Adaptive Belief Propagation

ICML 2015

Presenter: Chuhang Zou (czou4)

Motivation

- Consider inference in large-scale models
- Large-scale systems may monitor the health status of many patients; however, different physicians limit their interest to patients under their immediate care.
- Only a subset of latent variables are of interests, may vary from instance to instance for different application

Image from:
http://medicaleconomics.modernmedicine.com/medical-economics/news/user-defined-tags/physician-shortage/percentage-physician-assistants-who-work?page=full

Adaptive Inference

- **Refers to the task of recomputing the results of an inference problem which is only slightly different from one which has already been solved.**

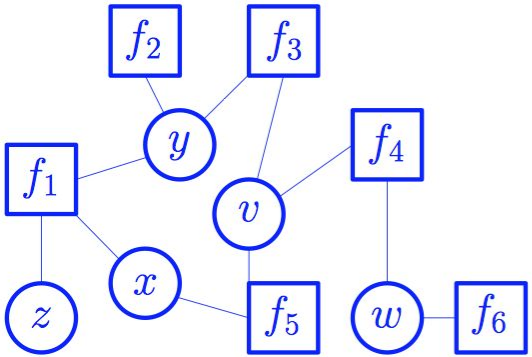
- Stems from incremental computing: the amount of work we expect to do to solve a problem is (hopefully) proportional to the amount of change in the problem from our already-solved version.

Related work

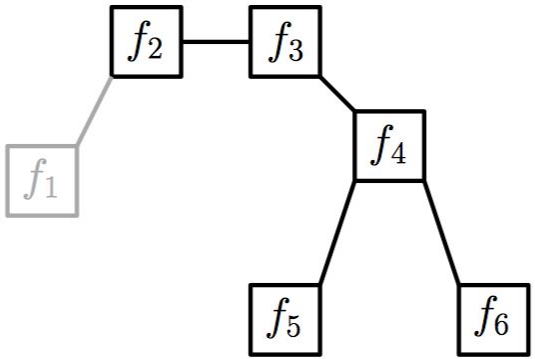
- Factor elimination in factor graph (Su ̈mer et al., 2011)
- Utilizing the factor elimination algorithm to evaluate node marginals

Factor elimination. Factor elimination takes a factor graph G_1 and an elimination tree T_1 as input and sequentially eliminates the leaf factors in the elimination tree. As an example, to eliminate f_1 in iteration t, we first marginalize out any variables that are only adjacent to the eliminated factor, and then propagate this information to the unique neighbor in T_t, that is, $f_2' = f_2 \sum_z f_1$.

