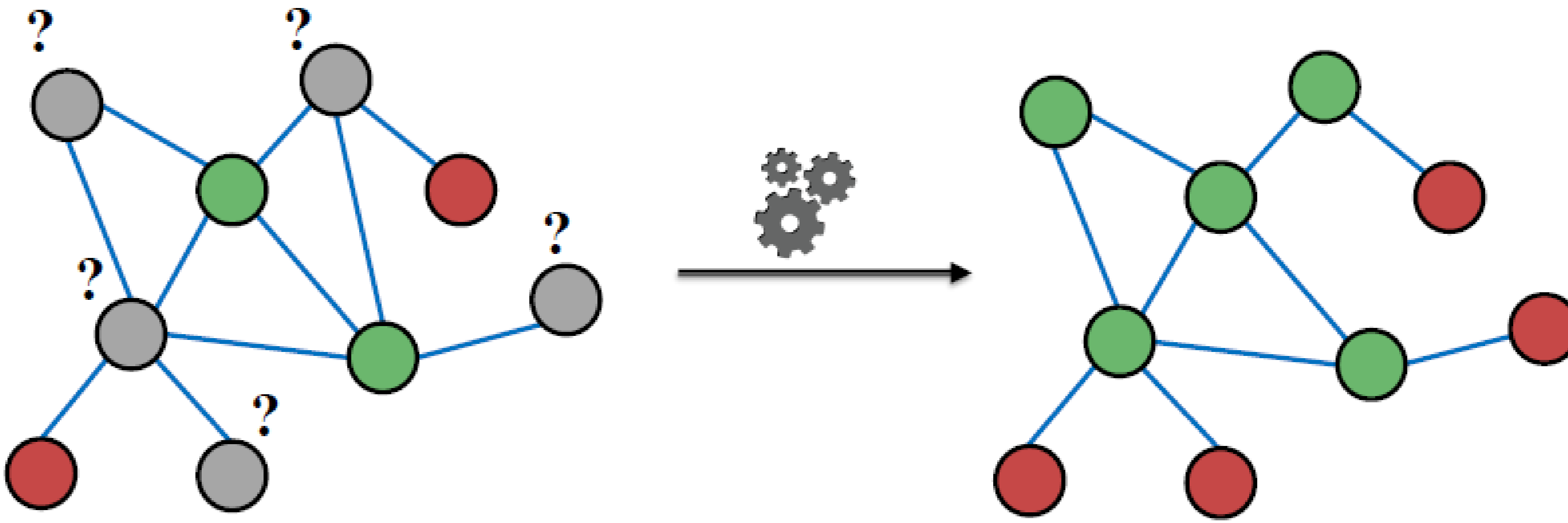
Message Passing and Node Classification

Slides Adapted from Leskovec's Lecture
CS224W Machine Learning with Graphs

Revisiting Node Classification



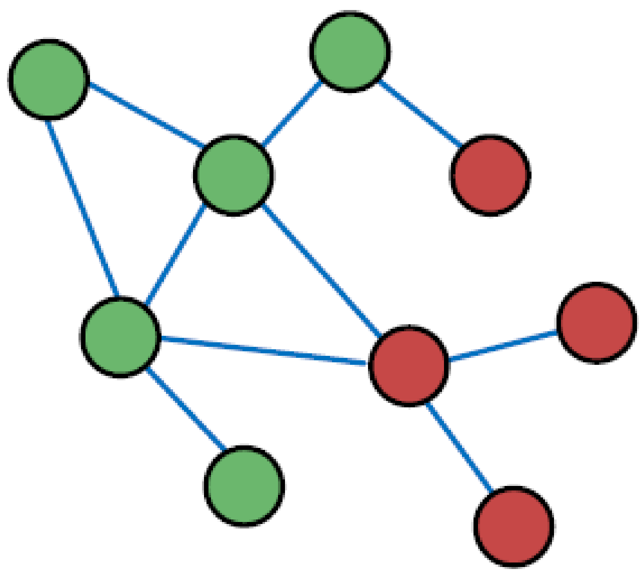
- Given labels of some nodes
- Let's predict labels of unlabeled nodes
- This is called semi-supervised node classification

An alternate framework for node classification – Message Passing

- **Intuition:** Correlations exist in networks.
  - In other words: Similar nodes are connected
  - **Key concept** is collective classification: Idea of assigning labels to all nodes in a network together
- **We will look at three techniques today:**
  - Relational classification
  - Iterative classification
  - Belief propagation

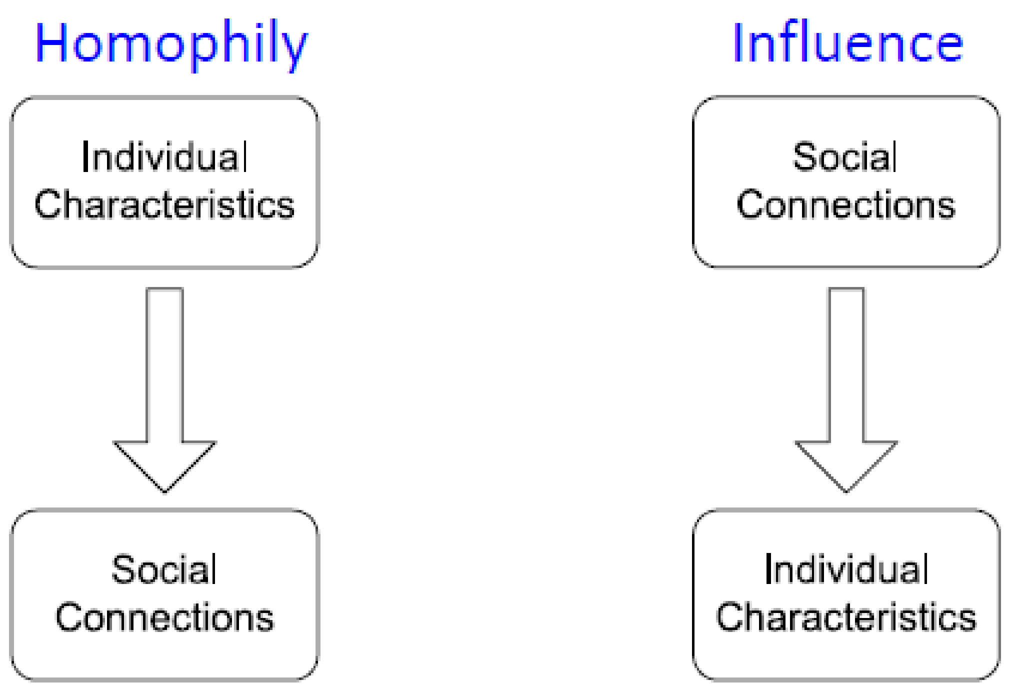
Correlations in Networks

- Individual behaviors are **correlated** in the network
- **Correlation**: nearby nodes have the same color (belonging to the same class)



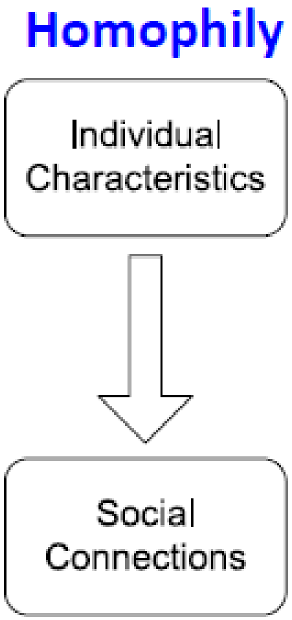
Correlations in Networks

- **Main types of dependencies that lead to correlation:**



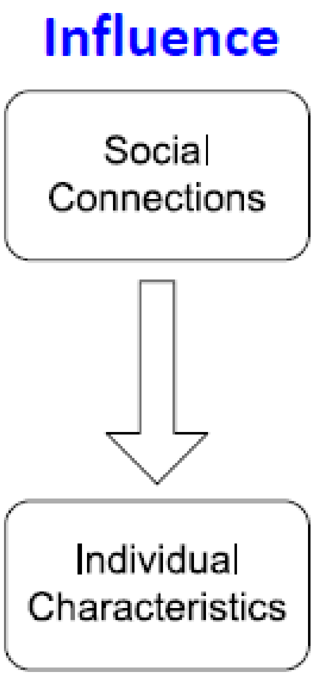
Homophily

- **Homophily**: The tendency of individuals to associate and bond with similar others
  - *"Birds of a feather flock together"*
  - It has been observed in a vast array of network studies, based on a variety of attributes (e.g., age, gender, organizational role, etc.)
  - **Example**: Researchers who focus on the same research area are more likely to establish a connection (meeting at conferences, interacting in academic talks, etc.)



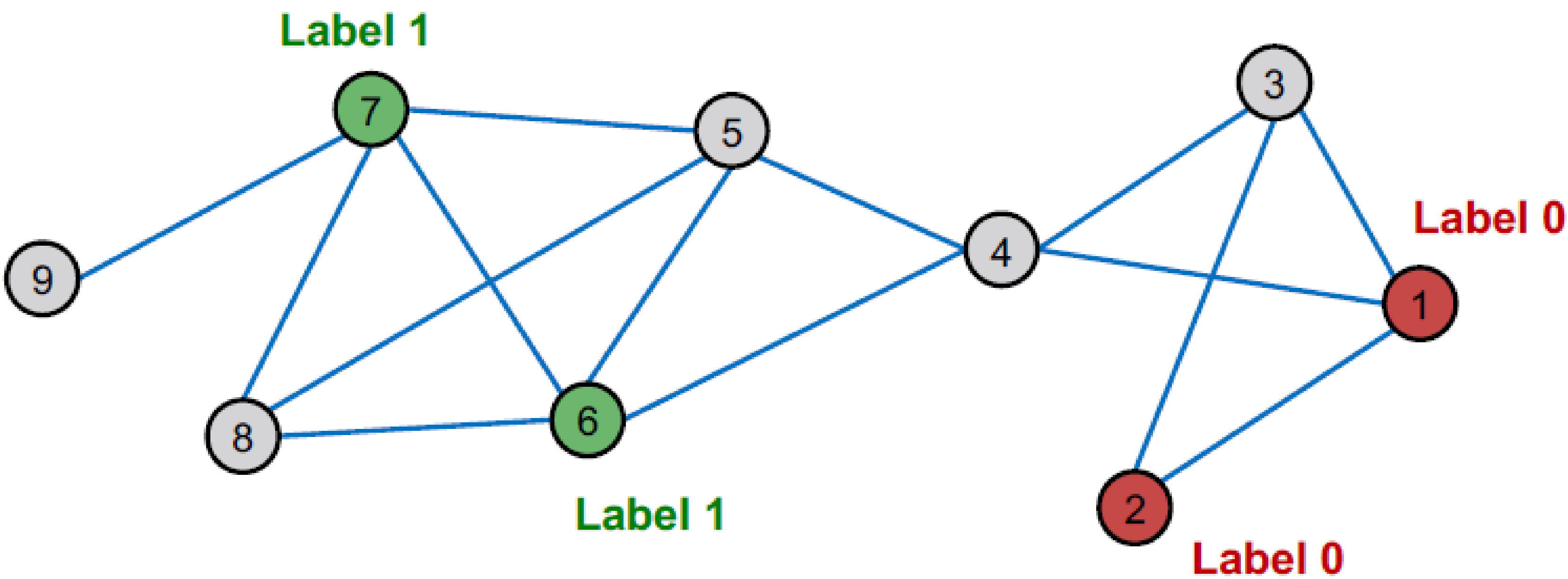
Influence

- **Influence**: Social connections can influence the individual characteristics of a person.
  - **Example**: I recommend my musical preferences to my friends, until one of them grows to like my same favorite genres!



