarchical approach. In *AAAI*, pages 133–136, 1982.

[21] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann, 2014.

[22] Y. Saad. *Iterative methods for sparse linear systems.* Siam, 2003.

[23] J. Sun, N.-N. Zheng, and H.-Y. Shum. Stereo matching using belief propagation. *IEEE Transactions on pattern analysis and machine intelligence*, pages 787–800, 2003.

[24] H. Tong, C. Faloutsos, and J.-Y. Pan. Fast random walk with restart and its applications. 2006.

[25] Y. Yamaguchi, C. Faloutsos, and H. Kitagawa. Camlp: Confidence-aware modulated label propagation. SDM, 2016.

[26] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. pages 236–239. 2003.

[27] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Citeseer, 2002.

APPENDIX

A. NOMENCLATURE

Table 5: Nomenclature

Symbol	Meaning
$\mathcal{S}$	set of node types
S	$\lvert\mathcal{S}\rvert$, the number of node types
s, s'	a type of node; an element in $\mathcal{S}$
$\mathcal{V}_s$	set of nodes belonging to type s
n_s	$\lvert\mathcal{V}_s\rvert$, the number of type-$s$ nodes
k_s	the number of classes (labels) for a type-s node
p_s	$n_s k_s$, the number of personas for all type-s nodes
$\mathbf{E}_s, \mathbf{B}_s$	$n_s \times k_s$ prior and final beliefs (resp.) for all type-s nodes
P	$\sum_{s\in\mathcal{S}} p_s$, the total number of personas for all nodes
$\mathcal{T}$	set of edge types
T	$\lvert\mathcal{T}\rvert$, the number of edge types
$\mathcal{T}_{ss'}$	set of edge types connecting node typess and s'
$\mathcal{T}_{uv}$	multiset of edge types connecting nodes u and v
t, t'	a type of edge; an element in $\mathcal{T}$
$\mathbf{A}_t$	if $t \in \mathcal{T}_{ss'}$, this is the $k_s \times k_{s'}$ adjacency matrix for edge-type t
ϵ_t	interaction strength for edge-type t
$\mathring{\mathbf{H}}_t, \mathbf{H}_t$	compatibility matrix for edge-type t - given and residual (resp.)
$\mathring{\mathbf{e}}_u, \mathbf{e}_u$	prior belief vector of u - given and residual (resp.)
$\mathring{\mathbf{b}}_u, \mathbf{b}_u$	final belief vector of u - given and residual (resp.)
$\mathring{\mathbf{m}}_{vu}, \mathbf{m}_{vu}$	message vector from v to u - given and residual (resp.)
$\mathbf{D}_{st}$	$n_s \times n_s$ diagonal matrix of type-t degrees of type-s nodes
$\mathbf{P}$	$P \times P$ persona-influence matrix
$\mathbf{Q}$	$P \times P$ echo-cancellation matrix
u, v, w	nodes
i, j, g	node classes (labels)

B. PROOFS

Proof of Lemma 4. We start by rewriting Yedidia's belief update assignment (Eq. (2)) for a node u belonging to type $s \in \mathcal{S}$, in terms of residual beliefs and messages.

$$\frac{1}{k_s} + b_u(i) \leftarrow \frac{1}{Z_u}\left(\frac{1}{k_s} + e_u(i)\right) \prod_{\substack{v\in\mathcal{N}(u)\\ t\in\mathcal{T}_{uv}}} \left(1 + m_{vu}^{(t)}(i)\right)$$

Now, we take logarithms and assume the residual beliefs are small compared to 1 to use the approximation $\ln(1+x) \approx x$. We obtain:

$$b_u(i) \quad \leftarrow \quad -\frac{1}{k_s}\ln Z_u + e_u(i) + \frac{1}{k_s} \sum_{\substack{v\in\mathcal{N}(u)\\ t\in\mathcal{T}_{uv}}} m_{vu}^{(t)}(i) \qquad (18)$$

$$\underbrace{k_s \sum_i b_u(i)}_{=0} \quad \leftarrow \quad -\sum_i \ln Z_u + k_s \underbrace{\sum_i e_u(i)}_{=0} + \sum_{\substack{v\in\mathcal{N}(u)\\ t\in\mathcal{T}_{uv}}} \underbrace{\sum_i m_{vu}^{(t)}(i)}_{=0}$$

In the last step above, we sum both sides over to estimate $Z_u = 1$, which turns out to be constant for all nodes. Substituting this back into Eq. (18) proves the first part of lemma.