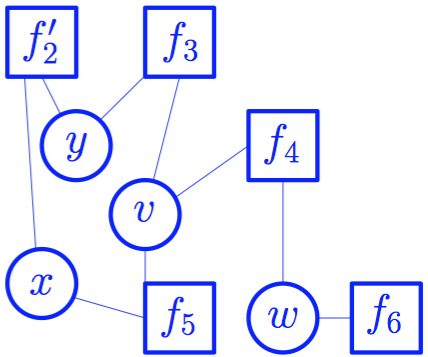
$$f_2' = f_2 \sum_z f_1$$



(a) G_t



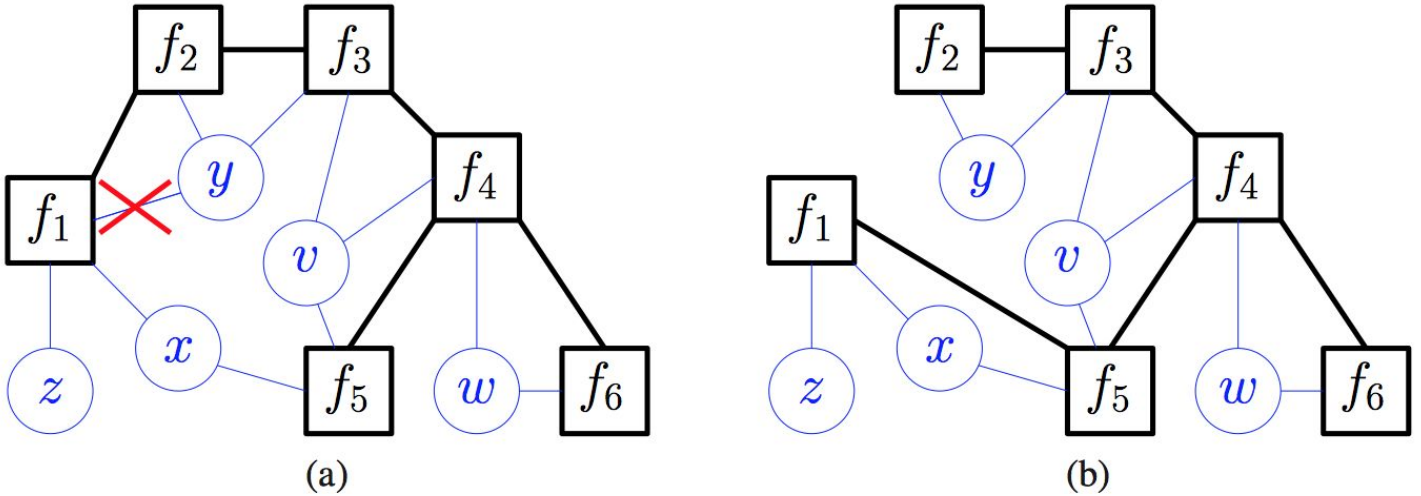
(b) T_t



(c) G_{t+1}

Related work

- Update = modifying the elimination tree



Related work

- A balanced representation of elimination tree: $\mathcal{O}(|\mathcal{X}|^{3t_w} N)$
- Computation of a node marginal: $\mathcal{O}(|\mathcal{X}|^{2t_w} \log N)$
- N Is number of nodes
- t_w is the elimination tree width
- $|\mathcal{X}|$ Is the alphabet size

- The preprocessing step becomes prohibitive as $|\mathcal{X}|$ and t_w grow large, thus making this method inappropriate for dense loopy graphs.
- For trees? Standard BP: $\mathcal{O}(|\mathcal{X}|N)$

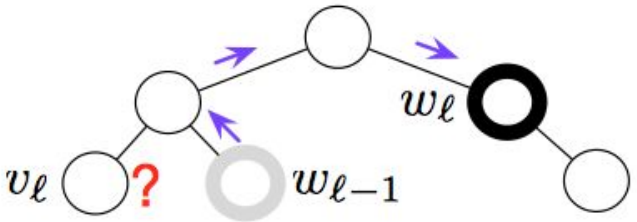
Related work

- Core problem for RCTree-BP: the computational complexity is impacted significantly by not taking into account the relative distances between consecutive nodes of interest.
- Other method:

(Wick & McCallum, 2011) consider the focused inference problem, propose a prioritized scheme of Metropolis Hastings algorithm that focuses on sampling variables from a set of interest.

(Chechetka & Guestrin, 2010) create a prioritized message schedule weighted towards messages to which the set of interest is most sensitive.

Limitation: limited to discrete graph
the set of interest is fixed and
the preprocessing time depends on #edges and #neighbors of the nodes.

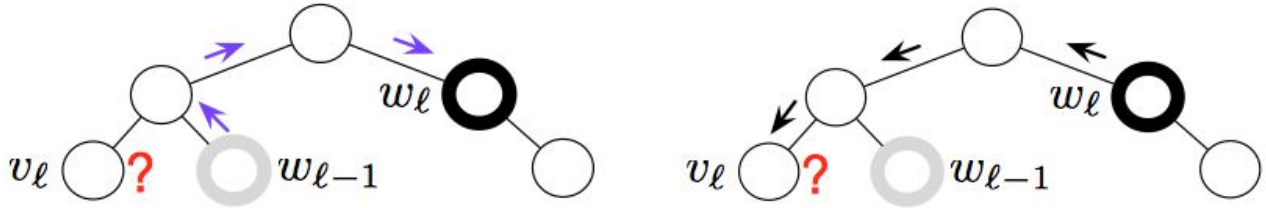


Main Idea

- BP + adaptive inference
- Goal: avoids redundant computations and whose average-case performance shows significantly lower complexity compared to BP.
- How?

