Efficient implementation of BP in P2P networks

Roman Schmidt

School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne (EPFL)

Evergrow Loopy Belief Propagation Algorithm and Applications Workshop
Jerusalem, Israel, February 19-21, 2007
Motivation

- Users share (correlated) data in P2P systems
  - currently mainly for retrieval
  - but correlations hold hidden knowledge
- Profit by correlations for new services
  - Distributed Knowledge Base (e.g., for software bugs)
  - Structure/cluster data (e.g., for better search results)
  - Recommendation system (e.g., for data annotation)
  - etc.
- Distributed Inference System on top of a P2P system
- Current focus and contribution
  - Message reduction
Outline

• Motivation

• Basic Concepts
  – Belief Propagation
  – The P-Grid Overlay

• P2P Belief Propagation
  – Inference Architecture
  – The Relaxation Algorithm

• Evaluation

• Conclusions
Belief Propagation

- Inference based on Bayesian networks
  - models dependencies between variables

```
OS1     Driver1
<table>
<thead>
<tr>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installed</td>
<td>0.2</td>
</tr>
</tbody>
</table>
```

```
OS1     Driver1
<table>
<thead>
<tr>
<th>Runs</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>
```

- Iterative message-passing algorithm
  - compute marginal probabilities ("beliefs")
  - provably efficient on trees, works for arbitrary networks
The BP message-passing algorithm

- Sends messages across edges
  - 2 messages per edge and iteration
  - if all messages from previous iteration were received

- Beliefs are updated per iteration
  - algorithm terminates if beliefs stabilize

- Messages are vectors
  - length corresponds to the number of node states

- Computation complexity grows exponentially with the number of states of nodes
The P-Grid Overlay

- Peers are organized in a **binary trie** structure
  - one node for every common prefix
  - trie is only virtual (exists only via routing tables)
  - all nodes remain at the leaf-level (no hierarchy)
- Multiple peers per key space partition
- Multiple routing entries (random choice)
  - per routing table level
- Logarithmic search complexity
  - even for skewed data distributions
P-Grid routing

- Keys resolved by longest prefix matching
  - Insures logarithmic search cost for skewed trees
The Distributed Inference System

- **P-Grid**
  - Bug reports, metadata, tags, etc.
  - Bayesian network
    - Variables (spread over P-Grid nodes)
    - Dependencies between variables
    - Distributive learning

- **Belief Propagation**
  - Distributed inference
    - Message-passing algorithm

- **Identified problem**
  - high message cost
Spring Relaxation

• Bayesian network as spring network
  – find minimum energy configuration (relax springs)
  – energy is proportional to the distance between P-Grid nodes
  – variables at the same node require no energy
  – optimal: all variables at one node (load balancing)

• Decentralized algorithm
  – nodes try to relax their springs
  – move correlated variables close to each other
  – optimally, at the same node (no physical message)
  – considering load distribution
Spring relations in P-Grid

Diagram showing the relationships between nodes A, B, C, D, and E, with rules such as:
- A: 1*: C, D 01*: B
  - a -> h, t f -> o, r...
- B: 1*: C, D 00*: F
  - h -> a, m m -> h, u...
- C: 0*: A, B 11*: E
- D: 0*: A, F 11*: E
- E: 0*: B, F 10*: D
  - t -> a, r u -> m...
The Relaxation Algorithm (relax variables)

currentLoad = length(localVars);
overload = currentLoad - avgLoad / 2;
IF (overload <= 0)
  return;
ENDIF
undirVars = variables having a tension only at one level;
WHILE ((overload > 0) AND (length(unidirVars) > 0))
  move variable to a peer from the level with the tension;
  removeFirst(unidirVars);
  overload = overload - 1;
ENDWHILE
...

The Relaxation Algorithm (balance load)

... 

multidirVars = vars having tensions at multiple levels;
WHILE ((currentLoad > avgLoad) AND
(length(multidirVars) > 0))
  FOR i = routingTable.levels TO 1
    IF (level i is underpopulated)
      cand = vars having a tension at level i;
      FOR j = 1 TO length(cand)
        IF (cand(j).tension(i) >= max(cand(j).tension))
          move variable to a peer from level i;
          remove(multidirVars, cand(j));
        currentLoad = currentLoad - 1;
        IF (currentLoad <= avgLoad)
          break;
    ENDIF; ENDFOR; ENDIF; ENDFOR; ENDWHILE
Algorithm execution

- Executed at each node
  - Iteratively
  - Independently (evaluated simultaneously)

- Termination
  - Max. number of iterations
  - No free or multi-directional variables to move
  - No tension reduction in last two iterations

- Effort
  - Variable movements require only 1 message
  - Trade-off to message reduction
  - Dynamic variables require “remote” updates
Evaluation

• Matlab implementation

• Diverse Bayesian networks
  – random, binary trees, scale-free
  – up to 2048 Bayesian nodes
  – up to 512 P-Grid nodes

• 10 repetitions

• 2 main evaluation criterions
  – message reduction
  – load balance
Random network

1024 nodes, average node degree 4
Binary tree network

1023 nodes

degree

variable

$10^0$  $10^1$  $10^2$  $10^3$

$10^0$  $10^1$  $10^2$  $10^3$
Scale-free network

1024 nodes, average node degree 4
Message reduction (random)

- 64 / 2048 / 4
- 128 / 2048 / 4
- 256 / 2048 / 4
- 512 / 2048 / 4
Message reduction (binary tree)

- **64 / 2047**
- **128 / 2047**
- **256 / 2047**
- **512 / 2047**
Message reduction (scale-free)

64 / 2048

128 / 2048

256 / 2048

512 / 2048
Load balancing (random)

64 / 2048 / 4

128 / 2048 / 4

256 / 2048 / 4

512 / 2048 / 4
Load balancing (binary tree)

64 / 2047

128 / 2047

256 / 2047

512 / 2047
Load balancing (scale-free)
Number of iterations

- Till relaxation algorithm termination
- Scale-free network (128 nodes, 1024 vars)
- 100 runs
Reduction effort (random)

- **64 / 2048 / 4**
- **128 / 2048 / 4**
- **256 / 2048 / 4**
- **512 / 2048 / 4**
Reduction effort (binary tree)
Reduction effort (scale-free)

- 64 / 2048
- 128 / 2048
- 128 / 2048
- 512 / 2048
Conclusions

- Decentralized relaxation algorithm
  - Reduces message cost for Belief Propagation
  - Considers load balance
- Several scenarios (Distributed Knowledge Base)
- First evaluation looks promising
- Intermediate steps are still missing
  - Learning of Bayesian network
Thank you!

Questions?