To prove the second part of the lemma, we first write Yedidia's update assignment for the message that a node v of type s' passes to a node u of type s through an edge of type t, i.e., $\mathring{m}_{vu}^{(t)}$:

$$\mathring{m}_{vu}^{(t)}(i) \quad \leftarrow \quad \frac{Z_v}{Z_{vu}^{(t)}} \sum_j \frac{\mathring{H}_t(i,j)}{\sum_{i'} \mathring{H}_t(i',j)} \frac{\mathring{b}_v(j)}{\mathring{m}_{uv}^{(t)}(j)}$$

$$1 + m_{vu}^{(t)}(i) \quad \leftarrow \quad \frac{Z_v}{Z_{vu}^{(t)}} \sum_j \frac{1 + \epsilon_t H_t(i,j)}{\sum_{i'} 1 + \epsilon_t H_t(i',j)} \frac{\frac{1}{k_{s'}} + b_v(j)}{1 + m_{uv}^{(t)}(j)}$$

We now use the following approximation (for small residuals):

$$\frac{\frac{1}{k_{s'}} + b_v(j)}{1 + m_{uv}^{(t)}(j)} \approx \frac{1}{k_{s'}} + b_v(j) - \frac{m_{uv}^{(t)}(j)}{k_{s'}}$$

along with $Z_v = 1$ and normalization constraints on residuals simplify the LHS of the above assignment update:

$$\frac{1}{Z_{vu}^{(t)}} \sum_j \frac{1 + \epsilon_t H_t(i,j)}{k_s} \left(\frac{1}{k_{s'}} + b_v(j) - \frac{m_{uv}^{(t)}(j)}{k_{s'}} \right)$$

$$= \frac{1}{Z_{vu}^{(t)} k_s} \Bigg(1 + \underbrace{\sum_j b_v(j)}_{=0} - \frac{1}{k_{s'}} \underbrace{\sum_j m_{uv}^{(t)}(j)}_{=0} + \frac{\epsilon_t}{k_{s'}} \underbrace{\sum_j H_t(i,j)}_{=0}$$

$$+ \epsilon_t \sum_j H_t(i,j) b_v(j) - \frac{\epsilon_t}{k_{s'}} \sum_j H_t(i,j) m_{uv}^{(t)}(j) \Bigg)$$

Thus, we obtain:

$$1 + m_{vu}^{(t)}(i) \leftarrow \frac{1 + \sum_j \epsilon_t H_t(i,j) \left(b_v(j) - \frac{m_{uv}^{(t)}(j)}{k_{s'}} \right)}{Z_{vu}^{(t)} k_s} \tag{19}$$

To calculate $Z_{vu}^{(t)}$, we sum Eq. (19) over i and use $\sum_i H_t(i,j) = \sum_i m_{vu}^{(t)}(i) = 0$. This leads to $Z_{vu}^{(t)} = \frac{1}{k_s}$. Substituting this in Eq. (19) proves the second part of the lemma. ■

Proof of Lemma 5. Rewriting the message update assignment from Lemma 4 after expanding the message sent in the opposite direction (i.e., $m_{uv}^{(t)}(j)$) we have:

$$\begin{aligned} m_{vu}^{(t)}(i) \quad \leftarrow \quad & \sum_j \frac{\epsilon_t H_t(i,j)}{k_{s'}} \Bigg(k_{s'} b_v(j) - \\ & \sum_g \frac{\epsilon_t H_t(g,j)}{k_s} \Big(k_s b_u(g) - m_{vu}^{(t)}(g) \Big) \Bigg) \end{aligned}$$

At convergence, $\mathbf{m}_{vu}^{(t)}$ on both sides need to be identical. So, we replace the update sign with an equality and group similar terms together as follows:

$$m_{vu}^{(t)}(i) - \frac{\epsilon_t^2}{k_s k_{s'}} \sum_j H_t(i,j) \sum_g H_t(g,j) m_{vu}^{(t)}(g) =$$

$$\epsilon_t \sum_j H_t(i,j) b_v(j) - \frac{\epsilon_t^2}{k_{s'}} \sum_j H_t(i,j) \sum_g H_t(g,j) b_u(g)$$

This equation can then be written in matrix-vector notation as:

$$\big(\mathbf{I}_{k_s} - \underbrace{\frac{\epsilon_t^2}{k_s k_{s'}} \mathbf{H}_t \mathbf{H}_t^T}_{\mathbf{X}} \big) \mathbf{m}_{vu}^{(t)} = \epsilon_t \mathbf{H}_t \mathbf{b}_v - \frac{\epsilon_t^2}{k_{s'}} \mathbf{H}_t \mathbf{H}_t^T \mathbf{b}_u$$