Main Idea

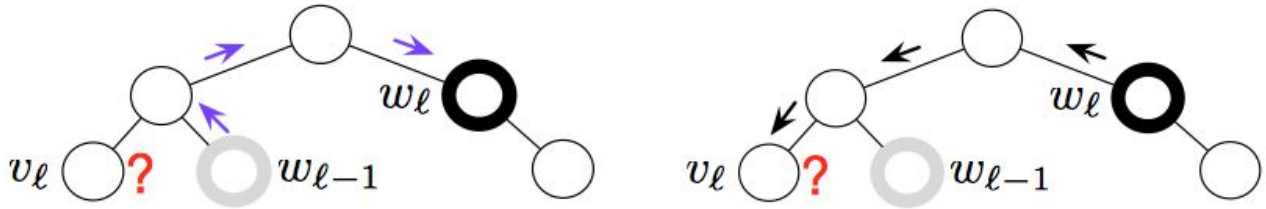
- BP + adaptive inference
- Goal: avoids redundant computations and whose average-case performance shows significantly lower complexity compared to BP.
- How?
- Send only messages between the node where a measurement has been obtained and the node whose marginal is of interest



(a) Messages from $w_{\ell-1}$ to w_ℓ (b) Messages from w_ℓ to v_ℓ

Main Idea

- BP + adaptive inference
- Goal: avoids redundant computations and whose average-case performance shows significantly lower complexity compared to BP.
- How?
- Send only messages between the node where a measurement has been obtained and the node whose marginal is of interest



(a) Messages from $w_{\ell-1}$ to w_ℓ (b) Messages from w_ℓ to v_ℓ

- Further, extend to loopy Gaussian models

Problem Statement

- Consider discrete MRFs
- Update: a measurement is added at a time and only one or a few marginals are of interest at any point.
- Notation:

MRF: $G = (V, \mathcal{E})$ with N latent variables $\{X_1, \ldots, X_N\}$

Each latent variable is linked to m_k observed set $\{Y_{k,1:m_k}\}$

Measurement plan $\{w_1, \ldots, w_M\} = \{w_{1:M}\}$ from each set

Sequence of latent node $\boldsymbol{v} = \{v_{1:M}\}$, whose marginal is of interest at each step

Problem Statement

- Standard BP update: $m_{i \to j}(x_j) = \sum_{x_i} \varphi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in \mathcal{N}(i) \setminus j} m_{k \to i}(x_i)$

- If a measurement is added at iteration ℓ linking to node x_{w_ℓ}, observed nodes are absorbed into the node potential of the latent node they link to:

$$\varphi^{(\ell)}_{w_\ell}(x_{w_\ell}) = \varphi^{(\ell-1)}_{w_\ell}(x_{w_\ell}) \chi_{w_\ell u}(x_{w_\ell}, y_u) \qquad \mathcal{O}(|\mathcal{X}|)$$

Node potential before incorporating the measurement

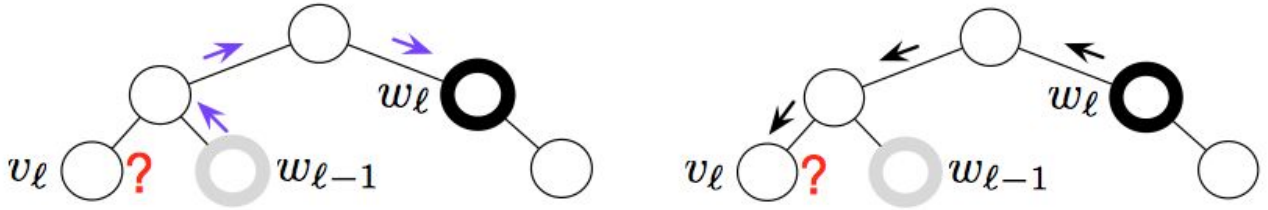
Measurement (observed node)

