Appendix

Proof of Theorem 4

We first prove that for the Walsh-Fourier Expansion of a clause c, denoted by $\texttt{WFE}$, the probability that c is satisfied by $\mathcal{R}(a)$, i.e., $\mathbb{P}[\texttt{WFE}(\mathcal{R}(a))=1]$ equals $\texttt{WFE}(a)$. Then by the definition of the objective function and linearity of expectation, Theorem 4 follows directly.

We prove $\mathbb{P}[\texttt{WFE}(\mathcal{R}(a))=1]=\texttt{WFE}(a)$ by induction on the number of variables n.

`Basis step`: Let $n=1$. $\texttt{WFE}$ is either constant or $\frac{1+x_1}{2}$, or $\frac{1-x_1}{2}$. It is easy to verify that the statement holds.

`Inductive step`: Suppose $n\geq 2$. Then, by Boole's Expansion, $\texttt{WFE}(\mathcal{R}(a))$ can be expanded as:

$$\texttt{WFE}(\mathcal{R}(a))=\tfrac{1-\mathcal{R}(a)_n}{2}\cdot\texttt{WFE}_{n\to(-1)}(\mathcal{R}(a)_{[n-1]})+\tfrac{1+\mathcal{R}(a)_n}{2}\cdot\texttt{WFE}_{n\to 1}(\mathcal{R}(a)_{[n-1]}),$$

where $\mathcal{R}(a)_n$ is the value of the n-th coordinate of $\mathcal{R}(a)$, $\mathcal{R}(a)_{[n-1]}\in\mathbb{R}^{n-1}$ is the point after removing the n-th coordinate of $\mathcal{R}(a)$, $\texttt{WFE}_{n\to 1}$ denotes the function of $\texttt{WFE}$ after fixing the value of x_n to 1.

Note that the value of $\mathcal{R}(a)_n$ and $\mathcal{R}(a)_{[n-1]}$ are independent, thus

$$\begin{aligned}\mathbb{P}[\texttt{WFE}(\mathcal{R}(a))=1]=&\mathbb{P}[\mathcal{R}(a)_n=-1]\cdot\mathbb{P}[\texttt{WFE}_{n\to(-1)}(\mathcal{R}(a)_{[n-1]})=1]+\mathbb{P}[\mathcal{R}(a)_n=1]\cdot\mathbb{P}[\texttt{WFE}_{n\to 1}(\mathcal{R}(a)_{[n-1]})=1]\\ =&\frac{1-a_n}{2}\cdot\texttt{WFE}_{n\to -1}(a_{[n-1]})+\frac{1+a_n}{2}\cdot\texttt{WFE}_{n\to 1}(a_{[n-1]})\text{ (by I.H.)}\\ =&\texttt{WFE}(a),\end{aligned}$$

$\square$

Proof of Theorem 3

Note that by $\mathbb{P}[\texttt{WFE}_c(\mathcal{R}(a))=1]=\texttt{WFE}_c(a)$ for all constraint $c\in C_f$, proved in the proof of Theorem 4, we have $\texttt{WFE}_c(a)\in[0,1]$ for all constraint c and $a\in[-1,1]^n$. Thus $F_{f,w}(a)\leq\sum_{c\in C_f}w(c)$ for all $a\in[-1,1]^n$ since $w:C_F\to\mathbb{R}^+$ is a positive function.

– "$\Rightarrow$": Suppose f is satisfiable and $b\in\{\pm1\}^n$ is one of its solutions. Then b is also a solution of every constraint of f. Thus for every $c\in C_f$, $\texttt{WFE}_c(b)=1$. Therefore $F_{f,w}(b)=\sum_{c\in C_f}w(c)$ and $\max\limits_{x\in[-1,1]^n}F_{f,w}(x)=\sum_{c\in C_f}w(c)$.