Leveraging the correlation

- How do we leverage this correlation observed in networks to help predict node labels?



How do we predict the labels for the nodes in grey?

Motivation

- **Similar nodes are typically close together or directly connected in the network:**
  - **Guilt-by-association**: If I am connected to a node with label X, then I am likely to have label X as well.
  - **Example: Malicious/benign web page:** Malicious web pages link to one another to increase visibility, look credible, and rank higher in search engines

Motivation

- **Classification label** of a node v in network may depend on:
  - Features of v
  - Labels of the nodes in v's neighborhood
  - Features of the nodes in v's neighborhood

Collective Classification Overview

- **Intuition:** Simultaneous classification of interlinked nodes using correlations
- Probabilistic framework
- Markov Assumption: *the label Y_v of one node v depends on the labels of its neighbors N_v*

$$P(Y_v) = P(Y_v | N_v)$$

- Collective classification involves 3 steps:

Local Classifier	Relational Classifier	Collective Inference
• Assign initial labels	• Capture correlations between nodes	• Propagate correlations through network

Collective Classification Overview

Local Classifier

- Assign initial labels

Local Classifier: Used for initial label assignment

- Predicts label based on node attributes/features
- Standard classification task
- Does not use network information

Relational Classifier

- Capture correlations between nodes

Relational Classifier: Capture correlations

- Learns a classifier to label one node based on the labels and/or attributes of its neighbors
- This is where network information is used

Collective Inference

- Propagate correlations through network

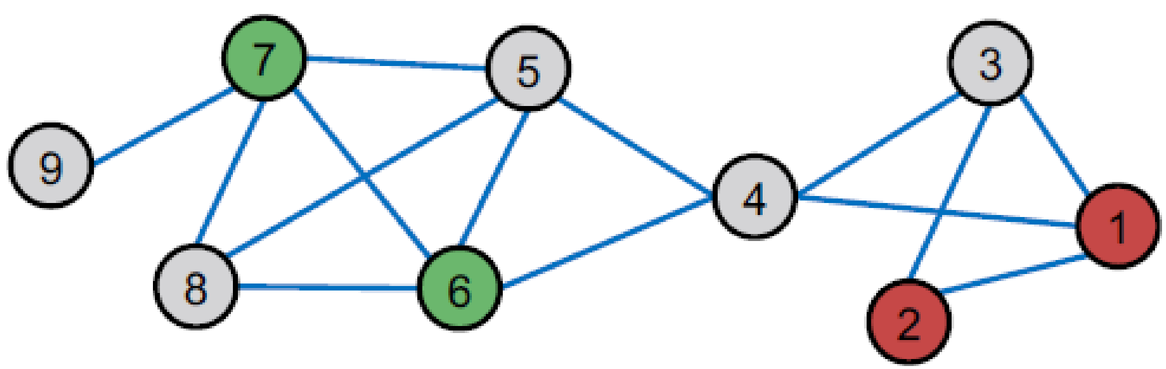
Collective Inference: Propagate the correlation

- Apply relational classifier to each node iteratively
- Iterate until the inconsistency between neighboring labels is minimized
- Network structure affects the final prediction

Problem Setting

- How to predict the labels Y_v for the unlabeled nodes v (in grey color)?
- Each node v has a feature vector f_v
- Labels for some nodes are given (1 for green, 0 for red)
- **Task:** Find $P(Y_v)$ given all features and the network

$P(Y_v) = ?$



Intuition and Techniques

- We focus on semi-supervised node classification
- Intuition is based on **homophily**: Similar nodes are typically close together or directly connected
- **Three techniques we will introduce:**
  - Relational classification
  - Iterative classification
  - Belief propagation

Probabilistic Relational Classifier

- **Basic idea:** Class probability Y_v of node v is a weighted average of class probabilities of its neighbors
- For **labeled nodes** v, initialize label Y_v with ground-truth label Y_v^*
- For **unlabeled nodes**, initialize $Y_v = 0.5$
- **Update** all nodes in a random order until convergence or until maximum number of iterations is reached

Probabilistic Relational Classifier

- **Update** for each node v and label c (e.g. 0 or 1)

$$P(Y_v = c) = \frac{1}{\sum_{(v,u)\in E} A_{v,u}} \sum_{(v,u)\in E} A_{v,u} P(Y_u = c)$$

  - If edges have strength/weight information, $A_{v,u}$ can be the edge weight between v and u
  - $P(Y_v = c)$ is the probability of node v having label c

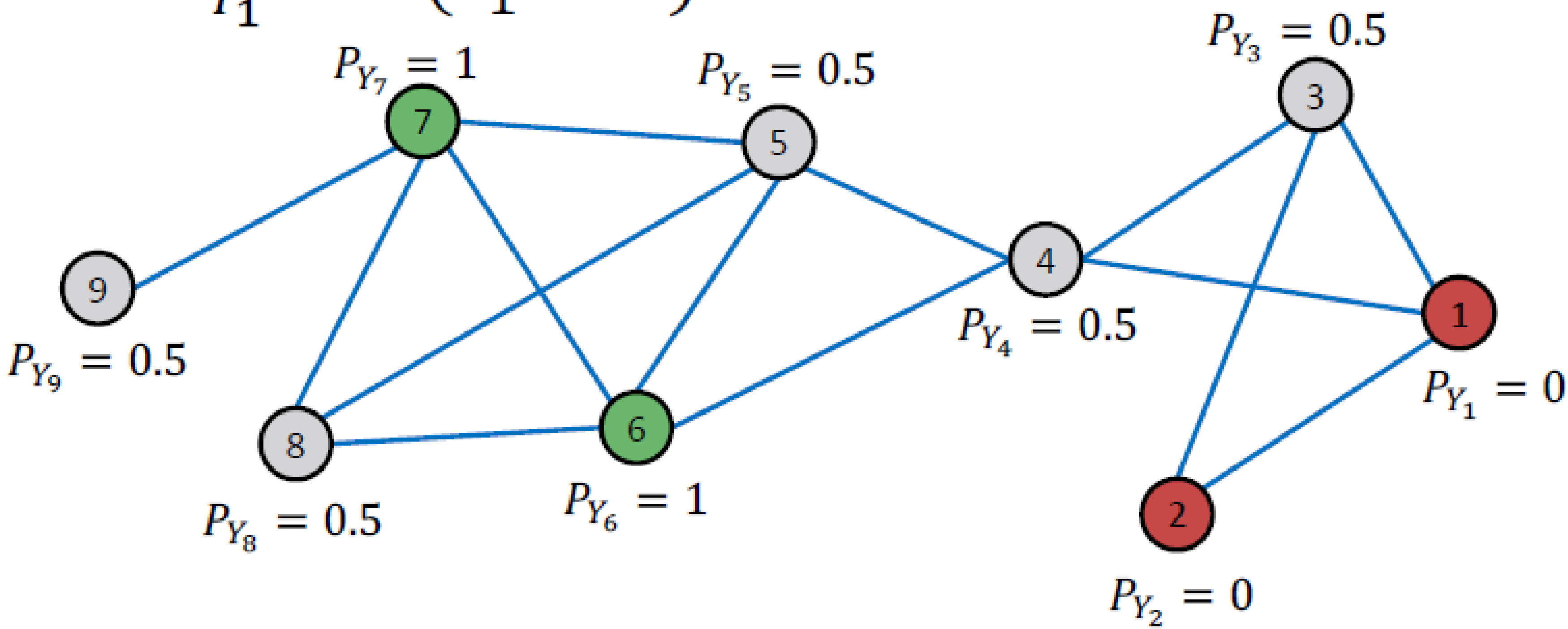
- Challenges:
  - Convergence is not guaranteed
  - Model cannot use node feature information

Example

Initialization:

- All labeled nodes with their labels
- All unlabeled nodes 0.5 (belonging to class 1 with probability 0.5)