Lowest Common Ancestor

$$\boldsymbol{w} = \{w_1, \ldots, w_M\}$$

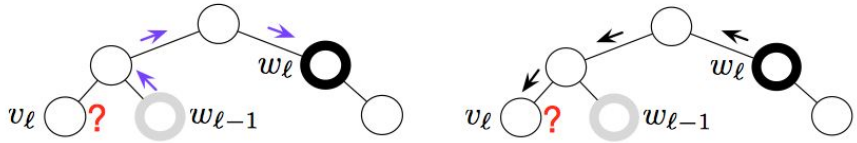
- Consider tree case, obtain one measurement at one step and are interested in finding the marginal at a given node $\boldsymbol{v} = \{v_1, \ldots, v_M\}$
- Path: lowest common ancestor -- Range Minimum Query (RMQ) problem

(Harel & Tarjan, 1984)



(a) Messages from $w_{\ell-1}$ to w_ℓ (b) Messages from w_ℓ to v_ℓ

Adaptive BP



(a) Messages from $w_{\ell-1}$ to w_ℓ (b) Messages from w_ℓ to v_ℓ

- Keypoint: after the incorporation of a measurement at node χ_{w_ℓ}, the evaluation of the messages along the unique path from node w_ℓ to node v_ℓ is sufficient for the determination of node v_ℓ 's marginal.
- Initialization: all node potentials we propagate messages along the entire graph in both directions
  1) the messages from w1 to v1 are computed.
- 2) propagate messages from w1 to w2, update the node potential of Xw2 and send messages from w2 to v2.
- 3) Continue above procedure

Messages are updated as

$$m_{i\to j}(x_j) = \sum_{x_i} \varphi_i(x_i)\psi_{ij}(x_i, x_j) \prod_{k\in\mathcal{N}(i)\setminus j} m_{k\to i}(x_i),$$

$$\forall (i,j) \in \mathcal{M}(w_{\ell-1} \to w_\ell) \text{ and } \mathcal{M}(w_\ell \to v_\ell),$$

Adaptive BP

If the latent graph is a chain, there is no need to find the lca: we simply propagate and update

Algorithm 1 ADAPTIVE BP

Preprocessing

Determine Euler tour $\boldsymbol{E}$, depths of elements in the Euler tour $\boldsymbol{D}_e$, vector $\boldsymbol{H}$ which stores the index of the first occurrence of node i in $\boldsymbol{E}$, and matrix $\boldsymbol{M}$ which stores the index of the minimum value of the subarray of $\boldsymbol{D}_e$ starting at i and having length 2^{j-1}.

Initialization

Initialize the node, pairwise potentials and messages.

Iteration

for $\ell = 1, \ldots, M$ **do**

Find $\text{lca}(w_{\ell-1}, w_\ell)$ from Eq. (5).

Determine the schedule $\mathcal{M}(w_{\ell-1} \to w_\ell)$.

Compute messages $m_{i \to j}(x_j)$ in $\mathcal{M}(w_{\ell-1} \to w_\ell)$.

Update the node potential at X_{w_ℓ}.

Find $\text{lca}(w_\ell, v_\ell)$ from Eq. (5).

Determine the schedule $\mathcal{M}(w_\ell \to v_\ell)$.

Compute messages $m_{i \to j}(x_j)$ in $\mathcal{M}(w_\ell \to v_\ell)$.

Compute the marginal of interest $p_{\mathsf{X}_{v_\ell}}(x_{v_\ell})$.

end for

Adaptive BP

Complexity of preprocessing:
Depth: $\mathcal{O}(N)$
Euler tour: $\mathcal{O}(N)$
M matrix: $\mathcal{O}(N \log_2 N)$
Overall : $\mathcal{O}(N \log_2 N)$

Algorithm 1 ADAPTIVE BP

Preprocessing

Determine Euler tour $\boldsymbol{E}$, depths of elements in the Euler tour $\boldsymbol{D}_e$, vector $\boldsymbol{H}$ which stores the index of the first occurrence of node i in $\boldsymbol{E}$, and matrix $\boldsymbol{M}$ which stores the index of the minimum value of the subarray of $\boldsymbol{D}_e$ starting at i and having length 2^{j-1}.

Initialization

Initialize the node, pairwise potentials and messages.

