Node Splitting: A Scheme for Generating Upper Bounds in Bayesian Networks

Arthur Choi, Mark Chavira and Adnan Darwiche

Purpose

To formulate a mini-bucket algorithm for approximate inference in terms of exact inference on an approximate model produced by splitting nodes in a Bayesian network.

Purpose

To formulate a mini-bucket algorithm for approximate inference in terms of exact inference on an approximate model produced by splitting nodes in a Bayesian network.

- Reduced search space

Purpose

To formulate a mini-bucket algorithm for approximate inference in terms of exact inference on an approximate model produced by splitting nodes in a Bayesian network.

- Reduced search space
- New mini-bucket heuristics

Purpose

To formulate a mini-bucket algorithm for approximate inference in terms of exact inference on an approximate model produced by splitting nodes in a Bayesian network.

- Reduced search space
- New mini-bucket heuristics
- Mini-bucket benefit from recent advances in exact inference

Introducing a problem

Most probable explanation - *MPE*

Definition

Given a Bayesian network N with variables x, inducing distribution Pr. Then the *MPE* for some evidence e is:

- $MPE(N, e) = argmax Pr(x)$, where x and e are compatible.

Note solution may not be unique. The *MPE* probability is:

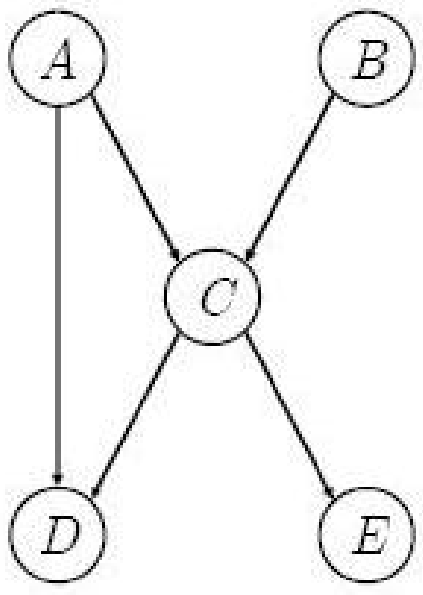
- $MPE_p(N, e) = max Pr(x)$.

Example: Splitting nodes

- Split according to children
- Split along an edge
- Fully split

Key property of split networks

$MPE_p(N, e) \leq \beta MPE_p(N', e, \vec{e})$



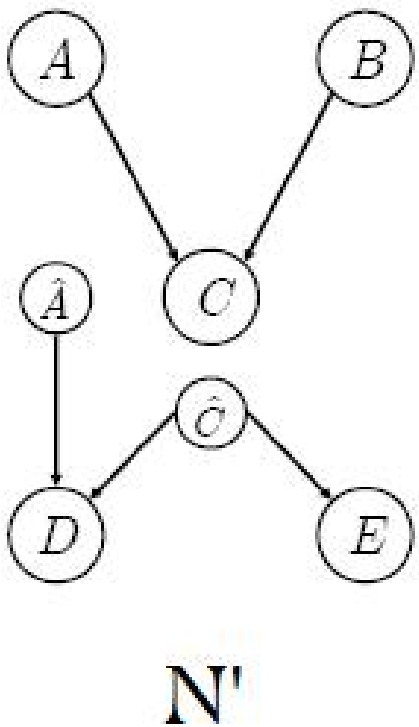
N

Example: Splitting nodes

- Split according to children
- Split along an edge
- Fully split

Key property of split networks

$MPE_p(N, e) \leq \beta MPE_p(N', e, \vec{e})$



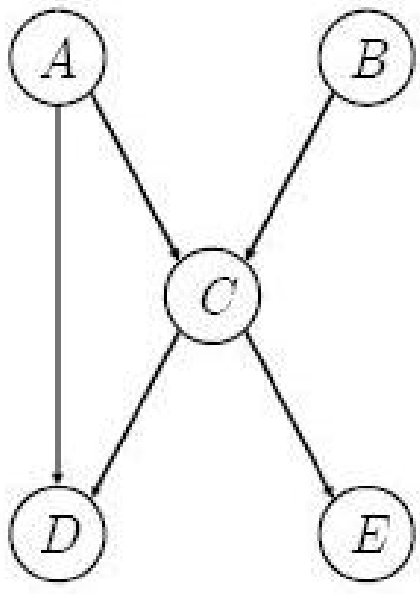
Mini-bucket elimination

What is mini-bucket elimination?