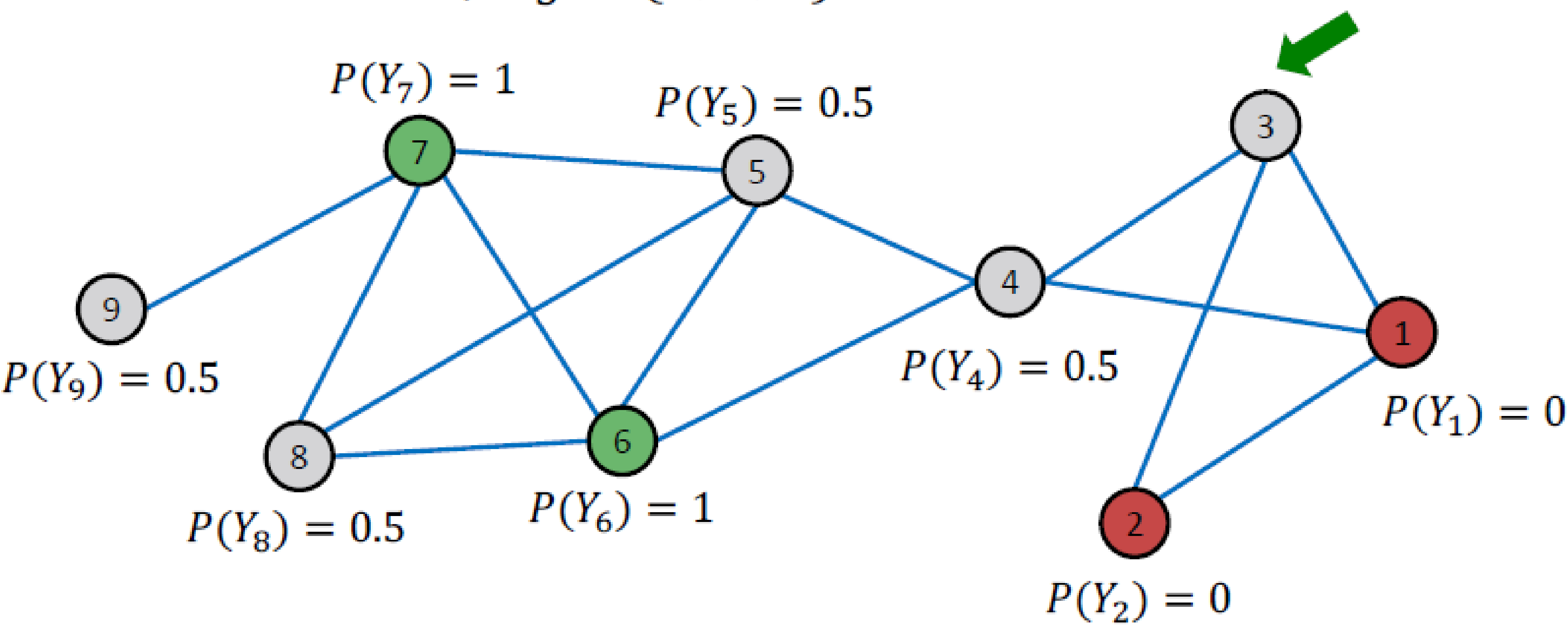
Let $P_{Y_1} = P(Y_1 = 1)$



1st Iteration – Update of node 3

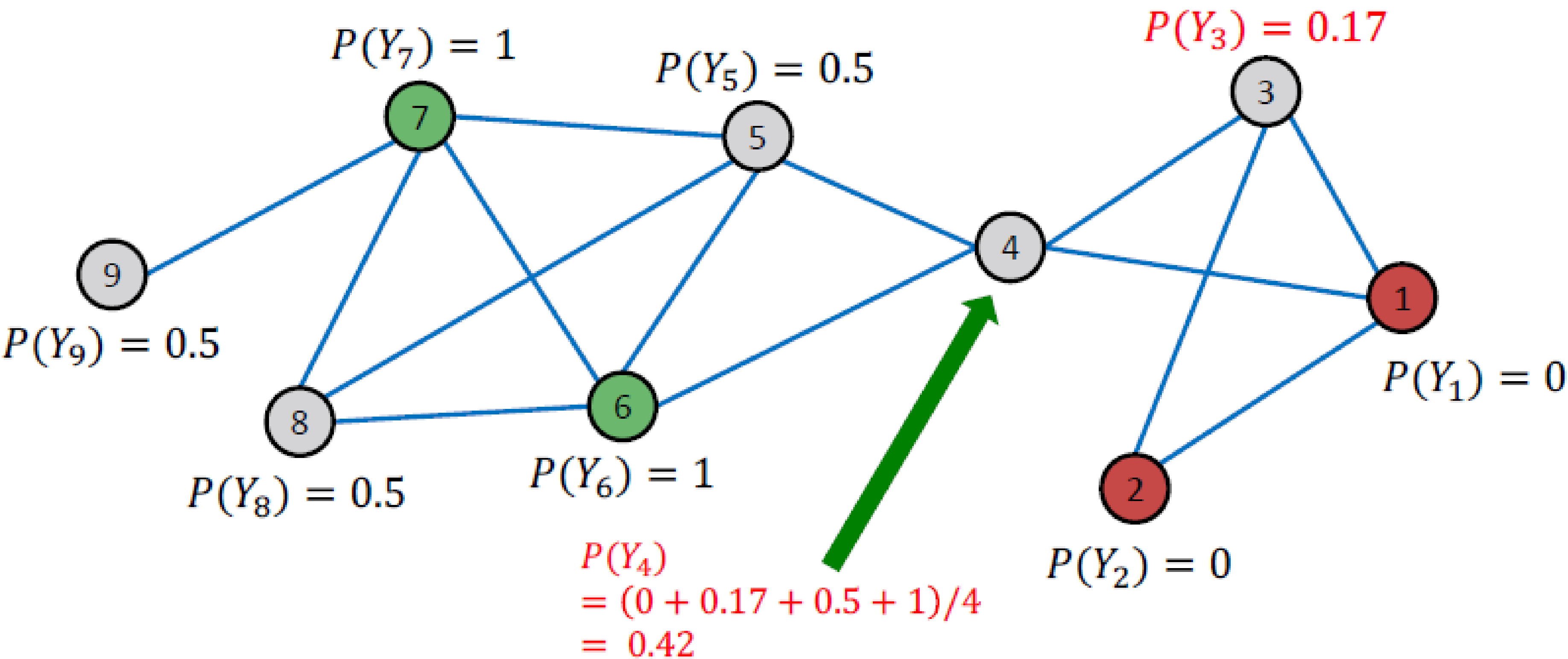
- Update for the 1st Iteration:
  - For node 3, $N_3 = \{1, 2, 4\}$

$P(Y_3) = (0 + 0 + 0.5)/3 = 0.17$



1st Iteration – Update of Node 4

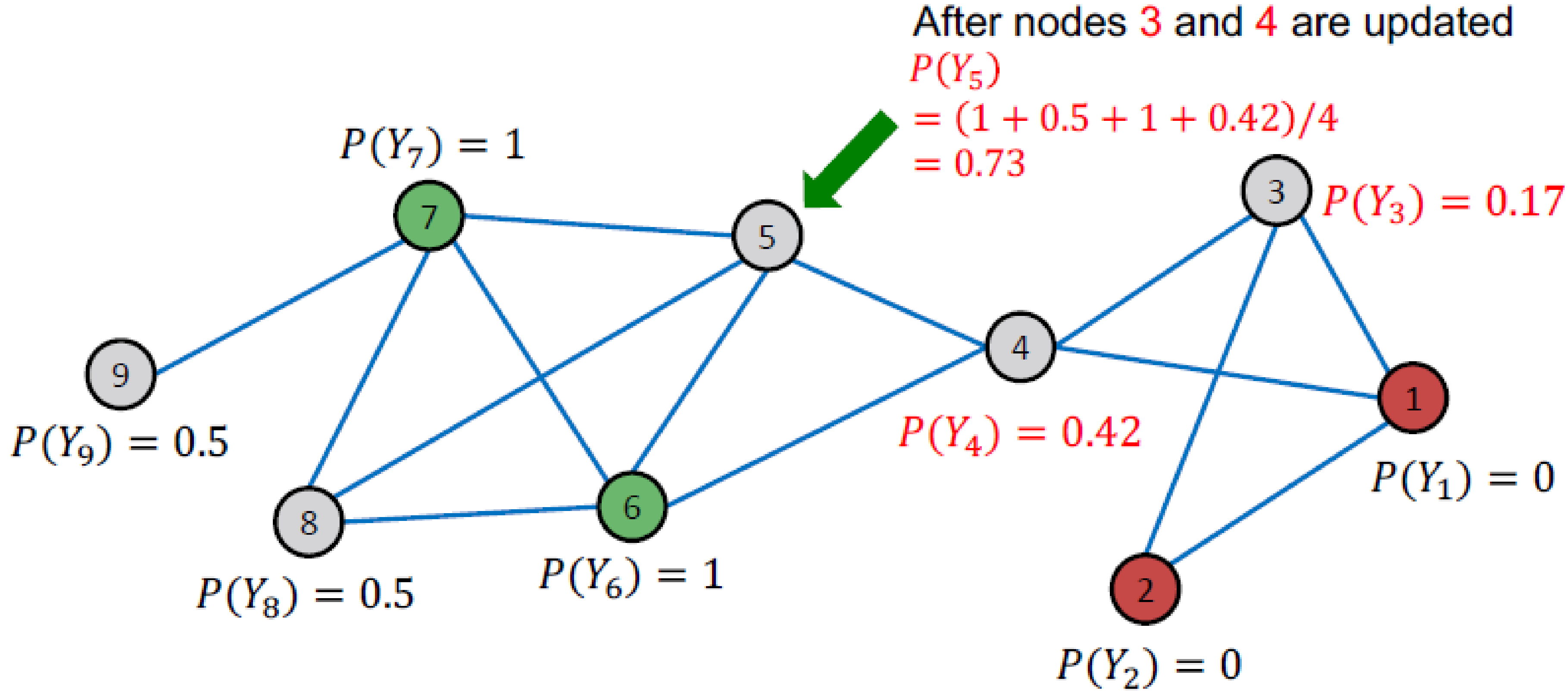
- Update for the 1st Iteration:
  - For node 4, $N_4 = \{1, 3, 5, 6\}$