Iteration

for $\ell = 1, \ldots, M$ **do**
  Find $\text{lca}(w_{\ell-1}, w_\ell)$ from Eq. (5).
  Determine the schedule $\mathcal{M}(w_{\ell-1} \to w_\ell)$.
  Compute messages $m_{i \to j}(x_j)$ in $\mathcal{M}(w_{\ell-1} \to w_\ell)$.
  Update the node potential at X_{w_ℓ}.
  Find $\text{lca}(w_\ell, v_\ell)$ from Eq. (5).
  Determine the schedule $\mathcal{M}(w_\ell \to v_\ell)$.
  Compute messages $m_{i \to j}(x_j)$ in $\mathcal{M}(w_\ell \to v_\ell)$.
  Compute the marginal of interest $p_{\mathsf{X}_{v_\ell}}(x_{v_\ell})$.
end for

Adaptive BP

Complexity of preprocessing:
Overall : $\mathcal{O}(N \log_2 N)$

Adaptive BP:

$\mathcal{O}(\sum_{\ell=1}^{M}(\text{dist}(w_{\ell-1}, w_\ell) + \text{dist}(w_\ell, v_\ell))|\mathcal{X}|^2)$

Standard BP: $\mathcal{O}(mN^2|\mathcal{X}|^2)$

The complexity of adaptive BP directly depends on the context of the measurement and marginal order, while standard BP has a fixed cost per iteration.

Algorithm 1 ADAPTIVE BP

Preprocessing

Determine Euler tour $\boldsymbol{E}$, depths of elements in the Euler tour $\boldsymbol{D}_e$, vector $\boldsymbol{H}$ which stores the index of the first occurrence of node i in $\boldsymbol{E}$, and matrix $\boldsymbol{M}$ which stores the index of the minimum value of the subarray of $\boldsymbol{D}_e$ starting at i and having length 2^{j-1}.

Initialization

Initialize the node, pairwise potentials and messages.

Iteration

for $\ell = 1, \ldots, M$ **do**
- Find $\text{lca}(w_{\ell-1}, w_\ell)$ from Eq. (5).
- Determine the schedule $\mathcal{M}(w_{\ell-1} \rightarrow w_\ell)$.
- Compute messages $m_{i\rightarrow j}(x_j)$ in $\mathcal{M}(w_{\ell-1} \rightarrow w_\ell)$.
- Update the node potential at X_{w_ℓ}.
- Find $\text{lca}(w_\ell, v_\ell)$ from Eq. (5).
- Determine the schedule $\mathcal{M}(w_\ell \rightarrow v_\ell)$.
- Compute messages $m_{i\rightarrow j}(x_j)$ in $\mathcal{M}(w_\ell \rightarrow v_\ell)$.
- Compute the marginal of interest $p_{\mathsf{X}_{v_\ell}}(x_{v_\ell})$.

end for

Adaptive BP

Adaptive BP:

$\mathcal{O}(\sum_{\ell=1}^{M}(\text{dist}(w_{\ell-1}, w_\ell) + \text{dist}(w_\ell, v_\ell))|\mathcal{X}|^2)$

Standard BP: $\mathcal{O}(mN^2|\mathcal{X}|^2)$

- Complexity analysis:

Worst case: tree is highly unbalanced (diameter N), complexity same as standard BP; Balanced tree with each node has q childrens, $\mathcal{O}(\lfloor \log_q N \rfloor)$ messages will be sent for each iteration

Best case: w_ℓ, v_ℓ are akin to each other, over all $\mathcal{O}(mN|\mathcal{X}|^2)$

Extension to Max-Product

- Task: find MAP sequence
- Keep track of **Delta message**: indicates the value of the source node that corresponds to the MAP sequence of the subtree rooted at the source node (excl. the branch containing the target node) for a specific value of the target node.

- Strategy same as before: propagate delta messages from w1 to w2 and then backtrack from w2 down to the leaves...

Extension to Gaussian Loopy MRFs

Apply the Feedback Message Passing (FMP) algorithm by (Liu et al., 2012), which provides a way to evaluate the exact means and variances in loopy Gaussian MRFs.

Then update is trivial.

