Optimizing MPF Queries: Decision Support and Probabilistic Inference

This paper proposes using relational MPF queries to support probabilistic inference and decision making. This proposal is based on the observation that a general query on a factorized probabilistic model could be simplified by pushing the summations into the products. This corresponds to pushing aggregation operators into a join sequence in the context of the relational model. Hence, using relational MPF queries to express probabilistic inference tasks, we're able to leverage and extend the database optimization techniques to support probabilistic inference and achieve scalability.

Merits

S1 The connection between MPF query processing and probabilistic inference observed in this paper is profound. Probabilistic inference is an important aspect of data analytics and statistics; leveraging database techniques allows us to achieve scalability and perform complex tasks.

S2 This paper provides readers with great technical explanations of query optimizing processing (CS) and how they can be used to optimize probabilistic queries (VE). The crucial observation is that aggregations and joins from relation algebra map to marginalizations and products in probabilistic inference. The authors developed a set of algorithms based on this observation and gave us both theoretical and experimental results showing their benefits under different circumstances. This is even worth further discussion on databases' native support for data analytics tasks.

S3 The authors are insightful enough to use their model to support decision making. In fact, since the model is general, they may support even more tasks than discussed, e.g, image segmentation, medical diagnosis, etc.

Concerns

W1 Though the authors mentioned "such as Variable Elimination (VE) and Belief Propagation (BP) from the probabilistic inference literature" in the abstract, only variable elimination is discussed in the paper. However, BP is often more efficient (e.g. when the junction tree width is large) and thus more widely adopted. Since it is not apparent how BP fits into the relational model, it would be better if the authors discuss whether this poses a limitation on their approach, e.g. how difficult it is to combine BP and the relational model.

W2 When comparing the proposed algorithms (CS+, VE, VE+), the authors tried to validate the VE+ algorithm in terms of quality (plan space) and performance (optimization cost), but both aspects are not strong enough:

- They tried some theoretical analysis, but concluded with a gap between the plan spaces of VE+ and CS+. Though they made up for this point in their experiments, we don't have a solid guarantee of how often VE+ is optimal and how large this theoretical gap is.
- In Section 4.5 Theorem 3, they compared they constructed a star example and analyzed the *worst-case* optimization cost of VE and CS+. This is not sufficient since they also need to show the average case and include VE+.

W3 The experiments supporting VE+ are not strong. In Section 5.2, the authors showed that VE+ only *sometimes* yielded the optimal plan but did not mention how often it is the case, and in Section 5.4, the planning time did not differ so much (around 0.1 seconds). Moreover, they did not mention whether VE+ actually output the optimal plan when it ran faster. If it did not, the actual query time might even be more significant than the saved plan time.