Distributed Message Passing for Large Scale Graphical Models

Yan Shen
University at Buffalo
Outline

• Algorithm
• Implementation Detail
• Experiment evaluation
• Conclusion
MAP as Convex optimization

- Maximum a posteriori (MAP) assignment

\[ x^* = \arg\max_x \prod_{i=1}^{n} \psi_i(x_i) \prod_{\alpha=1}^{m} \psi_\alpha(x_\alpha). \]  

- Reformulate the MAP problem as *convex optimization*

\[
\begin{align*}
\max & \sum_{\alpha, x_\alpha} b_\alpha(x_\alpha) \theta_\alpha(x_\alpha) + \sum_{i, x_i} b_i(x_i) \theta_i(x_i) \\
& + \epsilon \left( \sum_{\alpha} c_\alpha H(b_\alpha) + \sum_i c_i H(b_i) \right) \\
\text{subject to:} & \\
& \forall i, x_i, \alpha \in N(i), \sum_{x_\alpha \setminus x_i} b_\alpha(x_\alpha) = b_i(x_i)
\end{align*}
\]
Distributed convex belief propagation

• Breaking into sub-problems on Different Nodes

\[
\max \sum_{s \in G_P} \sum_{\alpha \in G_s, x_\alpha} b^s_\alpha(x_\alpha) \hat{\theta}_\alpha(x_\alpha) + \sum_{i \in G_s, x_i} b^s_i(x_i) \theta_i(x_i) \\
+ \epsilon \sum_{s \in G_P} \left( \sum_{\alpha \in G_s} \hat{c}_\alpha H(b^s_\alpha) + \sum_{i \in G_s} c_i H(b^s_i) \right)
\]

subject to:

\forall s, i, x_i, \alpha \in N(i), \sum_{x_\alpha \setminus x_i} b^s_\alpha(x_\alpha) = b^s_i(x_i)

\forall s, \alpha \in N_P(s), x_\alpha, b^s_\alpha(x_\alpha) = b_\alpha(x_\alpha)
Distributed convex belief propagation

Algorithm 1 (Distributed Convex Belief Propagation) Set $\hat{\psi}_\alpha(x_\alpha) = \exp(\hat{\theta}_\alpha(x_\alpha))$, $\psi_i(x_i) = \exp(\theta_i(x_i))$. Set $n_{i\to\alpha}(x_\alpha) = 1$ and set $n_{s\to\alpha}(x_\alpha) = 1$. Repeat until convergence:

1. For $s \in \mathcal{P}$ in parallel: Iterate over $i \in s$

   $\forall x_i \forall \alpha \in N(i), \quad m_{\alpha\to i}(x_i) = \left( \sum_{x_\alpha \setminus x_i} \left( \frac{\hat{\psi}_\alpha(x_\alpha)}{\prod_{j \in N(\alpha) \cap s \setminus i} n_{j\to\alpha}(x_j) \cdot n_{s\to\alpha}(x_\alpha)} \right) \right)^{1/\varepsilon \hat{c}_\alpha} \varepsilon \hat{c}_\alpha$

   $\forall \alpha \in N(i) \forall x_i, \quad n_{i\to\alpha}(x_i) \propto \left( \frac{\psi_i(x_i)}{\prod_{\beta \in N(i)} m_{\beta\to i}(x_i)} \right)^{\varepsilon \hat{c}_i} / m_{\alpha\to i}(x_i)$

2. $\forall s \in \mathcal{G}_\mathcal{P} \forall \alpha : \alpha$ is edge in $\mathcal{G}_\mathcal{P}$

   $n_{s\to\alpha}(x_\alpha) = \frac{1}{|Np(\alpha)|} \prod_{i \in N(\alpha)} n_{i\to\alpha} / \prod_{i \in s \cap N(\alpha)} n_{i\to\alpha}(x_i)$

Figure 3. Our distributed convex belief propagation algorithm extends the sequential convex belief-propagation by adding messages $n_{s\to\alpha}(x_\alpha)$ to maintain consistency between the distributed executions.
Implementation Detail

• Master and Worker Mode

![Diagram showing Master and multiple Workers]

Assign Task → Worker
Assign Task → Worker
Assign Task → Worker
Assign Task → Worker
Assign Task → Worker

Transmit Back Result → Master

• Round Communication

![Diagram showing Master and Worker communication]

Send Problem to Client → Worker
Send Local Belief to Master → Master
Send Lagrange Multiplier to Worker → Worker

......