Algorithm 2 FEEDBACK MESSAGE PASSING (FMP)

1. Construct K potential vectors: $h^p = J_{\mathcal{T}p}, \forall p \in \mathcal{F}$.
2. Run BP $K+1$ times on $\mathcal{T}$ with parameters $\{h_{\mathcal{T}}, J_{\mathcal{T}}\}$, $\{h^p, J_{\mathcal{T}}\}_{p\in\mathcal{F}}$, which will produce messages $h^{\mathcal{T}}_{i\to j}, h^p_{i\to j}, J^{\mathcal{T}}_{i\to j}$ and marginals $\hat{\mu}^{\mathcal{T}}_i, g^p_i, \hat{\Sigma}^{\mathcal{T}}_{ii}$.
3. Obtain the K–sized graph with updated parameters $\hat{h}_{\mathcal{F}}, \hat{J}_{\mathcal{F}}$ as

$$[\hat{J}_{\mathcal{F}}]_{pq} = J_{pq} - \sum_{i\in\mathcal{N}(p)\cap\mathcal{T}} J_{pi} g^q_i, \ \forall p,q \in \mathcal{F} \quad (6)$$

$$[\hat{h}_{\mathcal{F}}]_p = h_p \ - \sum_{i\in\mathcal{N}(p)\cap\mathcal{T}} J_{pi} \hat{\mu}^{\mathcal{T}}_i, \ \forall p \in \mathcal{F} \quad (7)$$

   and solve for $\Sigma_{\mathcal{F}} = \hat{J}^{-1}_{\mathcal{F}}$ and $\mu_{\mathcal{F}} = \Sigma_{\mathcal{F}}\hat{h}_{\mathcal{F}}$.
4. Revise the potential vector in $\mathcal{T}$ as $\tilde{h}_i = h_i - \sum_{j\in\mathcal{N}(i)\cap\mathcal{F}} J_{ij}[\mu_{\mathcal{F}}]_j, \ \forall i \in \mathcal{T}$ and obtain the exact means by running BP one more time on the revised potential vector (the corresponding messages will be denoted by $\tilde{h}^{\mathcal{T}}_{i\to j}$).
5. Correct the variances with

$$\Sigma_{ii} = \hat{\Sigma}^{\mathcal{T}}_{ii} + \sum_{p\in\mathcal{F}}\sum_{q\in\mathcal{F}} g^p_i [\Sigma_{\mathcal{F}}]_{pq} g^q_i, \ \forall i \in \mathcal{T}. \quad (8)$$

Experiments - synthetic dataset

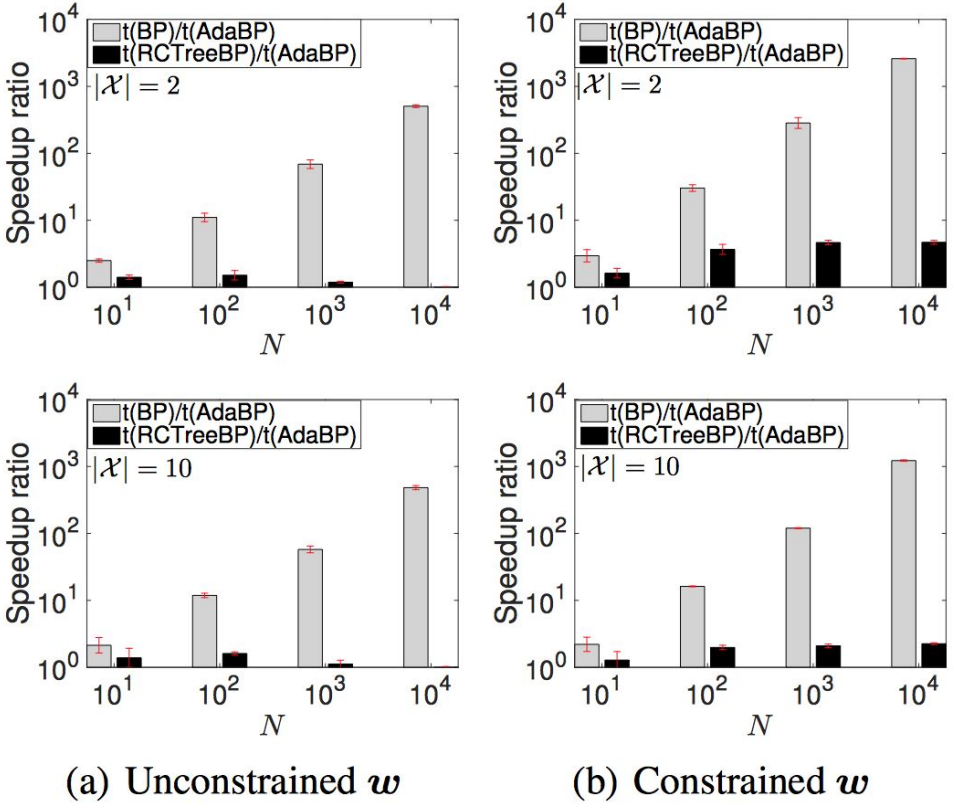
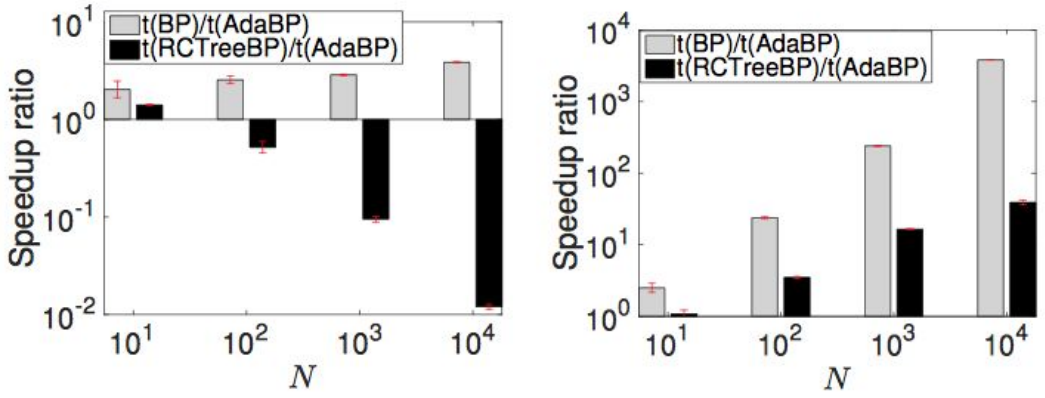