Mini-bucket elimination is a simple variation of the variable elimination algorithm.

Mini-bucket elimination

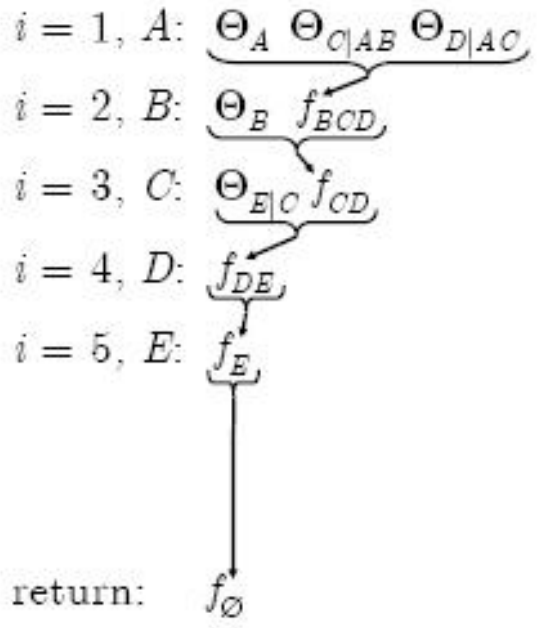
- Bayesian network N
  1. Variable elimination
  2. Mini-bucket elimination



N

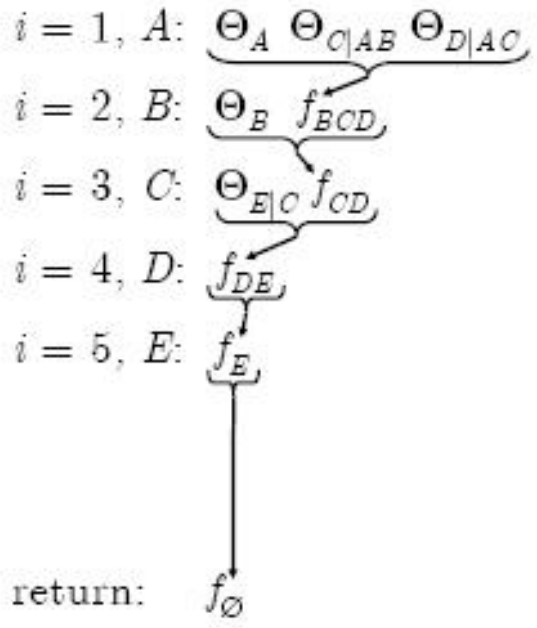
Example: Variable elimination

- Algorithm 1 $VE(N, e)$
- Variable elimination on N



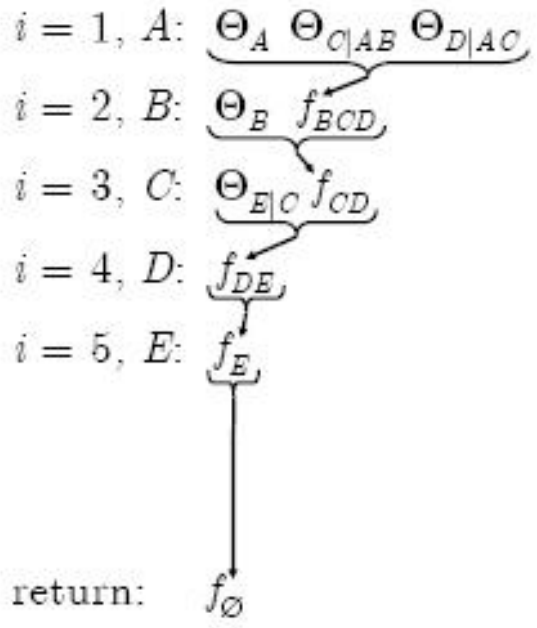
Example: Variable elimination

- Algorithm 1 $VE(N, e)$
- Variable elimination on N
- To eliminate variable X
  1. Select **all** factors that contain X (line 6)
  2. Multiply all factors that contain X and max-out X from the result (line 7)