After Node 3 is updated

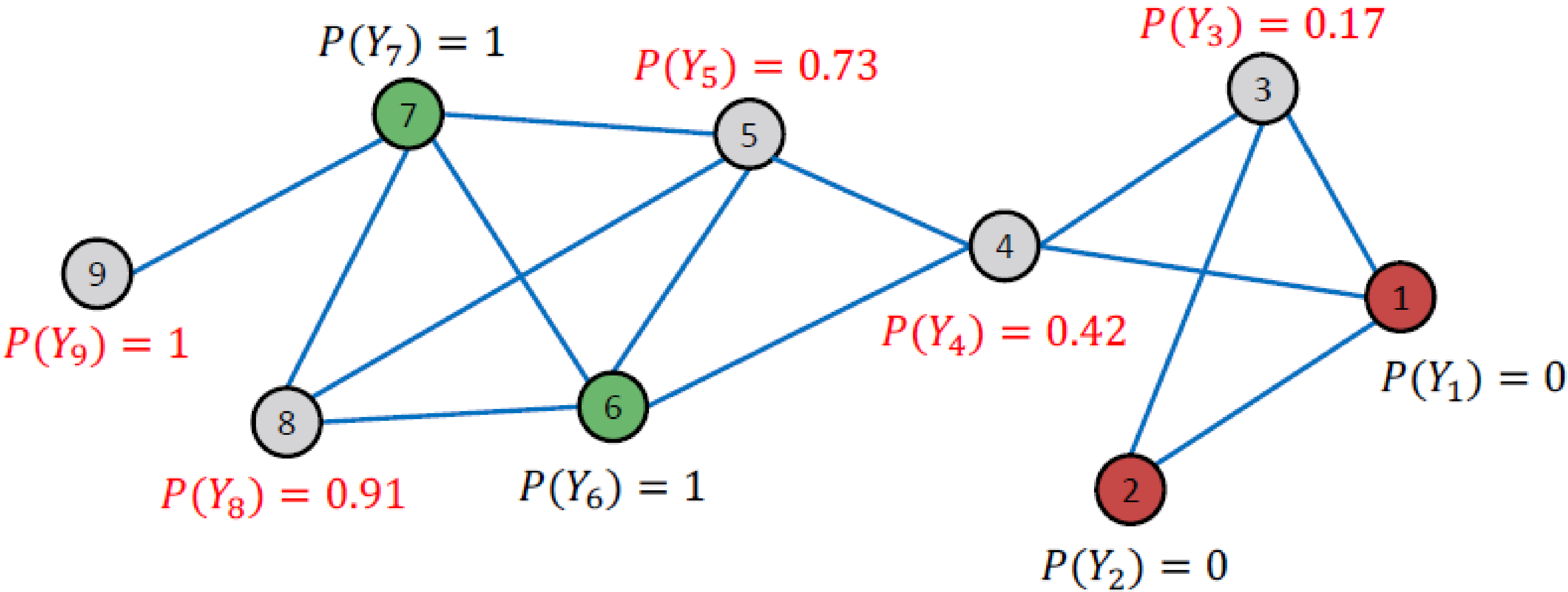
1^{st} Iteration – Update of node 5

- Update for the 1^{st} Iteration:
  - For node 5, $N_5 = \{4, 6, 7, 8\}$

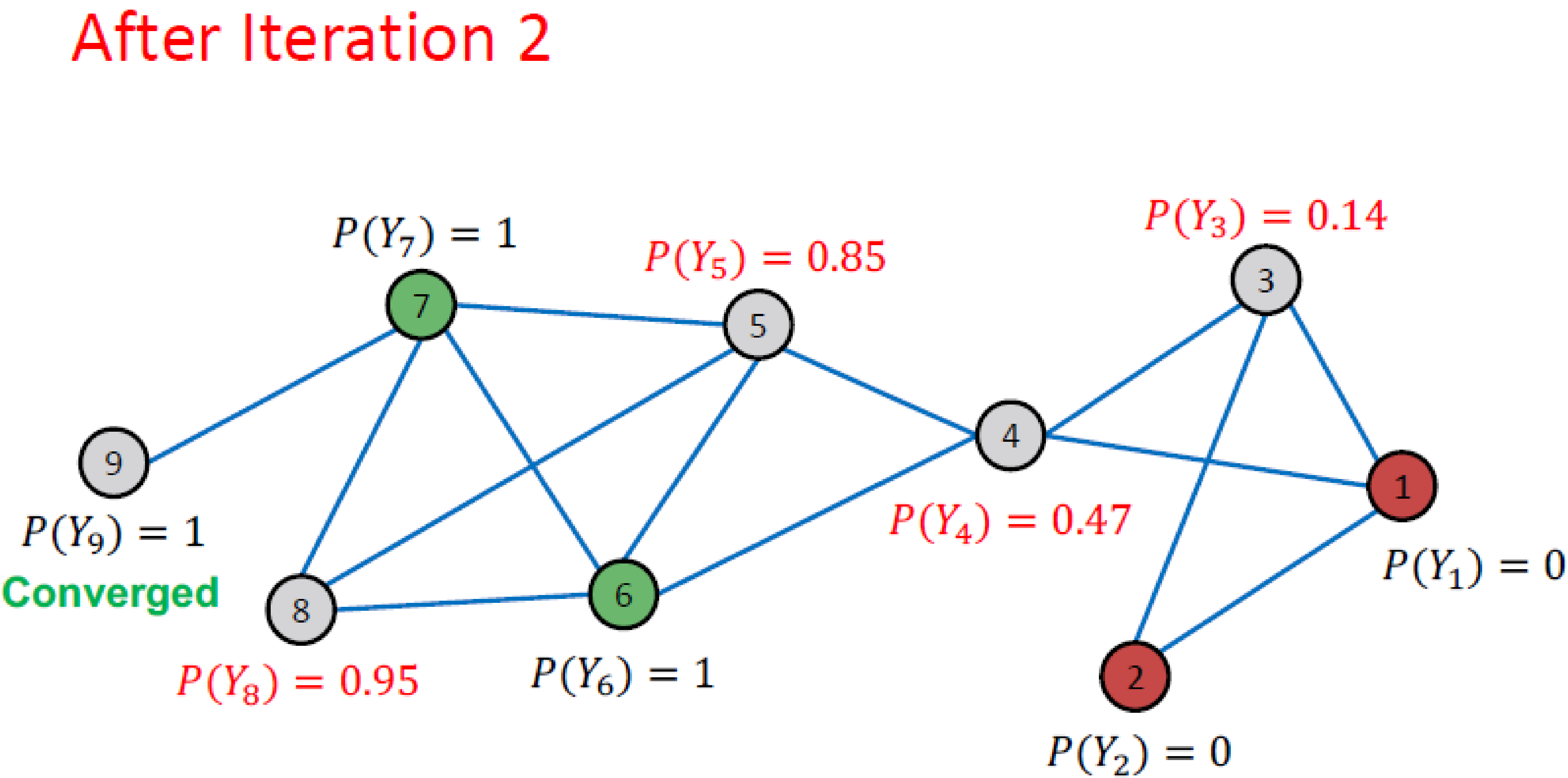


After 1st Iteration

After Iteration 1 (a round of updates for all unlabeled nodes)

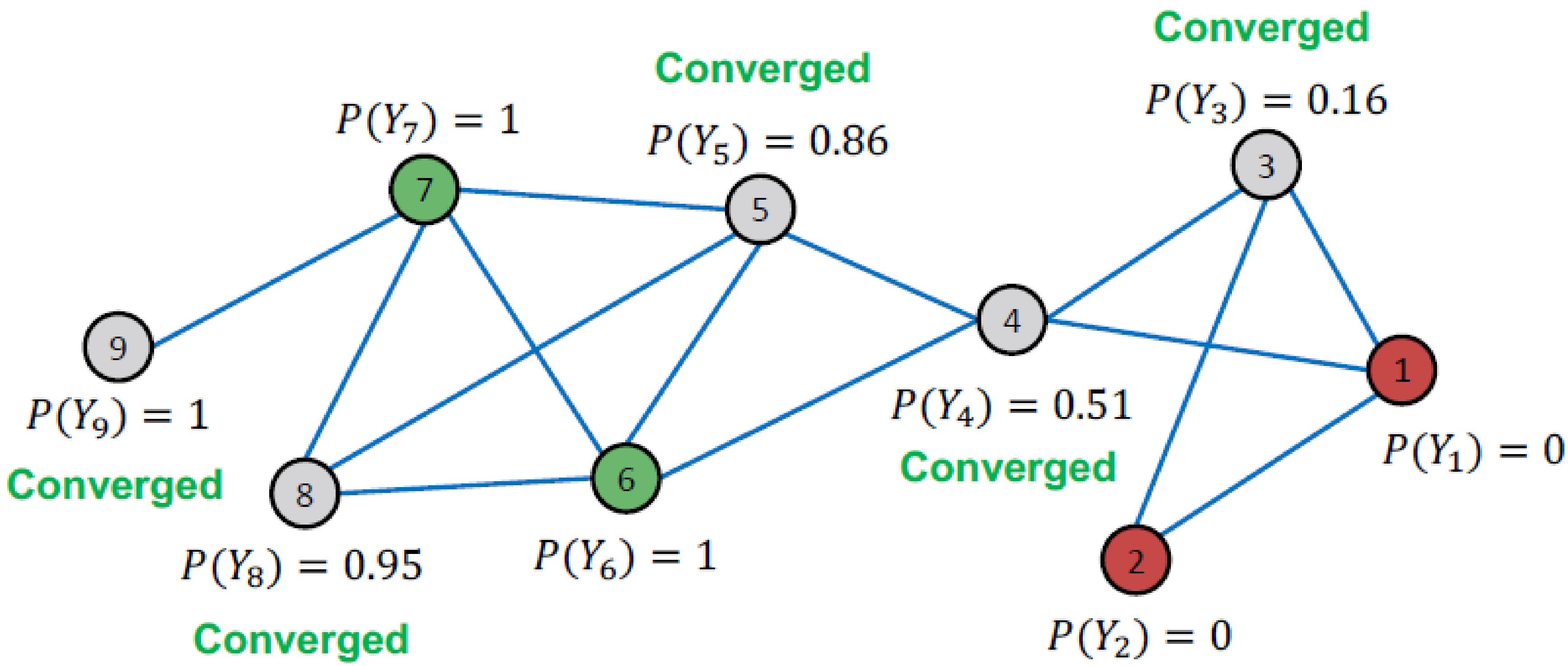


After 2nd Iteration



Convergence after 4 iterations

- All scores stabilize after 4 iterations. We therefore predict:
  - **Nodes 4, 5, 8, 9 belong to class 1** ($P_{Y_v} > 0.5$)
  - **Nodes 3 belong to class 0** ($P_{Y_v} < 0.5$)



Iterative Classification

- **Relational classifiers do not use node attributes. How can one leverage them?**

- **Main idea of iterative classification:** Classify node v based on its **attributes** $\boldsymbol{f_v}$ as well as **labels** $\boldsymbol{z_v}$ of neighbor set $\boldsymbol{N_v}$

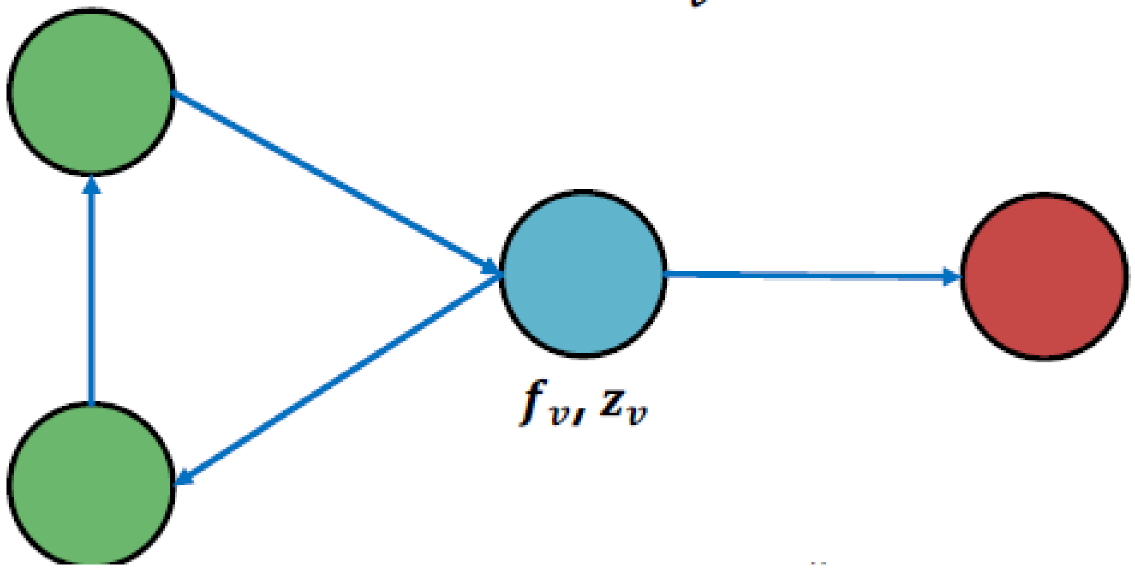
Iterative Classification Approach

- **Input: Graph**
  - f_v : feature vector for node v
  - Some nodes v are labeled with Y_v
- **Task:** Predict label of unlabeled nodes
- **Approach: Train two classifiers:**
- $\phi_1(f_v)$ = Predict node label based on node feature vector f_v
- $\phi_2(f_v, z_v)$ = Predict label based on node feature vector f_v and summary z_v of labels of v's neighbors.

Computing Summary Z_v

How do we compute the summary z_v of labels of v's neighbors N_v?

- Ideas: $\boldsymbol{z_v}$ = **vector**
  - Histogram of the number (or fraction) of each label in N_v
  - Most common label in N_v
  - Number of different labels in N_v



Iterative Classifier Approach

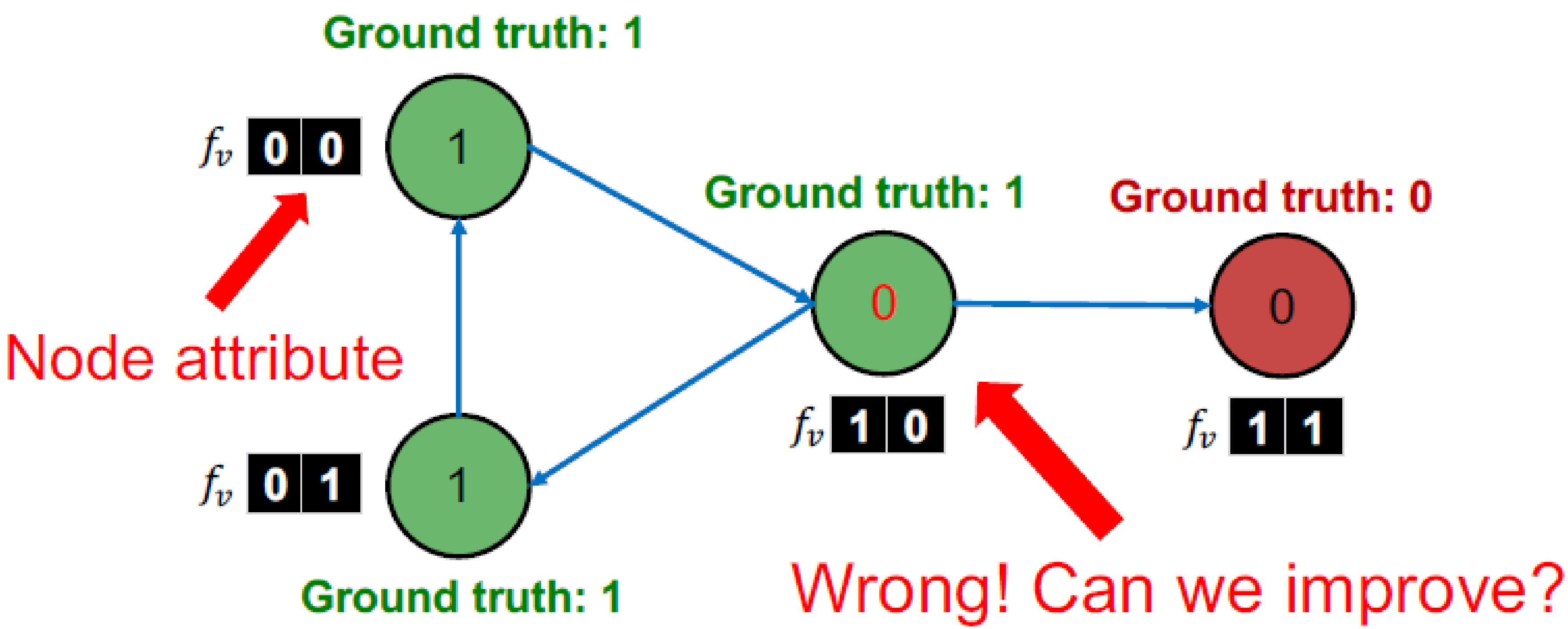
- **Phase 1: Classify based on node attributes alone**
  - On a **training set**, train classifier (e.g., linear classifier, neural networks, …):
  - $\phi_1(f_v)$ to predict Y_v based on f_v
  - $\phi_2(f_v, z_v)$ to predict Y_v based on f_v and summary z_v of labels of v's neighbors
- **Phase 2: Iterate till convergence**
  - On **test set**, set labels Y_v based on the classifier ϕ_1, compute z_v and predict the labels with ϕ_2
  - **Repeat** for each node v
    - Update z_v based on Y_u for all $u \in N_v$
    - Update Y_v based on the new z_v (ϕ_2)
  - Iterate until class labels stabilize or max number of iterations is reached
  - Note: Convergence is not guaranteed