Figure 2. Comparison of the total running times of AdaBP against standard BP (gray) and RCTreeBP (black) over different alphabet sizes, $|\mathcal{X}| \in \{2, 10\}$. **(a)** Distance between consecutive elements $\mathbb{E}[\text{dist}(w_{\ell-1}, w_\ell)]$ is unconstrained. **(b)** $\mathbb{E}[\text{dist}(w_{\ell-1}, w_\ell)] \leq |\mathcal{X}| \log N$. For average distance $\mathbb{E}[\text{dist}(w_{\ell-1}, w_\ell)]$ smaller than $|\mathcal{X}| \log N$, AdaBP is 1.3–4.7 faster than RCTreeBP.

Experiments - synthetic dataset



(a) $\mathbb{E}\{\text{dist}(w_{\ell-1}, w_{\ell})\} \sim \mathcal{O}(N)$ (b) $\mathbb{E}\{\text{dist}(w_{\ell-1}, w_{\ell})\} = 2$

Figure 3. **(a)** Worst case. Distance between consecutive elements is on the order of N. AdaBP is comparable to standard BP (still being 2–4 times faster) and orders of magnitude slower than RC-TreeBP. **(b)** Best case. Consecutive elements are very close to each other. Only a constant number of updates is required per step for AdaBP.

Experiment - real data

- AdaMP (max-product) to biological data.
- Explore the effects of point-wise mutations in DNA sequences to the birth or disappear- ance of CpG islands.

