TPLP: Page 1–X. © The Author(s), 2022. Published by Cambridge University Press 2022
doi:10.1017/xxxxx

Appendix: Efficient Knowledge Compilation Beyond Weighted Model Counting

RAFAEL KIESEL*, PIETRO TOTIS†, ANGELIKA KIMMIG†
*TU Vienna; †KU Leuven

submitted xx xx xxxx; revised xx xx xxxx; accepted xx xx xxxx

Appendix A Full Proofs and Omitted Lemmas

A.1 Full Proofs

Here, we restate the Theorems and Lemmas from the main paper and give their full proofs.

Theorem 4 (*Tractable* 2AMC *with* $\mathbf{X}_O$*-first sd-DNNFs*)
Let $A = (\mathcal{T}, \mathbf{X}_I, \mathbf{X}_O, \alpha_I, \alpha_O, \mathcal{S}_I, \mathcal{S}_O, t)$ be a 2AMC instance, where $\mathcal{T}$ is an $\mathbf{X}_O$-first sd-DNNF. Then, we can compute $2\text{AMC}(A)$ in linear time in the size of $\mathcal{T}$.

Proof
The basic idea is as follows: We can see $\mathcal{T}$ as an algebraic circuit, by replacing or-nodes, and-nodes, false and true by sum, product, zero and one, respectively and by replacing all literals by their weight. Then we only need to make sure to use the sum, product, zero and one from the correct semiring: For pure nodes n such that $Vars(n) \subseteq X_I$ this is the inner one for all other nodes it is the outer one. Additionally, for mixed nodes $n = n_1 \otimes^O n_2$, where w.l.o.g. $Vars(n_1) \subseteq X_I$ we need to use $t(n_1) \otimes^O n_2$ to have values that are over the same semiring.

Consider a subgraph n of $\mathcal{T}$ with exactly one outgoing edge for each or-node and all outgoing edges for each and-node. As $\mathcal{T}$ satisfies $\mathbf{X}_O$-first and is smooth, there is a node n' in n such that $Vars(n') = X_I$, i.e., exactly the outer variables occur above n' (see also the lowest and-nodes of the left NNF in Figure **??**). Thus, n' is equivalent to $\mathcal{T}\,|_{\mathbf{x}_O}$ for some assignment $\mathbf{x}_O$ to the outer variables, for which n' computes the value of the inner AMC instance. As evaluation sums over all these subgraphs, it obtains the correct result. □

Lemma 9 (*Exponential Separation*)
Let $\mathcal{T} = \bigwedge_{i=1}^{n} X_i \leftrightarrow Y_i$, $\mathbf{X} = \{X_1, \ldots, X_n\}$ and $\mathbf{D} = \{Y_1, \ldots, Y_n\}$, then the size of the smallest $\mathbf{X}$-first sd-DNNF for $\mathcal{T}$ is exponential in n and the size of the smallest $\mathbf{X}/\mathbf{D}$-first sd-DNNF for $\mathcal{T}$ is linear in n.

Proof
Let $\mathcal{C}$ be a $\mathbf{X}$-first sd-DNNF for $\mathcal{T}$. Recall that this implies that for every assignment $\mathbf{x}$ to $\mathbf{X}$ there exists a node $n_\mathbf{x}$ in $\mathcal{C}$ such that $n_\mathbf{x}$ is equivalent to $\mathcal{T}\,|_\mathbf{x}$. Since $\mathcal{T}\,|_\mathbf{x}$ and $\mathcal{T}\,|_{\mathbf{x}'}$ are different, whenever $\mathbf{x}$ and $\mathbf{x}'$ are different there are 2^n different residual theories $\mathcal{T}\,|_\mathbf{x}$, one for each $\mathbf{x} \in int(\mathbf{X})$. This implies that $\mathcal{C}$ has at least 2^n different nodes, which proves the first part of the lemma.

We still need to prove that there exists a $\mathbf{X}/\mathbf{D}$-first sd-DNNF $\mathcal{S}$ for $\mathcal{T}$ the size of which is linear in n. For this we first consider, what $\mathbf{X}/\mathbf{D}$-firstness means for $\mathcal{T}$. The variables that are defined by $\mathbf{X}$ in terms of $\mathcal{T}$ are $\mathbf{D} = \{Y_1, \ldots, Y_n\}$. Thus, any node over variables $\mathbf{X} \cup \mathbf{D}$ is a pure node and every NNF over these variables is an $\mathbf{X}/\mathbf{D}$-first NNF. This means that we can use any sd-DNNF $\mathcal{S}$ for $\mathcal{T}$. It is easy to see, that when one decides variables according to the variable order $X_1, Y_1, \ldots, X_n, Y_n$ the resulting NNF is of size linear in n. □