Final Result Request → Worker
Final Result → Master
Implementation Detail

- Distributed Loading Potential Data
  - Graph Structure and Node Assignment in Master
  - Worker load their subgraph and local potential from file in parallel

\[ \hat{\psi}_\alpha(x_\alpha) \]
Implementation Detail

- Hybrid use of MPI and OpenMP
  - Worker updates their Local Belief in (Implemented in OpenMP)
  - Worker and Master run on their Local Data and use MPI to communicate.

Worker Node

```plaintext
#pragma omp parallel on i in s
```

\[
\forall x_i \forall \alpha \in N(i), \quad m_{\alpha-i}(x_i) = \left( \sum_{x_\alpha \setminus x_i} \left( \prod_{j \in N(\alpha) \cap \{s\}} n_{j-\alpha}(x_j) \cdot n_{s-\alpha}(x_\alpha) \right)^{1/\epsilon_\alpha} \right)^{\epsilon_\alpha}
\]

\[
\forall \alpha \in N(i) \forall x_i, \quad n_{i-\alpha}(x_i) \propto \left( \psi_i(x_i) \prod_{\beta \in N(i)} m_{\beta-i}(x_i) \right)^{\epsilon_\alpha/\epsilon_i} / m_{\alpha-i}(x_i)
\]

Master Node

```plaintext
#pragma omp parallel on s
```

\[
n_{s-\alpha}(x_\alpha) = \frac{1}{|N_P(\alpha)|} \prod_{i \in N(\alpha)} n_{i-\alpha} / \prod_{i \in \alpha \setminus N(\alpha)} n_{i-\alpha}(x_i)
\]
Implementation Detail

• Serial and Parallel Mode on Master Updates
  
  • Serial Mode

  • Parallel Mode
## Experiment evaluation

- Result and Speedup of Parallelism

<table>
<thead>
<tr>
<th>Method</th>
<th>Run Time</th>
<th>duality gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>cen. cBP</td>
<td>70</td>
<td>0.043</td>
</tr>
<tr>
<td>dis. cBP(S)</td>
<td>25</td>
<td>0.0449</td>
</tr>
<tr>
<td>dis. cBP(P)</td>
<td>20</td>
<td>0.045</td>
</tr>
</tbody>
</table>

(a) Tsukuba

(d) cBP $\epsilon = 0$
(e) cBP $\epsilon = 0.01$
(f) cBP $\epsilon = 0.1$
Experiment evaluation

• Convergence Result

<table>
<thead>
<tr>
<th>Sever Iteration Time</th>
<th>Primal Value</th>
<th>Dual Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>28672</td>
<td>38672</td>
</tr>
<tr>
<td>2</td>
<td>32021</td>
<td>35621</td>
</tr>
<tr>
<td>3</td>
<td>34781</td>
<td>35583</td>
</tr>
<tr>
<td>4</td>
<td>35473</td>
<td>35581</td>
</tr>
<tr>
<td>5</td>
<td>35498</td>
<td>35581</td>
</tr>
<tr>
<td>6</td>
<td>35543</td>
<td>35580</td>
</tr>
<tr>
<td>7</td>
<td>35570</td>
<td>35579</td>
</tr>
<tr>
<td>8</td>
<td>35573</td>
<td>35578</td>
</tr>
<tr>
<td>9</td>
<td>35575</td>
<td>35578</td>
</tr>
</tbody>
</table>

Convergence Result as number of master iteration
(Master iteration times= 10 , Worker iteration times= 10)
Experiment evaluation

- Larger Data on Bigger clusters
  - 1GB data of Local Belief
  - Partition Local Belief Data in 16 nodes in Grid
  - 17 Total Nodes (16 workers + 1 master)

```bash
#!/bin/bash
#SBATCH --nodes=17
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=3
#SBATCH --time=05:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=yshen22@buffalo.edu
#SBATCH --output=slurmQ.out
#SBATCH --job-name=mpi-testmodule
load intel-mpi intel-mpi
export I_MPI_PMI_LIBRARY=/usr/lib64/libpmi.so
srun ./dcBP -f tsukuba.cbp -c 10 -s 20
```

Runtime Report

- Job ID: 8804896
- Cluster: ub-hpc
- User/Group: yshen22/cse633s18
- State: COMPLETED (exit code 0)
- Nodes: 17
- Cores per node: 3
- CPU Utilized: 1-11:45:21
- CPU Efficiency: 60.42% of 2-11:10:27 core-walltime
- Memory Utilized: 4.76 GB (estimated maximum)
- Memory Efficiency: 3.42% of 139.45 GB (2.73 GB/core)
Experiment evaluation

- Convergence Result on Clusters

- Running Times
  3329.66s for total 200 iterations
Conclusion

• Large scale graphical models by dividing the computation and memory requirements into multiple machines.

• Convergence and optimality guarantees are preserved.

• Main benefit: the use of multiple computers.