Example Web Page Classification

- **Input:** Graph of web pages
- **Node:** Web page
- **Edge:** Hyper-link between web pages
  - **Directed edge:** a page points to another page
- **Node features:** Webpage description
  - For simplicity, we only consider 2 binary features
- **Task:** Predict the topic of the webpage

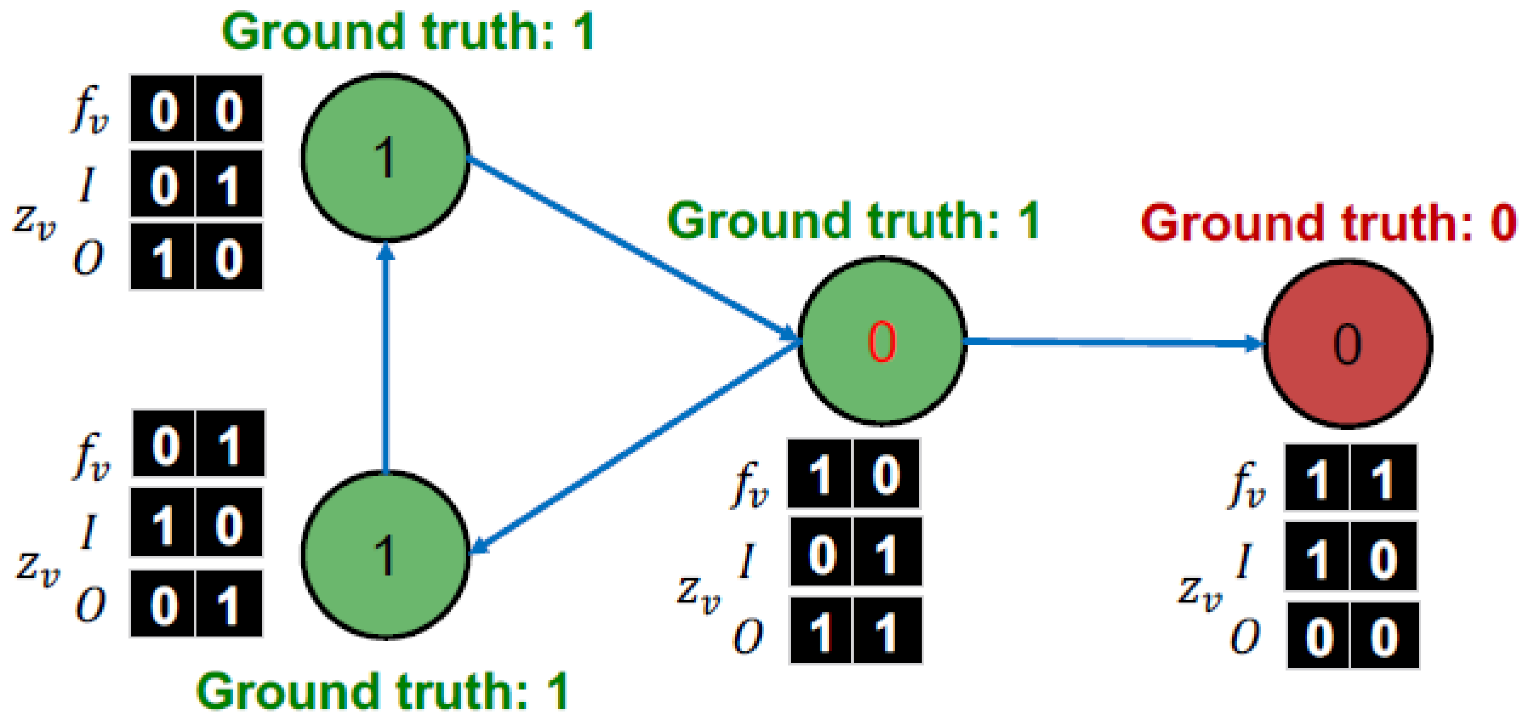
Example Web Page Classification

- Baseline: train a classifier (e.g., linear classifier) to classify pages based on binary node attributes.



Example: Web Page Classification

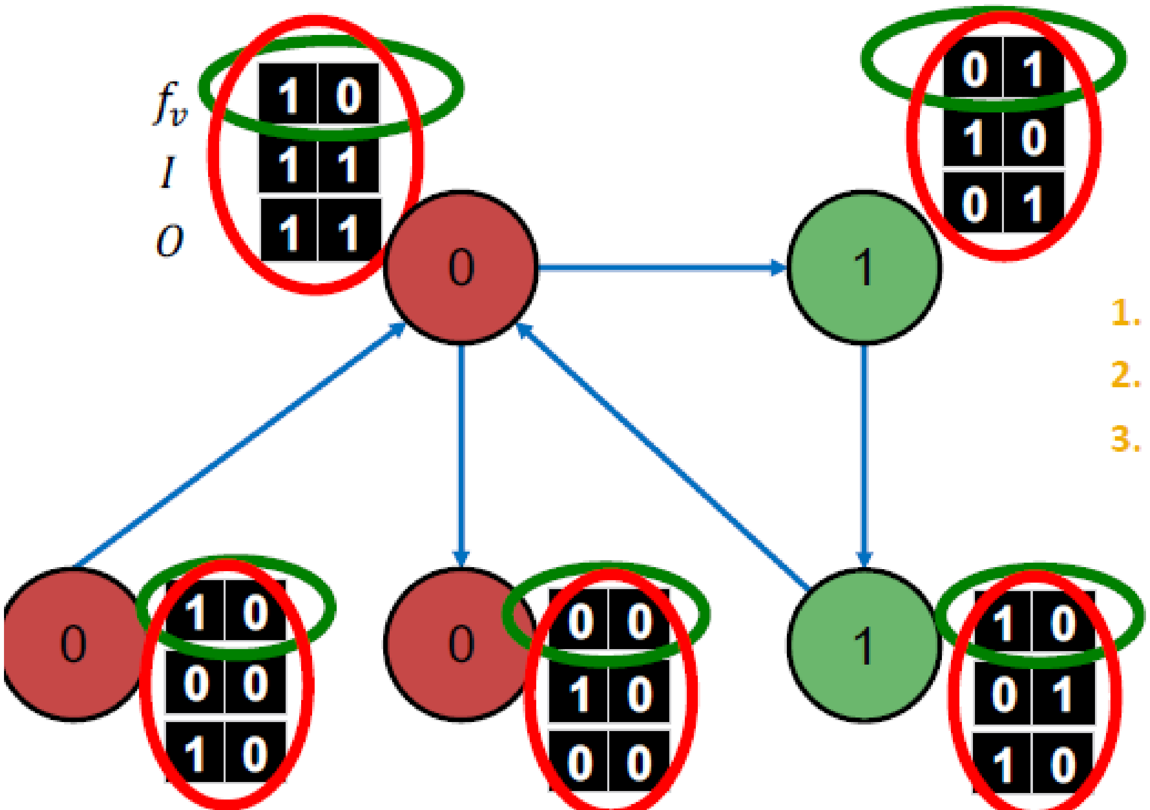
- Each node maintains vectors $\mathbf{z}_v$ of neighborhood labels:
  I = **Incoming** neighbor label information vector
  O= **Outgoing** neighbor label information vector
- $I_0 = 1$ if at least one of the incoming pages is labelled 0.
  Similar definitions for I_1, O_0, and O_1



Include network features

Iterative Classifier Step 1

- On a different **training set**, train two classifiers:
  - Node attribute vector only (green circles): ϕ_1
  - Node attribute and link vectors (red circles): ϕ_2

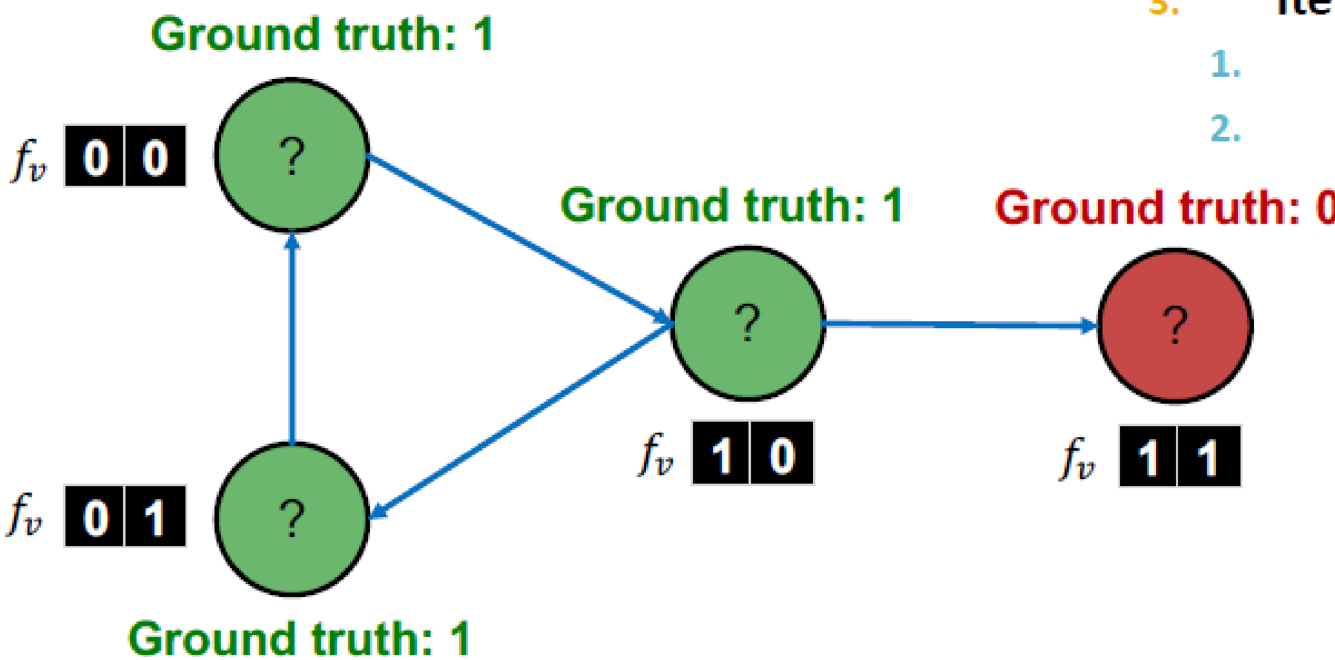


1. **Train classifier**
2. **Apply classifier to test set**
3. **Iterate**
   1. **Update relational features z_v**
   2. **Update label Y_v**

Iterative Classifier Step 2

- On the **test set**:
  - Use trained node feature vector classifier ϕ_1 to set Y_v