Theorem 12 (*Tractable AMC with $\mathbf{X}/\mathbf{D}$-first sd-DNNFs*)
The value of a 2AMC instance $A = (\mathcal{T}, \mathbf{X}_I, \mathbf{X}_O, \alpha_I, \alpha_O, \mathcal{S}_I, \mathcal{S}_O, t)$ can be computed in polynomial time under the following conditions:

- $\mathcal{T}$ is an $\mathbf{X}/\mathbf{D}$-first sd-DNNF
- t is a monoid homomorphism from the monoid $\mathcal{M} = \langle O(A) \rangle_{(R_I, \otimes^I, e_{\otimes^I})}$ generated by the observable values to $(R_O, \otimes^O, e_{\otimes^O})$.

Proof (Sketch).
Let n be the root of $\mathcal{T}$. We know that the variables in $\mathbf{D}(n, \mathbf{X}_O)$ are defined by $\mathbf{X}_O$ in terms of T. For $d \in \mathbf{D}(n, \mathbf{X}_O)$ and $\mathbf{x}_O \in int(\mathbf{X}_O)$, we denote by $d\mid_{\mathbf{x}_O}$ the literal of d that must be included in $\mathbf{x}_I$ in order for $\mathbf{x}_I \cup \mathbf{x}_O$ to be a satisfying assignment (if $\mathbf{x}_O$ can be extended to a satisfying assignment, otherwise choose an arbitrary but fixed value).

Recall that the value of A is defined as

$$\bigoplus^O_{\mathbf{x}_O \in int(\mathbf{X}_O)} \bigotimes^O_{x \in \mathbf{x}_O} \alpha_O(x) \otimes^O t\left(\bigoplus^I_{\mathbf{x}_I \in int(\mathbf{X}_I), \mathbf{x}_I \cup \mathbf{x}_O \models T} \bigotimes^I_{y \in \mathbf{x}_I} \alpha_I(y)\right).$$

Since the inner sum only takes interpretations that satisfy T, we do not need to take the sum over both values of a defined variable d but can restrict ourselves to the value $d\mid_{\mathbf{x}_O}$ determined by $\mathbf{x}_O$.

$$\begin{aligned}
&\bigoplus^I_{\mathbf{x}_I \in int(\mathbf{X}_I), \mathbf{x}_I \cup \mathbf{x}_O \models T} \bigotimes^I_{y \in \mathbf{x}_I} \alpha_I(y) \\
=&\bigoplus^I_{\mathbf{x}_I \in int(\mathbf{X}_I \setminus \{d\}), \mathbf{x}_I \cup \mathbf{x}_O \models T} \otimes^I \alpha_I(d\mid_{\mathbf{x}_O}) \otimes^I \bigotimes^I_{y \in \mathbf{x}_I} \alpha_I(y) \\
=&\alpha_I(d\mid_{\mathbf{x}_O}) \otimes^I \bigoplus^I_{\mathbf{x}_I \in int(\mathbf{X}_I \setminus \{d\}), \mathbf{x}_I \cup \mathbf{x}_O \models T} \bigotimes^I_{y \in \mathbf{x}_I} \alpha_I(y)
\end{aligned}$$

Next, we can plug these equalities into the expression for the value of A and use that t is a homomorphism.

$$\begin{aligned}
&\bigoplus^O_{\mathbf{x}_O \in int(\mathbf{X}_O)} \bigotimes^O_{x \in \mathbf{x}_O} \alpha_O(x) \otimes^O t\left(\bigoplus^I_{\mathbf{x}_I \in int(\mathbf{X}_I), \mathbf{x}_I \cup \mathbf{x}_O \models T} \bigotimes^I_{y \in \mathbf{x}_I} \alpha_I(y)\right) \\
=&\bigoplus^O_{\mathbf{x}_O \in int(\mathbf{X}_O)} \bigotimes^O_{x \in \mathbf{x}_O} \alpha_O(x) \otimes^O t\left(\alpha_I(d\mid_{\mathbf{x}_O}) \otimes^I \bigoplus^I_{\mathbf{x}_I \in int(\mathbf{X}_I \setminus \{d\}), \mathbf{x}_I \cup \mathbf{x}_O \models T} \bigotimes^I_{y \in \mathbf{x}_I} \alpha_I(y)\right) \\
=&\bigoplus^O_{\mathbf{x}_O \in int(\mathbf{X}_O)} \bigotimes^O_{x \in \mathbf{x}_O} \alpha_O(x) \otimes^O t(\alpha_I(d\mid_{\mathbf{x}_O})) \otimes^O t\left(\bigoplus^I_{\mathbf{x}_I \in int(\mathbf{X}_I \setminus \{d\}), \mathbf{x}_I \cup \mathbf{x}_O \models T} \bigotimes^I_{y \in \mathbf{x}_I} \alpha_I(y)\right)
\end{aligned}$$

Due to fact that t satisfies $t(e_{\otimes^I}) = e_{\otimes^O}, t(e_{\oplus^I}) = e_{\oplus^O}$, whenever the assignment $\mathbf{x}_O$ cannot be extended to a satisfying of T, the weight for the given assignment will be $e_{\oplus^O}$.