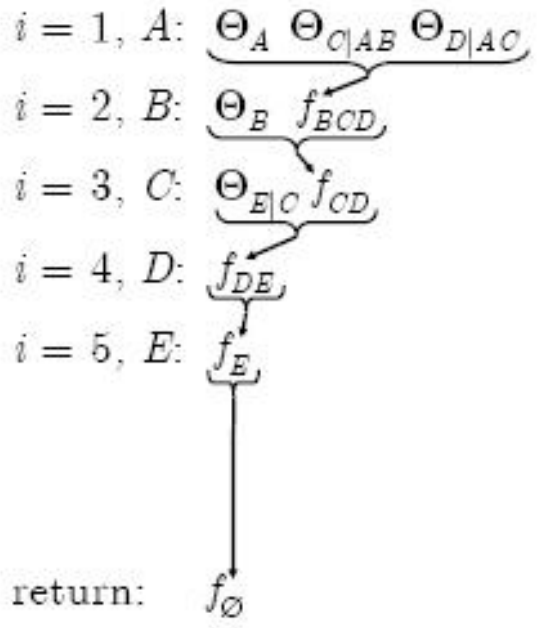
Example: Variable elimination

- Algorithm 1 $VE(N, e)$
- Variable elimination on N
- To eliminate variable X
  1. Select **all** factors that contain X (line 6)
  2. Multiply all factors that contain X and max-out X from the result (line 7)
- Returns $MPE_p(N, e)$



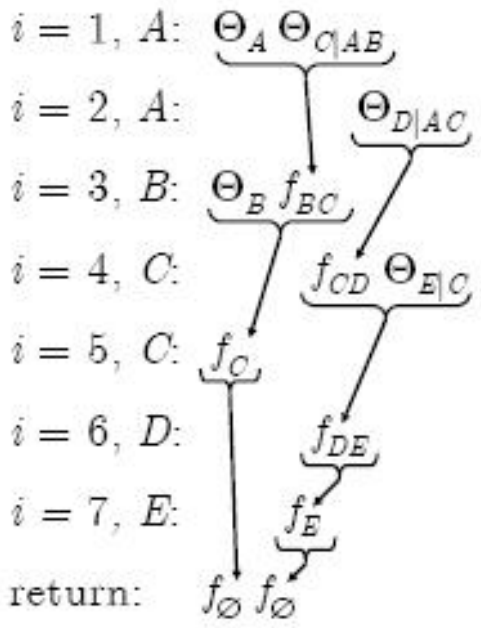
Example: Variable elimination

- Algorithm 1 $VE(N, e)$
- Variable elimination on N
- To eliminate variable X
  1. Select **all** factors that contain X (line 6)
  2. Multiply all factors that contain X and max-out X from the result (line 7)
- Returns $MPE_p(N, e)$
- Bottleneck on computational resources when multiplying



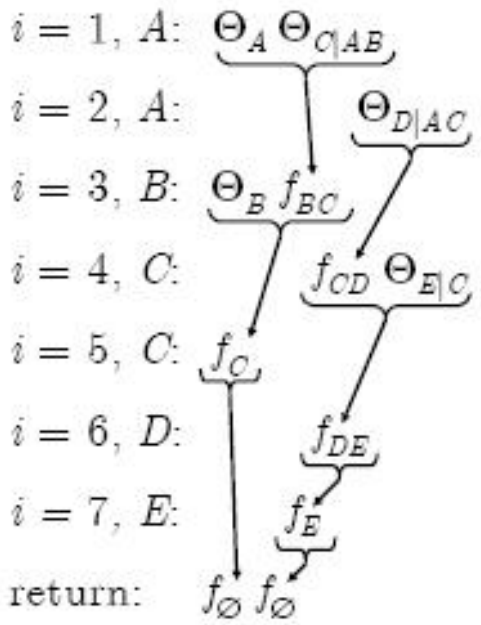
Example: Mini-bucket elimination

- Algorithm 2 $MBE(N, e)$
- Mini-bucket elimination on N



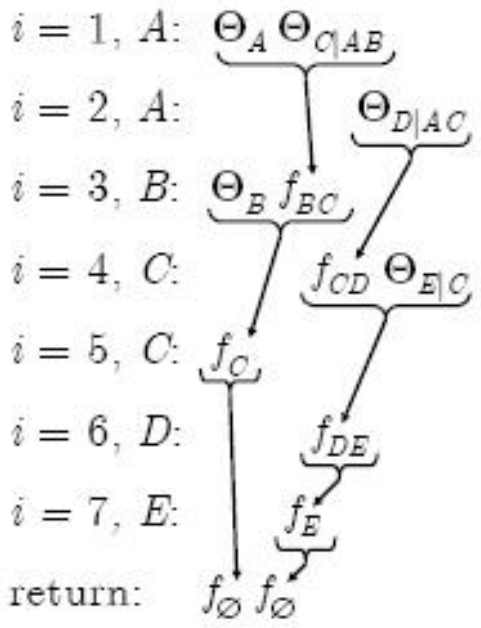
Example: Mini-bucket elimination

- Algorithm 2 $MBE(N, e)$
- Mini-bucket elimination on N
- To eliminate variable X
  1. Select **some** factors that contain X (line 6)
  2. Multiply the selected factors that contain X and max-out X from the result (line 7)
  3. Repeat till X have been completely removed