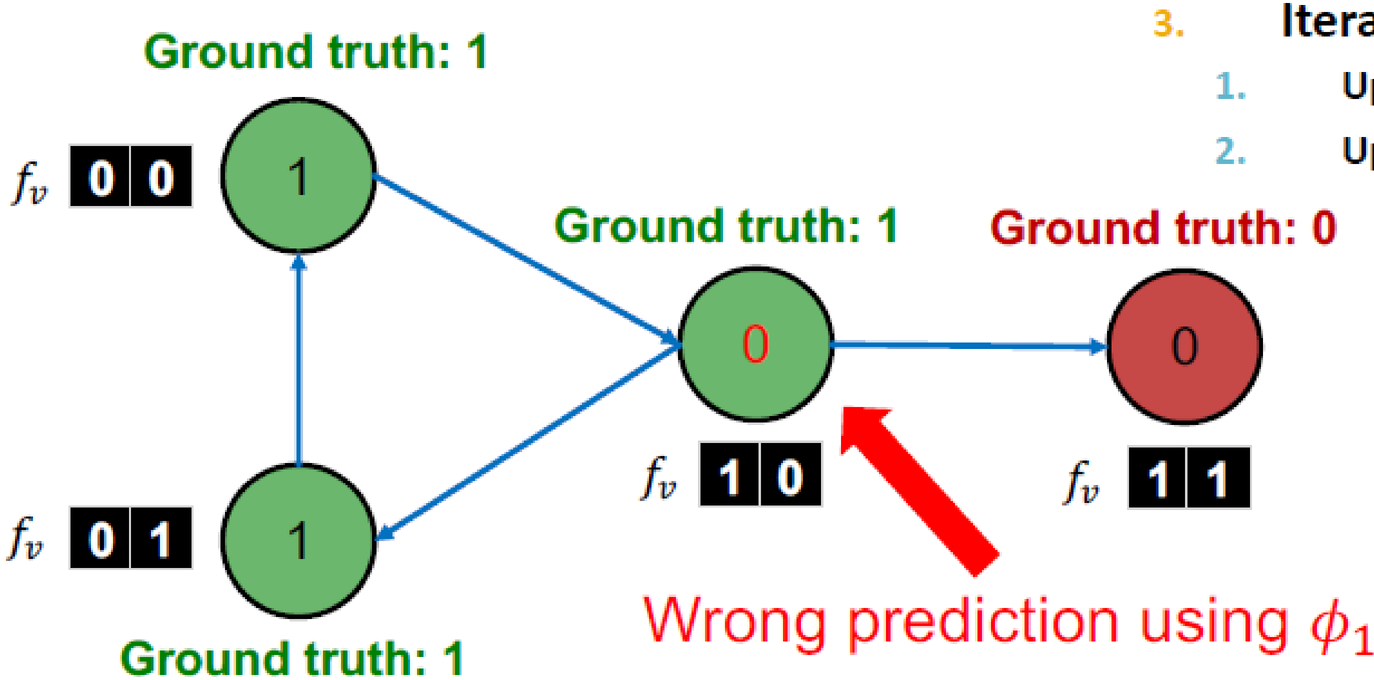
1. Train classifier
2. **Apply classifier to test set**
3. **Iterate**
   1. **Update relational features z_v**
   2. **Update label Y_v**



Iterative Classifier Step 2

- On the **test set**:
  - Use trained node feature vector classifier ϕ_1 to set Y_v

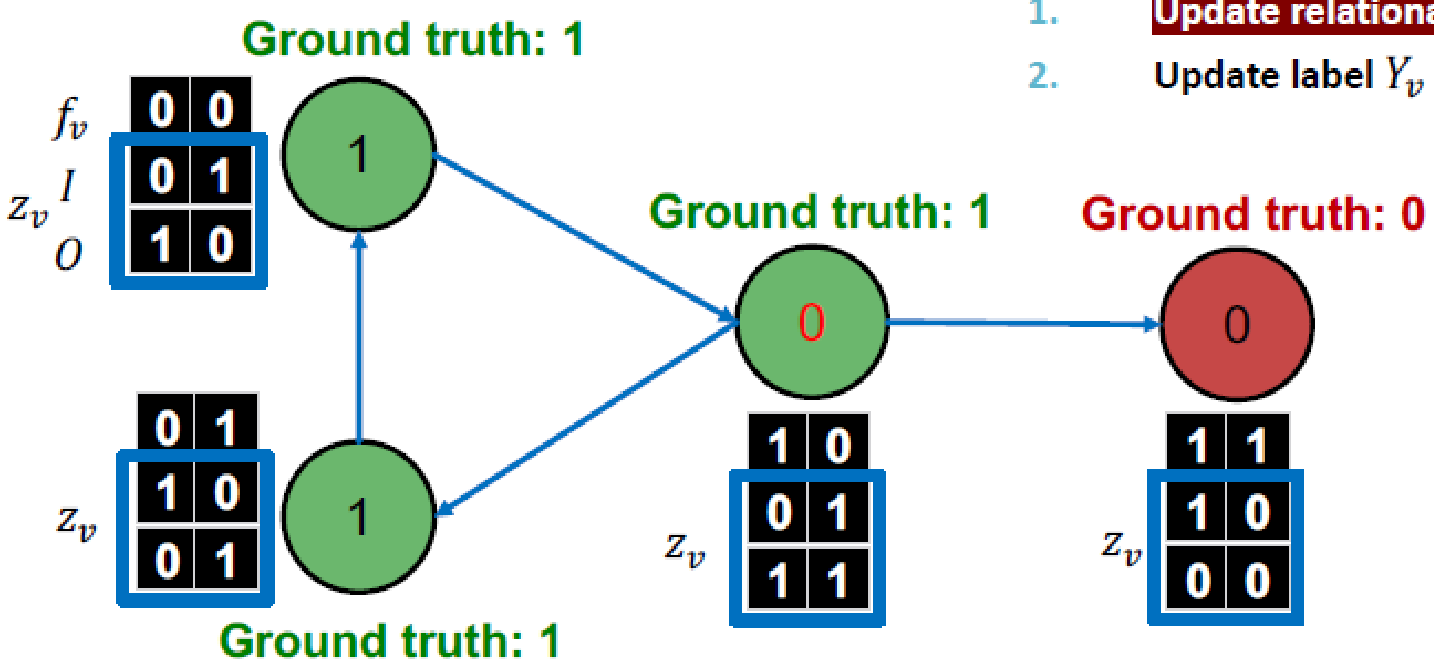
1. Train classifier
2. **Apply classifier to test set**
3. **Iterate**
   1. **Update relational features z_v**
   2. **Update label Y_v**



Iterative Classifier Step 3a

- Update z_v for all nodes

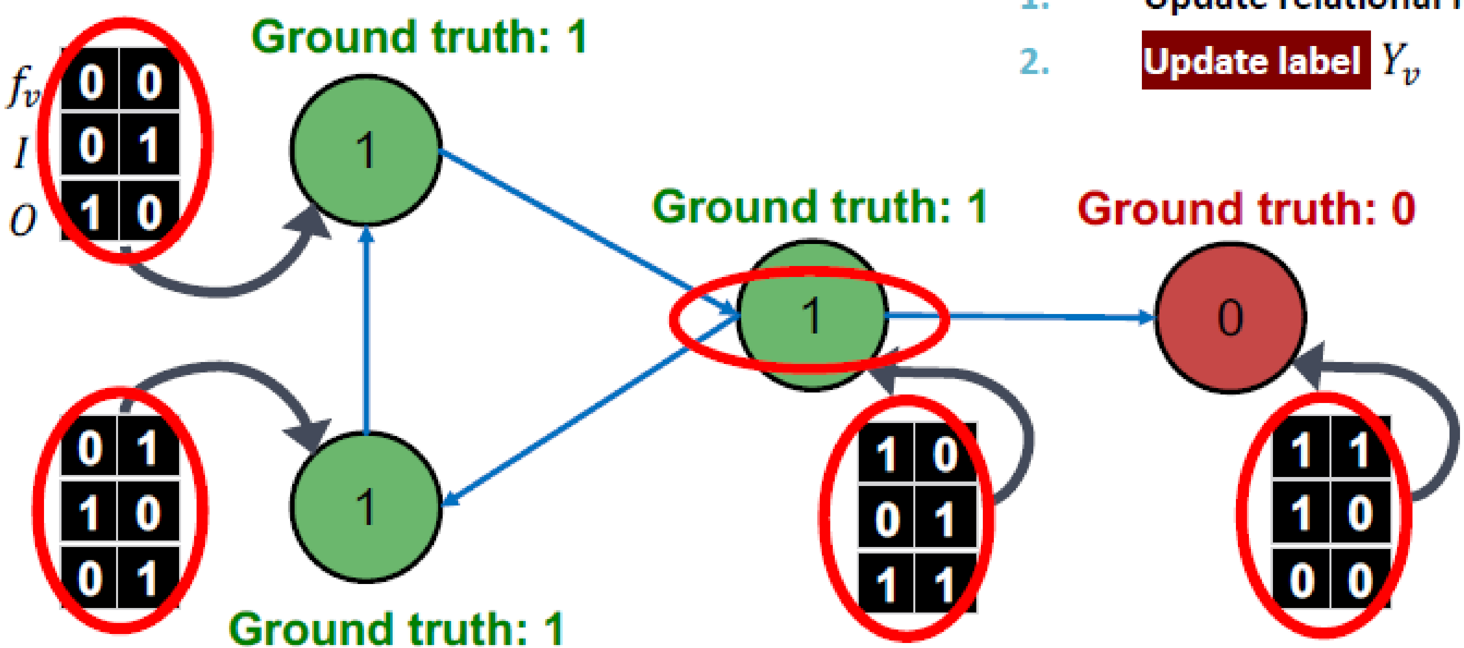
1. Train classifier
2. Apply classifier to test
3. Iterate
   1. Update relational features z_v
   2. Update label Y_v



Iterative Classifier Step 3b

- **Re-classify all nodes with ϕ_2**

1. Train classifier
2. Apply classifier to test
3. Iterate
    1. Update relational features z_v
    2. **Update label** Y_v

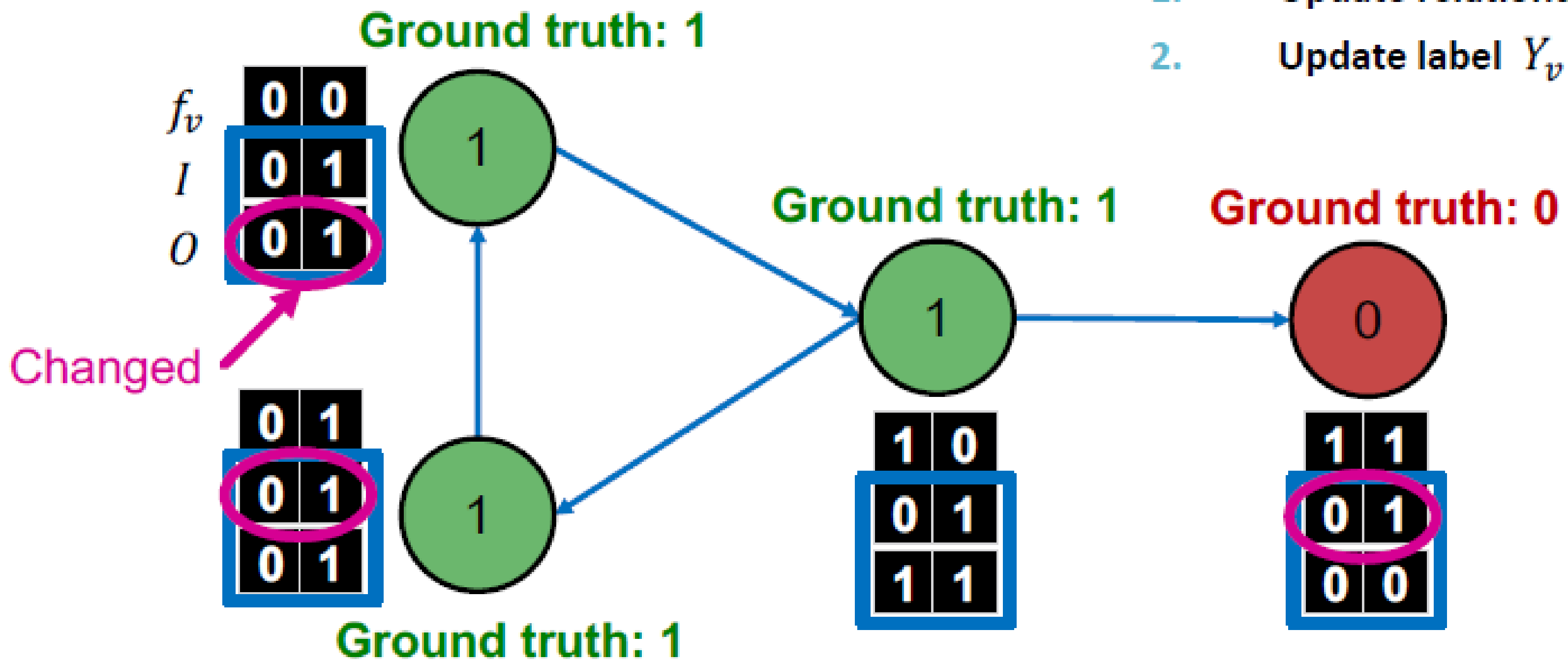


Now it's correct prediction!