Thus, we can again use the fact that the variable d is defined and sum over both of its values *in the outer sum*, resulting in

$$\begin{aligned}
&\bigoplus^O_{(\mathbf{x}_O, \mathbf{d}) \in int(\mathbf{X}_O \cup \{d\}} \bigotimes^O_{x \in \mathbf{x}_O} \alpha_O(x) \otimes^O t\left(\bigotimes^I_{d \in \mathbf{d}} \alpha_I(d)\right) \otimes^O t\left(\bigoplus^I_{\mathbf{x}_I \in int(\mathbf{X}_I \setminus \{d\}), \mathbf{x}_I \cup \mathbf{x}_O \models T} \bigotimes^I_{y \in \mathbf{x}_I} \alpha_I(y)\right) \\
=&\bigoplus^O_{(\mathbf{x}_O, \mathbf{d}) \in int(\mathbf{X}_O \cup \{d\})} \bigotimes^O_{x \in \mathbf{x}_O} \alpha_O(x) \otimes^O \bigotimes^O_{d \in \mathbf{d}} t(\alpha_I(d)) \otimes^O t\left(\bigoplus^I_{\mathbf{x}_I \in int(\mathbf{X}_I \setminus \{d\}), \mathbf{x}_I \cup \mathbf{x}_O \models T} \bigotimes^I_{y \in \mathbf{x}_I} \alpha_I(y)\right)
\end{aligned}$$

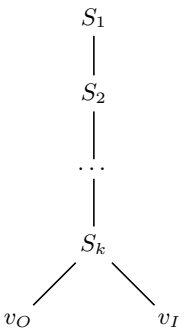

Fig. A 1: Schematic variable order constructed in the proof of Lemma **??**.

We observe that this expression is equal to the value of another 2AMC instance $B = (T, \mathbf{X}_I \setminus \{d\}, \mathbf{X}_O \cup \{d\}, \beta_I, \beta_O, \mathcal{R}_I, \mathcal{R}_O)$, where

$$\beta_I(x) = \alpha_I(x) \quad \text{for } x \in lit(\mathbf{X}_I \setminus \{d\})$$
$$\beta_O(x) = \begin{cases} \alpha_O(x) & \text{if } x \in lit(\mathbf{X}_O), \\ t(\alpha_I(x)) & \text{if } x \in lit(\{d\}). \end{cases}.$$

Now, if d occurs before some variable $x \in \mathbf{X}$ in some node n of $\mathcal{T}$, then n is not an $\mathbf{X}$-first NNF. However, it is an $\mathbf{X}_O \cup \{d\}$-first sd-DNNF. On this NNF we can solve the 2AMC-instance B in polynomial time according to Theorem **??**. By induction on the number of variables that occur before x the claim follows. □

Lemma 14
Let $\mathcal{T}$ be a CNF over variables $\mathbf{Y}$ and (T, χ) a TD of PRIM($\mathcal{T}$) of width k. Furthermore, let $\mathbf{X} \subseteq \mathbf{Y}$ and $\mathbf{D} = \mathbf{D}(\mathcal{T}, \mathbf{X})$. If there exists $t^* \in V(T)$ such that (1) $\chi(t^*) \subseteq \mathbf{X} \cup \mathbf{D}$ and (2) $\chi(t^*)$ is a *separator* of $\mathbf{X}$ and $\mathbf{Y} \setminus (\mathbf{X} \cup \mathbf{D})$, i.e., every path from $\mathbf{X}$ to $\mathbf{Y} \setminus (\mathbf{X} \cup \mathbf{D})$ in PRIM($\mathcal{T}$) uses a vertex from $\chi(t^*)$, then we can compile $\mathcal{T}$ into an $\mathbf{X}/\mathbf{D}$-first sd-DNNF in time $\mathcal{O}(2^k \cdot \text{poly}(|\mathcal{T}|))$.

Proof
The performance guarantee is due to Korhonen and Järvisalo (2021) and holds when we decide the variables in the order they occur in the TD starting from the root. $\mathbf{X}/\mathbf{D}$-firstness can be guaranteed by taking t^* as the root of the TD and, thus, first deciding all variables in $\chi(t^*) = \{S_1, \ldots, S_n\}$. From condition (2) it follows that afterwards the CNF has decomposed into separate components, which either only use variables from $\mathbf{X} \cup \mathbf{D}$ or use no variables from $\mathbf{X}$. Thus, their compilation only leads to pure NNFs.