Example: Mini-bucket elimination

- Algorithm 2 $MBE(N, e)$
- Mini-bucket elimination on N
- To eliminate variable X
  1. Select **some** factors that contain X (line 6)
  2. Multiply the selected factors that contain X and max-out X from the result (line 7)
  3. Repeat till X have been completely removed
- Returns an upper bound on $MPE_p(N, e)$



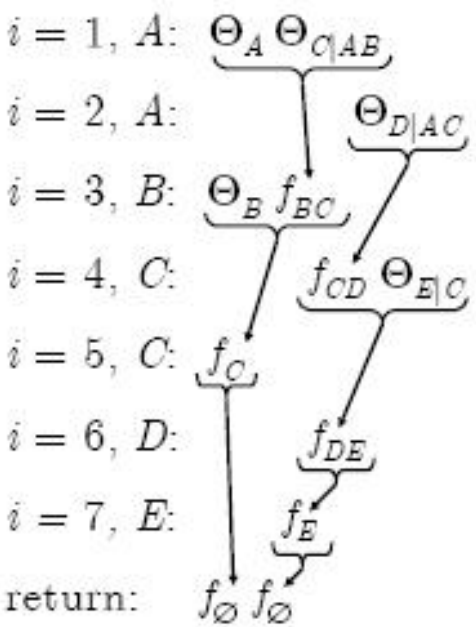
Correspondence

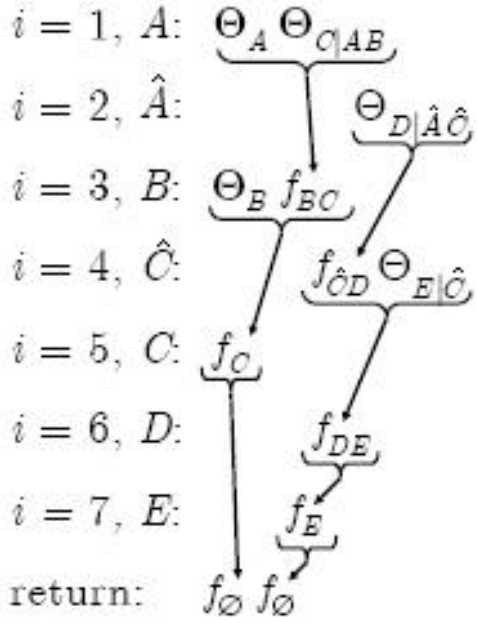
Node splitting approach

Algorithm 3 $SPLIT - MBE(N, e)$

Given a network N and evidence e, algorithm 3 returns a split network N' and a variable ordering π', that corresponds to a run of mini-bucket elimination on N and e.

Example: Correspondence





New mini-bucket heuristics

- Given the correspondences, every mini-bucket heuristic can be interpreted as a node splitting strategy.
  1. Old mini-bucket heuristics
  2. New mini-bucket heuristics

Old mini-bucket heuristics

Given a bound on the size of the largest factor:

1. Choose variable order
2. Pick set that is within the given bound of X
3. Repeat 2 till variable X is eliminated
4. Continue with next variable

- Seeks to minimize the number of clones introduced into the approximation N'

New mini-bucket heuristics

Given a bound on the largest jointree cluster

1. Build a jointree of the network
2. Pick variable X who's removal will introduce the largest reduction in the sizes of the cluster and seperator tables
3. Fully split variable X
4. Repeat till the bound is met

- Seeks to minimize the number of split variables

Reduced search space

Proposition 1

$MPE_p(N, z) \leq \beta MPE_p(N', z, \vec{z})$

- Z contains all variables that were split in N to produce N'
- Once all variables in Z have been instantiated, the approximation is exact
- Once the bound on MPE_p becomes exact, no better solution is reachable in N'

Thus

- The search space size is exponential in the number of split variables, down from being exponential in the number of network variables

Additional result

Mini-buckets benefit from recent advances in exact inference

- The evaluation of the mini-bucket approximation need not rely on any specific exact inference algorithm
- Node splitting approach in combination with a state-of-the-art arithmetic circuit outperform variable elimination