Iterative Classifier Step Iterate

- Continue until convergence
  - Update z_v
  - Update $Y_v = \phi_2(f_v, z_v)$

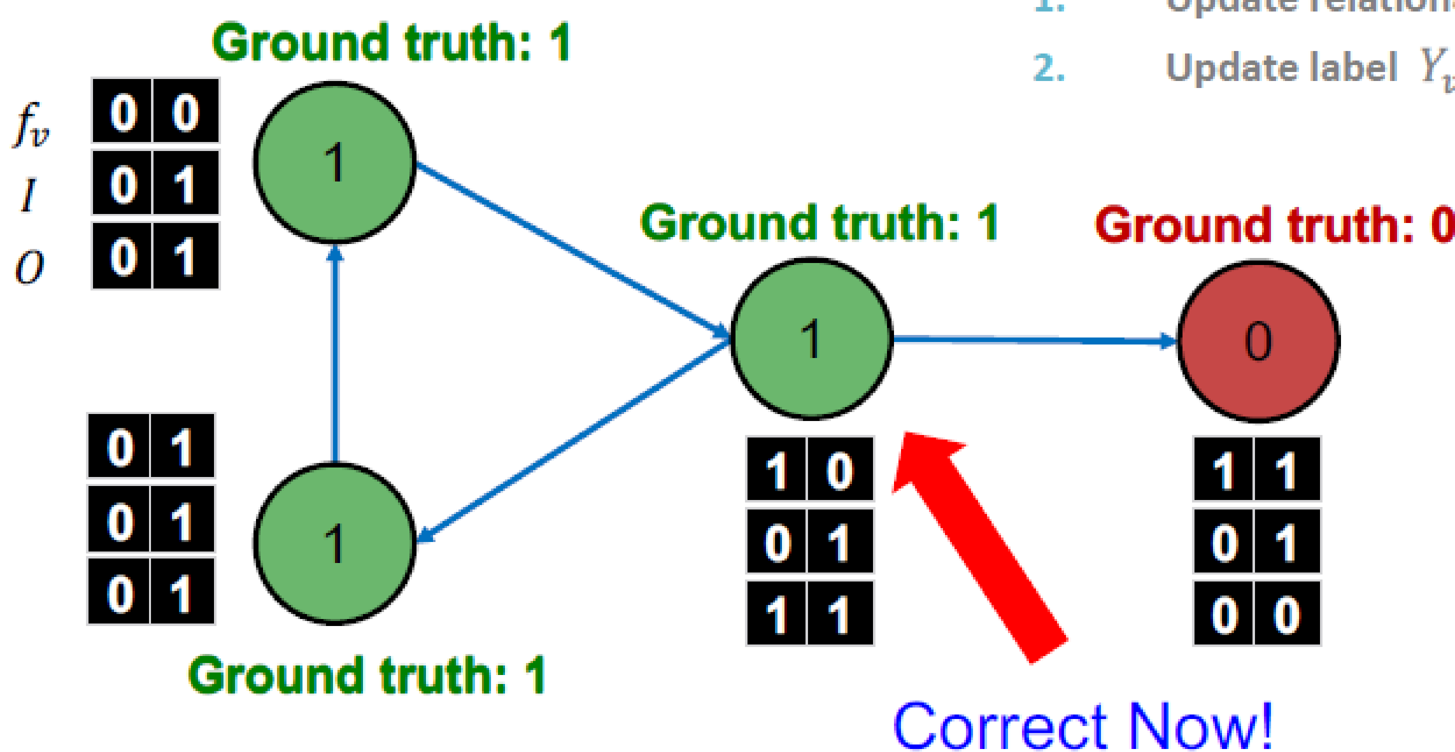
1. Train classifier
2. Apply classifier to test
3. **Iterate**
   1. Update relational features z_v
   2. Update label Y_v



Iterative Classifier Prediction

- Stop iteration
  - After convergence or when maximum iterations are reached

1. Train classifier
2. Apply classifier to test set
3. Iterate
   1. Update relational features z_v
   2. Update label Y_v



Loopy Belief Propagation

- Belief Propagation is a dynamic programming approach to **answering probability queries in a graph** (e.g. probability of node v belonging to class 1)
- Iterative process in which neighbor nodes "talk" to each other, **passing messages**

"I (node v) believe you (node u) belong to class 1 with likelihood …"

- **When consensus is reached, calculate final belief**

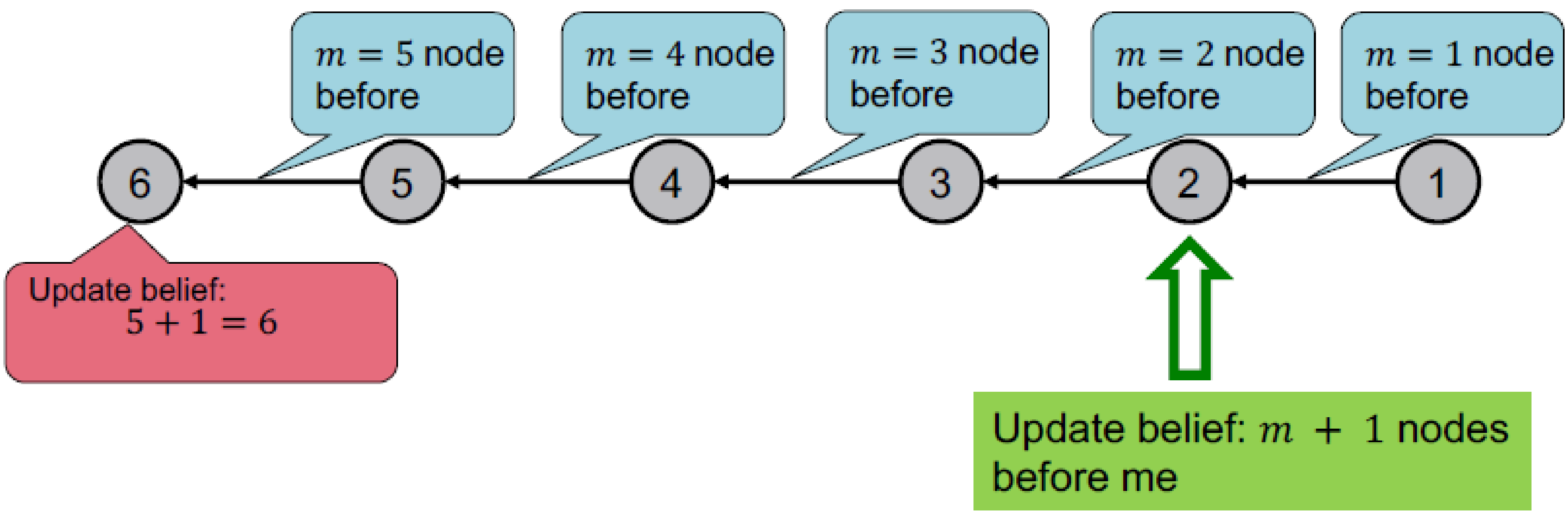
Message Passing Basics

Task: Count the number of nodes in a graph

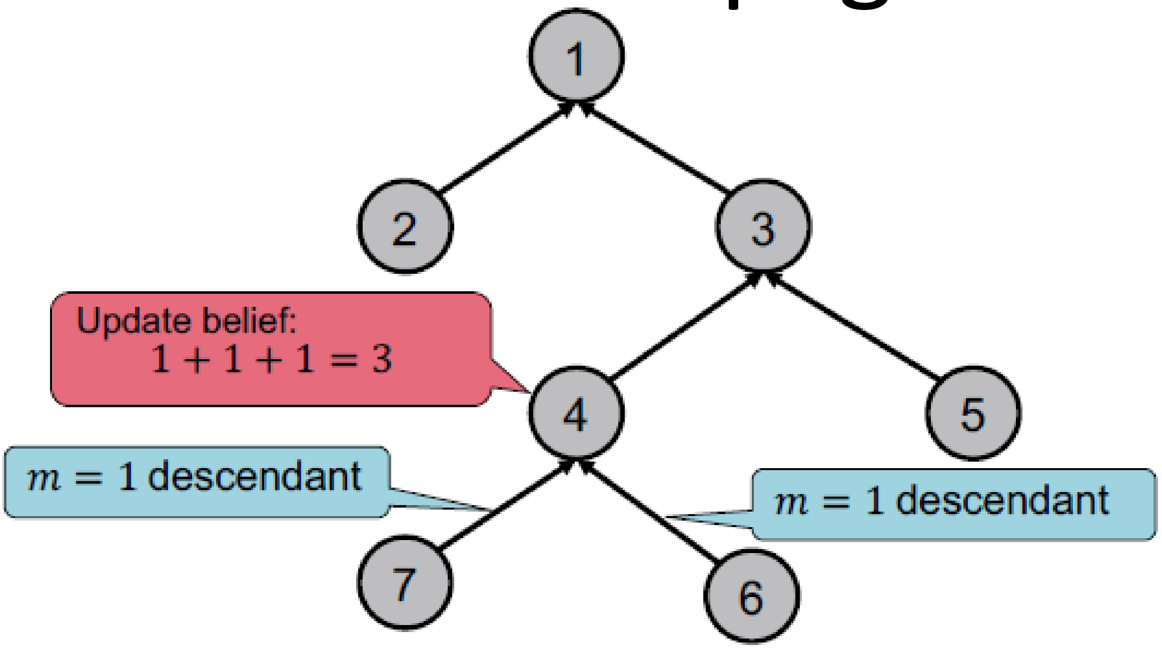
Condition: Each node can only interact (pass message) with its neighbors