The variable order can be inspected schematically in Figure A 1. Here, v_I is the remaining variable order for the inner variables and v_O is the remaining variable order for the outer (and possibly defined) variables. The split signifies that the CNF composes into different components and we can consider respective variable orders for them independently. □

A.2 Omitted Lemmas Showing Homomorphism Property

Here, we give proofs for the fact that the transformation functions of the different 2AMC problems are homomorphisms.

Lemma i
The function

$$h : \mathbb{N}^2 \to [0,1], (n_1, n_2) \mapsto \begin{cases} 0 & \text{if } n_2 = 0, \\ {}^{n_1}/_{n_2} & \text{otherwise.} \end{cases}$$

is a monoid homomorphism from $(\{(n_1, n_2) \mid n_1 \leq n_2, n_1, n_2 \in \mathbb{N}\}, \cdot, (1,1))$, where multiplication is coordinatewise, to $([0,1], \cdot, 1)$.

Note that if n_2 is zero then also n_1 is zero, so no other division by zero is avoided by h.

Proof
It is clear that $h((1,1)) = 1$, therefore the neutral element is preserved. Furthermore, let $(n_1, n_2), (n_3, n_4) \in \mathbb{N}^2$. First assume, that one of n_1 and n_3 is zero. In this case,

$$h((n_1, n_2) \cdot (n_3, n_4)) = h((0, n_2 n_4)) = 0 = h((n_1, n_2))h((n_3, n_4)),$$

since one of $h((n_1, n_2)$ and $h((n_3, n_4))$ is zero. Otherwise, both n_1 and n_3 and therefore also n_2 and n_4 are unequal to zero. Then

$$h((n_1, n_2) \cdot (n_3, n_4)) = h((n_1 n_3, n_2 n_4)) = \frac{n_1 n_3}{n_2 n_4} = \frac{n_1}{n_2}\frac{n_3}{n_4} = h((n_1, n_2))h((n_3, n_4)).$$

□

Lemma ii
Let A be a 2AMC instance corresponding to the evaluation problem of an DTProbLog program. Then

$$h : \{(p, pu) \mid p \in [0,1], u \in \mathbb{R}\} \to \mathbb{R} \cup \{-\infty\}, (p, pu) \mapsto \begin{cases} -\infty & \text{if } p = 0 \\ pu & \text{otherwise.} \end{cases}.$$

is a monoid homomorphism from the monoid generated by the observable values to $(\mathbb{R} \cup \{-\infty\}, +, 0)$.

While h is not a homomorphism on the whole set of values, it is one on the set of observable values, since they are all of the form (p, pu) for $p \in \{0, 1\}$.

Proof
For ProbLog programs it is known, that probabilistic facts are not defined, instead, every assignment of the probabilistic facts can be uniquely extended to a satisfying assignment of the program.

This implies two things that are useful for us. Firstly, the probability of a ProbLog program being satisfied is always 1. Since an DTProbLog program is a ProbLog program conditioned on any complete assignment to the decision variables, this also implies that the expected utility of the conditioned program is of the form $(1, u)$. This means that all the values of the inner sum are of the form $(1, u)$ or $(0, 0)$.

Secondly, since probabilistic facts cannot be defined, any defined variable must have weights of the form $(1, u)$ as well, which also implies that even if we take out the defined variables from the inner sum, the value of the inner sum is still of the form $(1, u)$ or $(0, 0)$.

Overall, it follows that the monoid generated by the observable values is as submonoid of

$$\mathcal{M} = (\{(1, u) \mid u \in \mathbb{R}\} \cup (0, 0), \otimes, (1, 1)).$$

Now, the only thing left to prove is that h is a monoid homomorphism from $\mathcal{M}$ to $(\mathbb{R} \cup \{-\infty\}, +, 0)$. This can again be easily as follows.

Let $(p, pu), (p', p'u') \in \mathcal{M}$. If p or p' are zero, we have

$$h((p, pu) \otimes (p', p'u')) = h((0, 0)) = -\infty = h((p, pu))h((p', p'u')).$$

Otherwise, we know that both are one. Then

$$h((1, u) \otimes (1, u')) = h((1, u + u')) = u + u' = h((1, u)) + h((1, u')).$$

□

References

KORHONEN, T. AND JÄRVISALO, M. Integrating tree decompositions into decision heuristics of propositional model counters (short paper). In *CP* 2021, volume 210 of *LIPIcs*, pp. 8:1–8:11.

Acknowledgements This work has been supported by the Austrian Science Fund (FWF) Grant W1255-N23 and by the Fonds Wetenschappelijk Onderzoek (FWO) project N. G066818N.