Theorem 4.2. Let \((G,C,S)\) be an arbitrary junction tree, and \(\textbf{b}\) and \(\delta^\ast\) defined in Theorem 4.1. Then \(\delta^\ast\) is a locally person-by-person optimal strategy: \(\text{EU}(\{\delta^\ast_i, \delta^\ast_D|\}) \leq \text{EU}(\delta^\ast)\) for any \(i \in D\) and \(\delta_D|\).

Additively Decomposable Utilities. Our algorithms rely on the factorization structure of the augmented distribution \(q(x)\). For this reason, multiplicative utilities fit naturally, but additive utilities are more difficult (as they also are in exact inference) [Koller and Friedman, 2009]. To create factorization structure in additive utility problems, we augment the model with a latent “selector” variable, similar to that in mixture models. For details, see the appendix.

4.2 Proximal Algorithms

In this section, we present a proximal point approach [e.g., Martinet, 1970, Rockafellar, 1976] for the MEU problems. Similar methods have been applied to standard inference problems, e.g., Ravikumar et al. [2010].

We start with a brief introduction to the proximal point algorithm. Consider an optimization problem \(\min_{\tau \in \mathcal{M}} f(\tau)\). A proximal method instead iteratively solves a sequence of “proximal” problems \(\tau^{t+1} = \arg\min_{\tau \in \mathcal{M}} \{f(\tau) + w^t D(\tau||\tau^t)\}\), (20)

where \(\tau^t\) is the solution at iteration \(t\) and \(w^t\) is a positive coefficient. \(D(\cdot||\cdot)\) is a distance, called the proximal function; typical choices are Euclidean or Bregman distances or \(\psi\)-divergences [e.g., Teboulle, 1992, Iusem and Teboulle, 1993]. Convergence of proximal algorithms has been well studied: the objective series \(\{f(\tau^t)\}\) is guaranteed to be non-increasing at each iteration, and \(\{\tau^t\}\) converges to an optimal solution (sometimes superlinearly) for convex programs, under some regularity conditions on the coefficients \(\{w^t\}\). See, e.g., Rockafellar [1976], Tseng and Bertsekas [1993], Iusem and Teboulle [1993].

Here, we use an entropic proximal function that naturally fits the MEU problem:

\[
D(\tau||\tau^t) = \sum_{i \in D} \sum_{x} \tau(x) \log \frac{\tau_i(x|x_{\text{pa}(i)})}{\tau_i^t(x|x_{\text{pa}(i)})},
\]

a sum of conditional KL-divergences. The proximal update for the MEU dual (7) then reduces to

\[
\tau^{t+1} = \arg\max_{\tau \in \mathcal{M}} \{\theta^t, \tau\} + H(x) - (1 - w^t) H(x_i|x_{\text{pa}(i)})\}
\]

where \(\theta^t(x) = \theta(x) + w^t \sum_{i \in D} \log \tau_i^t(x_i|x_{\text{pa}(i)})\). This has the same form as the annealed problem (10) and can be solved by the message passing scheme (12)-(13). Unlike annealing, the proximal algorithm updates \(\theta^t\) each iteration and does not need \(w^t\) to approach zero.

We use two choices of coefficients \(\{w^t\}\): (1) \(w^t = 1\) (constant), and (2) \(w^t = 1/t\) (harmonic). The choice \(w^t = 1\) is especially interesting because the proximal update reduces to a standard marginalization problem, solvable by standard tools without the MEU’s temporal elimination order restrictions. Concretely, the proximal update in this case reduces to

\[
\tau^{t+1}_i(x_i|x_{\text{pa}(i)}) \propto \tau^t_i(x_i|x_{\text{pa}(i)}) \text{EU}(x_i|x_{\text{fam}(i)} : \delta^\ast_{i})
\]

with \(\text{EU}(x_i|x_{\text{fam}(i)} : \delta^\ast_{i})\) as defined in (6). This proximal update can be seen as a “soft” and “parallel” version of the greedy update (6), which makes a hard update at a single decision node, instead of a soft modification simultaneously for all decision nodes. The soft update makes it possible to correct earlier suboptimal choices and allows decision nodes to make cooperative movements. However, convergence with \(w^t = 1\) may be slow; using \(w^t = 1/t\) takes larger steps but is no longer a standard marginalization.

5 Experiments

We demonstrate our algorithms on several influence diagrams, including randomly generated IDs, large scale IDs constructed from problems in the UAI08 inference challenge, and finally practically motivated IDs for decentralized detection in wireless sensor networks. We find that our algorithms typically find better solutions than SPU with comparable time complexity; for large scale problems with many decision nodes, our algorithms are more computationally efficient than SPU because one step of SPU requires updating (6) (a global expectation) for all the decision nodes.

In all experiments, we test single policy updating (SPU), our MEU-BP running directly at zero temperature (BP-0\(^+\)), annealed BP with temperature \(\epsilon^t = 1/t\) (Anneal-BP-1/t), and the proximal versions with \(w^t = 1\) (Prox-BP-one) and \(w^t = 1/t\) (Prox-BP-1/t). For the BP-based algorithms, we use two constructions of junction graphs: a standard junction tree by triangulating the DAG in backwards topological order, and a loopy junction graph following [Mateescu et al., 2010] that corresponds to Pearl’s loopy BP; for SPU, we use the same junction graphs to calculate the inner update (6). The junction trees ensure the inner updates of SPU and Prox-BP-one are performed exactly, and has optimality guarantees in Theorem 4.1, but may be computationally more expensive than the loopy junction graphs. For the proximal versions, we set a maximum of 5 iterations in the inner loop; changing this value did not seem to lead to significantly different results. The BP-based algorithms may return non-deterministic strategies; we round to deterministic strategies by taking the largest values.