$$v_\ell = w_\ell, \forall \ell.$$

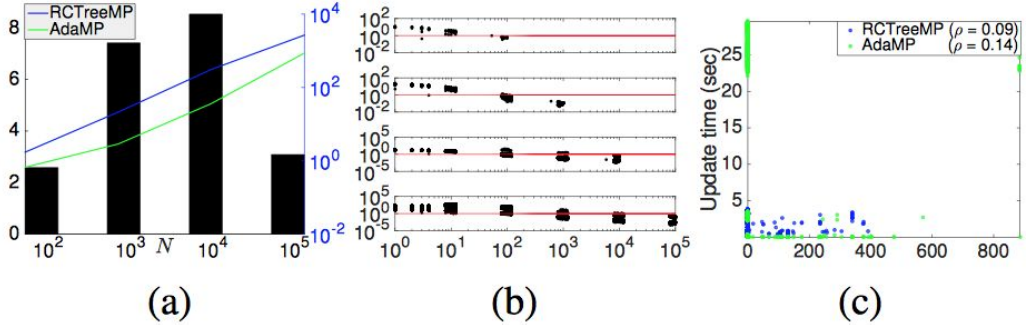


Figure 4. **(a)** Speedups of AdaMP over RCTreeMP for varying-size stretches of chr 21 (10^2–10^5 bp). **(a)** Left y-axis shows the speedup over RCTreeMP, while right y-axis the actual running times in sec (represented as lines). **(b)** Ratios of update times of AdaMP over RCTreeMP for different values of $\text{dist}(w_{\ell-1}, w_\ell)$ (x-axis: $\text{dist}(w_{\ell-1}, w_\ell)$, y-axis: speedup). The four log-log plots correspond to four different DNA stretches of $10^2, 10^3, 10^4, 10^5$ bp size, respectively. For smaller distances, AdaMP outperforms, but for distances closer to the graph size N, RCTreeMP is preferable. Red line indicates ratio of 1. **(c)** Both methods are not very sensitive to changes in the MAP sequence between consecutive iterations (x-axis: # of bp that differ between consecutive MAP sequences).

Experiment - real data

- Temperature measurements collected from 53 wireless sensors at 30 sec intervals from the Intel Berkeley Research lab.
- latent temperatures in the various locations of the lab as a grid graph

$$\bar{X}_t = AX_{t-1} + V_{t-1}$$

$$V_{t-1} \sim \mathcal{N}(0, Q)$$

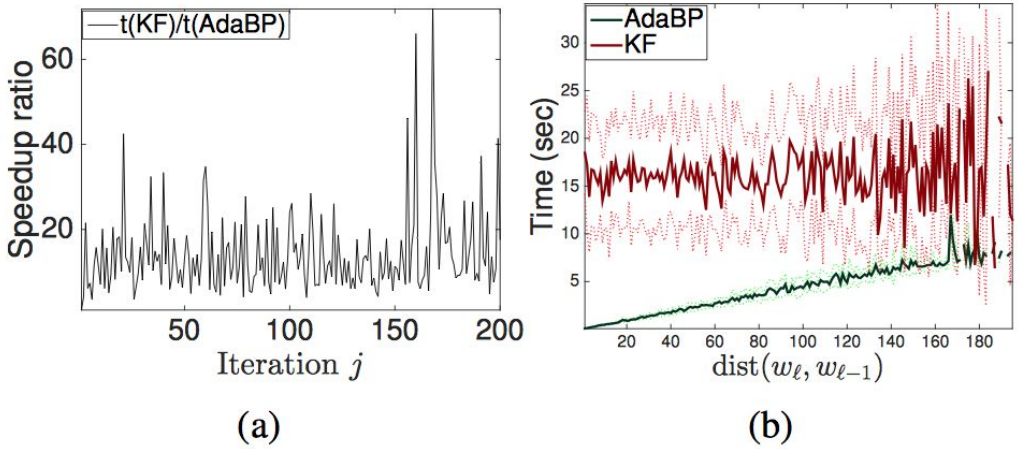


Figure 5. **(a)** This figure shows the speedup over Kalman filter (KF). AdaBP is 1–42 times faster than standard kalman filtering/smoothing techniques. **(b)** Running time per iteration of AdaBP and KF as a function of consecutive distance between elements. AdaBP is much more sensitive to dist$(w_{\ell-1}, w_\ell)$ and as the figure suggests it is much faster than KF when dist$(w_\ell, w_{\ell-1})$ is small. Dotted plots represents deviation due to different runs of the experiment.

Conclusion

- Present AdaBP, suited to sequential inference problems, when there is little or no knowledge of the measurement schedule in advance.
- Gives exact results for trees in discrete and Gaussian Markov Random Fields (MRFs)
- Shows extensions to Gaussian loopy graphs

- In synthetic and real experiments it outperforms standard BP by orders of magnitude and explore the settings that it is advantageous over RCTreeBP

Thank You