The entries of $\mathbf{X} << \frac{1}{k_s}$, and thus inverse of $(\mathbf{I}_{k_s} - \mathbf{X})$ always exists and is, further, approximately $\mathbf{I}_{k_s}$ as $\mathbf{X}$ is composed of second order terms of the (low) interaction strength. This leads to Lemma 5. ■

Proof of Lemma 6. Using Lemma 4, the residual belief of a node $u \in \mathcal{V}_s$ can be written in vector notation as:

$$\mathbf{b}_u \quad \leftarrow \quad \mathbf{e}_u + \frac{1}{k_s} \sum_{v \in \mathcal{N}(u)} \sum_{t \in \mathcal{T}_{uv}} \mathbf{m}_{vu}^{(t)}$$

Substituting the steady state value of $\mathbf{m}_{vu}^{(t)}$ from Lemma 5, the final belief of u at convergence is:

$$\begin{aligned} \mathbf{b}_u \quad &= \quad \mathbf{e}_u + \frac{1}{k_s} \sum_{v \in \mathcal{N}(u)} \sum_{t \in \mathcal{T}_{uv}} \left(\epsilon_t \mathbf{H}_t \mathbf{b}_v - \frac{\epsilon_t^2}{k_{s'}} \mathbf{H}_t \mathbf{H}_t^T \mathbf{b}_u \right) \\ &= \quad \mathbf{e}_u + \sum_{v \in \mathcal{N}(u)} \sum_{t \in \mathcal{T}_{uv}} \frac{\epsilon_t}{k_s} \mathbf{H}_t \mathbf{b}_v - \frac{\epsilon_t^2 d_u^{(t)}}{k_s k_{s'}} \mathbf{H}_t \mathbf{H}_t^T \mathbf{b}_u \end{aligned}$$

where $d_u^{(t)}$ is the type-t degree of u, i.e., the number of type-t edges incident on u. Rewriting the above equation in matrix form using type-t adjacency matrices $\mathbf{A}_t$ for $t \in \mathcal{T}$, prior and final residual type-s belief matrices $\mathbf{B}_s$ for $s \in \mathcal{S}$ and diagonal degree matrices $\mathbf{D}_{st}$ summarizing type-t degree of type-s nodes, yields Lemma 6:

$$\mathbf{B}_s = \mathbf{E}_s + \sum_{s' \in \mathcal{S}} \sum_{t \in \mathcal{T}_{ss'}} \frac{\epsilon_t}{k_s} \mathbf{A}_t \mathbf{B}_{s'} \mathbf{H}_t^T - \frac{\epsilon_t^2}{k_s k_{s'}} \mathbf{D}_{st} \mathbf{B}_s \mathbf{H}_t \mathbf{H}_t^T$$

■

Proof of Theorem 1. In order to bring $\mathbf{B}_s$ and $\mathbf{B}_{s'}$ out of the matrix product in Eq. (13), we vectorize Eq. (13) and then use Roth's column lemma (Eq. (1)).

$$\begin{aligned} vec(\mathbf{B}_s) \quad = \quad & vec(\mathbf{E}_s) + \sum_{s' \in \mathcal{S}} \sum_{t \in \mathcal{T}_{ss'}} \frac{\epsilon_t}{k_s} (\mathbf{H}_t \otimes \mathbf{A}_t) vec(\mathbf{B}_{s'}) \\ & - \frac{\epsilon_t^2}{k_s k_{s'}} (\mathbf{H}_t \mathbf{H}_t^T \otimes \mathbf{D}_{st}) vec(\mathbf{B}_s) \end{aligned}$$

Rewriting the above equation using $\mathbf{P}_{ss'}$ and $\mathbf{Q}_s$ defined in Eq. (8) and Eq. (9) leads to:

$$(\mathbf{I} + \mathbf{Q}_s) vec(\mathbf{B}_s) \quad = \quad vec(\mathbf{E}_s) + \sum_{t \in \mathcal{T}_{ss'}} \mathbf{P}_{ss'} vec(\mathbf{B}_{s'}) \tag{20}$$

Eq. (20) gives the update equation for beliefs of nodes of type s. Here $\mathbf{I}$ is an identity matrix of appropriate dimensions. Stacking S such matrix equations together and rewriting using $\mathbf{e}, \mathbf{b}, \mathbf{P}$ and $\mathbf{Q}$ (defined in Section 4.3) gives the equation in Theorem 1.

■