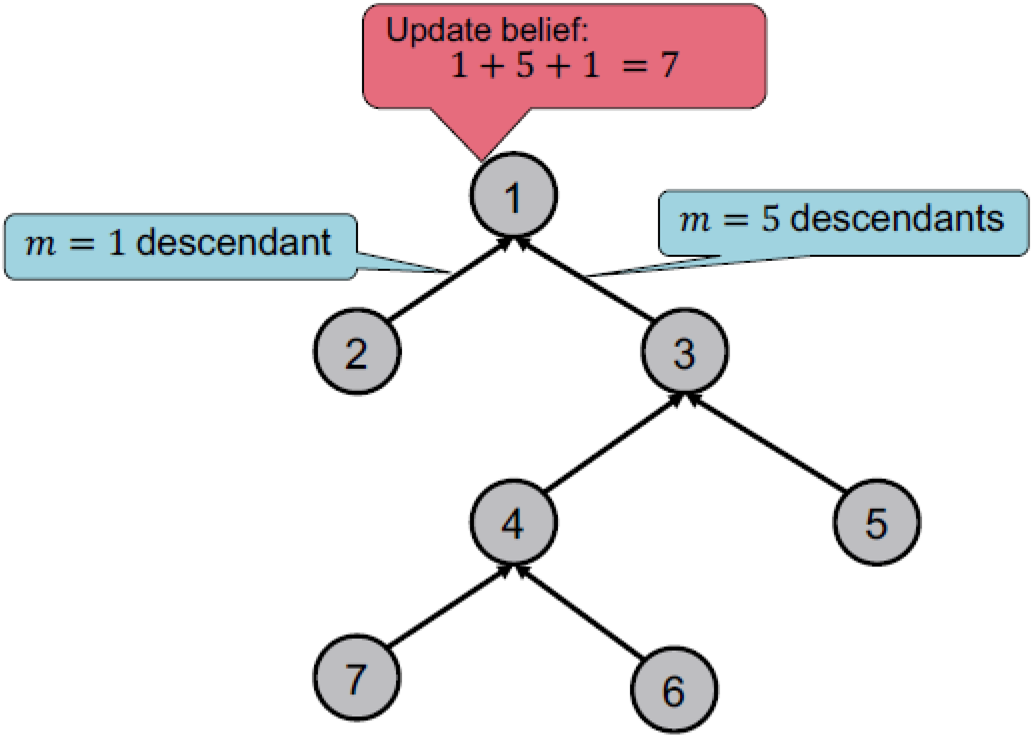
Solution: Each node listens to the message from its neighbor, updates it, and passes it forward

m: the message

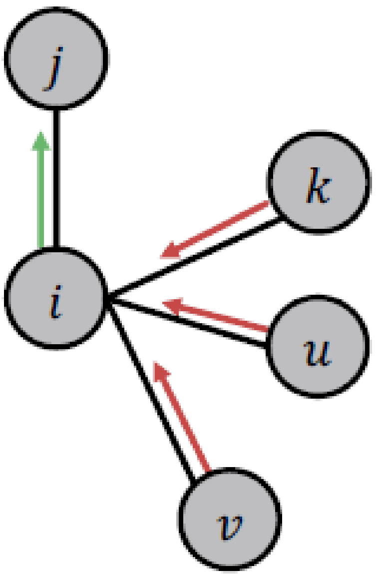


Belief Propagation in a tree





Loopy Belief Propagation



What message will i send to j?

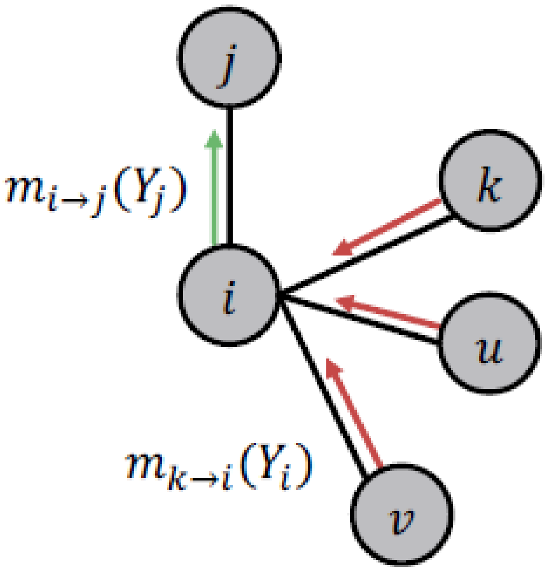
- It depends on what i hears from its neighbors
- Each neighbor passes a message to i its beliefs of the state of i

I (node i) believe that you (node j) belong to class Y_j with probability …

Notation

- ***Label-label potential matrix*** ψ : Dependency between a node and its neighbor. $\psi(Y_i, Y_j)$ is proportional to the probability of a node j being in class Y_j given that it has neighbor i in class Y_i.
- ***Prior belief*** ϕ : $\phi(Y_i)$ is proportional to the probability of node i being in class Y_i.
- $m_{i \to j}(Y_j)$ is i's message / estimate of j being in class Y_j.
- $\mathcal{L}$ is the set of all classes/labels

Loopy Belief Propagation



1. Initialize all messages to 1
2. Repeat for each node:

Label-label potential

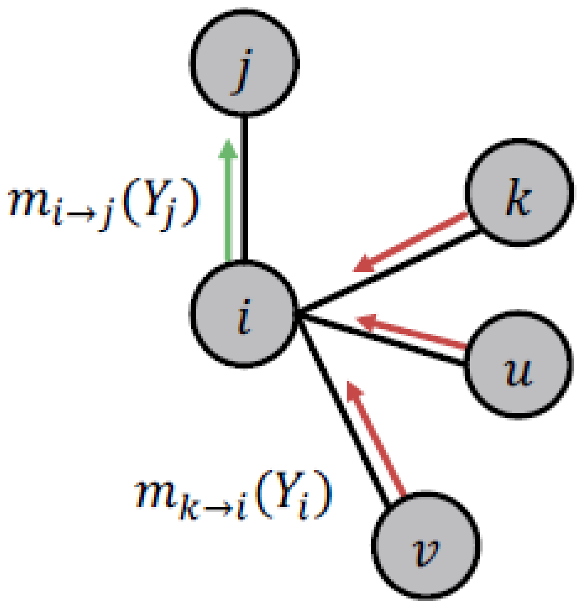
All messages sent by neighbors from previous round

$$m_{i\to j}(Y_j) = \sum_{Y_i \in \mathcal{L}} \psi(Y_i, Y_j)\, \phi_i(Y_i) \prod_{k \in N_i \setminus j} m_{k\to i}(Y_i), \forall Y_j \in \mathcal{L}$$

Sum over all states

Prior

Loopy Belief Propagation



After convergence:

$b_i(Y_i)$ = node i's belief of being in class Y_i

Prior

All messages from neighbors

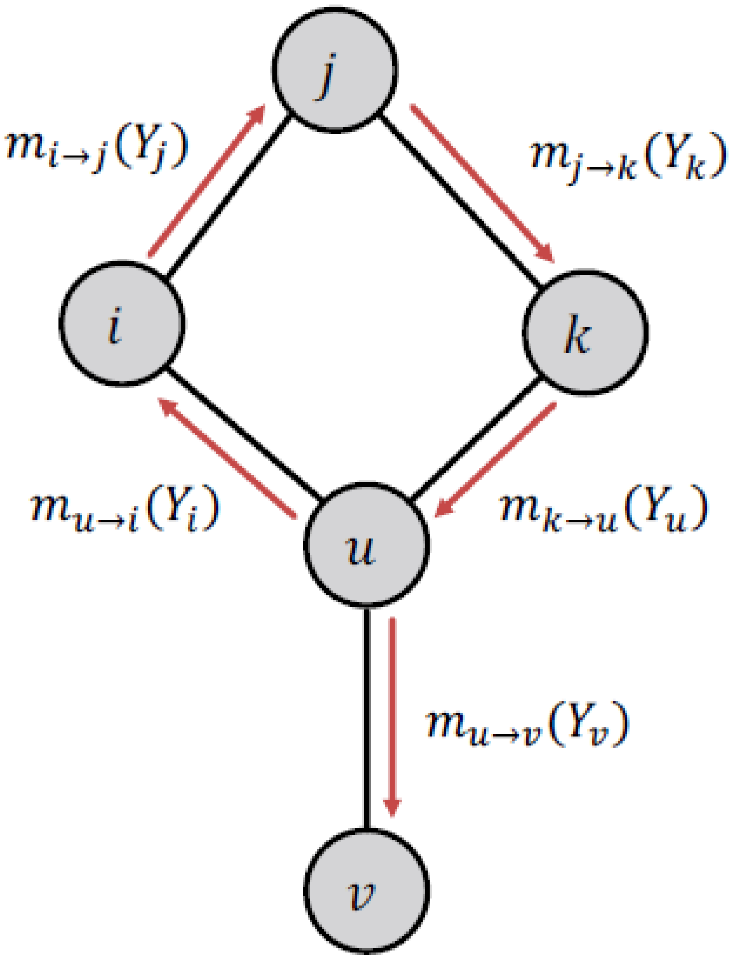
$$b_i(Y_i) = \boxed{\phi_i(Y_i)}\boxed{\Pi_{j\in N_i} m_{j\to i}(Y_i),}\ \forall\, Y_i \in \mathcal{L}$$

Loopy Belief Propagation in Graphs

- Now we consider a graph with cycles
- There is no longer an ordering of nodes
- We apply the same algorithm as in previous slides:
  - Start from arbitrary nodes
  - Follow the edges to update the neighboring nodes

What if the graph has cycles?

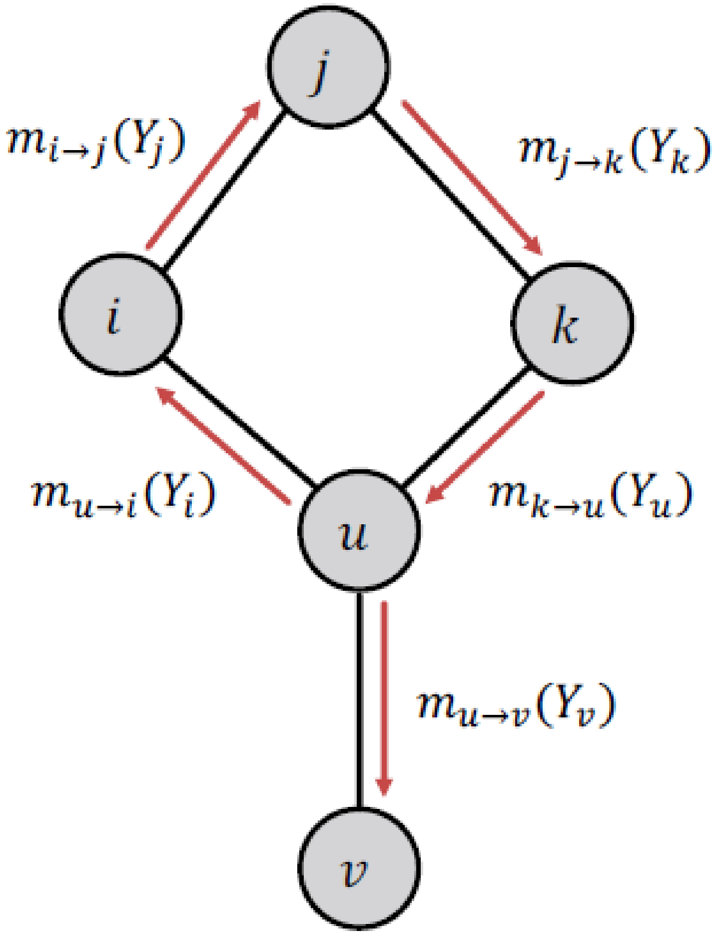
What if our graph has cycles?



Messages from different subgraphs are **no longer independent**!

But we can still run BP, but it will pass messages in loops.

What can go wrong?



- **Beliefs may not converge**
  - Message $m_{u\to i}(Y_i)$ is based on initial belief of i, not a **separate evidence** for i
  - The initial belief of i (which could be incorrect) is reinforced by the cycle $i \to j \to k \to u \to i$

- However, in practice, Loopy BP is still a good heuristic for complex graphs which contain many branches.