– "$\Leftarrow$": Suppose $\max\limits_{x\in[-1,1]^n}F_{f,w}(x)=\sum_{c\in C_f}w(c)$. Thus $\exists\, a$ such that $F_f(a)=\sum_{c\in C_f}w(c)$. Since $\texttt{WFE}_c(a)\in[0,1]$, we have $\texttt{WFE}_c(a)=1$ for every $c\in C_f$. Since $\texttt{WFE}_c(a)=\mathbb{P}[\texttt{WFE}_c(\mathcal{R}(a))=1]=1$, rounding a arbitrarily to $b\in\{-1,1\}^n$ will give us $\texttt{WFE}_c(b)=1$ for all constraint c. Thus b is a solution of all constraints, which is also a solution of the formula f. $\square$

Proof of Corollary 1

$$\begin{aligned}\mathop{\mathbb{E}}_{b\sim\mathcal{S}_a}[F_{f,w}(b)]&=\mathop{\mathbb{E}}_{b\sim\mathcal{S}_a}[\sum_{c\in C_f}w(c)\cdot\texttt{WFE}_c(b)]\\ &=\sum_{c\in C_f}w(c)\cdot\mathop{\mathbb{E}}_{b\sim\mathcal{S}_a}[\texttt{WFE}_c(b)]\\ &=\sum_{c\in C_f}w(c)\cdot\mathop{\mathbb{P}}_{b\sim\mathcal{S}_a}[c(b)=1]\\ &=\sum_{c\in C_f}w(c)\cdot\texttt{COP}(P_a,c)\end{aligned}$$

Proof of Theorem 5

Proof Sketch. *We first prove a result for single-rooted BDDs in Lemma 2. Then we show that running Algorithm 3 on individual single-rooted BDDs gives the same result with running Algorithm 3 on a corresponding multi-rooted BDD in Lemma 3.*

Lemma 2. *Let B be a single-rooted BDD and run Algorithm 3 on B, a real assignment $a\in[-1,1]^n$ and the constraint weight w. Let $b\in\{-1,1\}^n$ be the randomly rounded assignment from a by $\mathbb{P}[b_i=-1]=\frac{1-a_i}{2}$. Then for each node $v\in B.V$, we have*

$$M_{TD}[v]=w\cdot\mathcal{P}(B,a,v),$$

where $\mathcal{P}(B,a,v)$ is the probability that the node v is on the path generated by b on B.

Especially,

$$M_{TD}[B.one]=w\cdot\mathit{COP}(P_a,c)$$

where $P_a(x_i)=\frac{1-a_i}{2}$ for all $i\in\{1,\cdots,n\}$.

Proof. *We prove the statement by structural induction on single-rooted BDD B.*
`Basis step`*: For the root r of B,*

$$M_{TD}[B.r] = 1$$

the statement holds because r is on the path generated by every discrete assignment on B.
`Inductive step`*: For each non-root node v of B, let $par_T(v, B)$ be the set of parents of v with an edge labeled by `True` (-1) and $par_F(v, B)$ be the set of parents of v with an edge labeled by `False` (1) in BDD B, after Algorithm 3 terminates, we have*

$$M_{TD}[v] = \sum_{u \in par_T(v)} \frac{1 - a_i}{2} \cdot M_{TD}[u] + \sum_{u \in par_F(v)} \frac{1 + a_i}{2} \cdot M_{TD}[u]$$

By inductive hypothesis,

$$\begin{aligned} M_{TD}[v] &= \sum_{u \in par_T(v)} w \cdot Pr[b_i = -1] \cdot \mathcal{P}(B, a, u) + \sum_{u \in par_F(v)} w \cdot Pr[b_i = 1] \cdot \mathcal{P}(B, a, u) \\ &= w \cdot \mathcal{P}(B, a, v) \end{aligned}$$

The last equality holds because the events of reaching v from different parents are exclusive. □

Next we will generalize our result for multi-rooted BDDs.

Lemma 3. *Let $C_f = \{c_1, \cdots, c_m\}$ be the constraints set of formula f. Let $B_1, \cdots, B_m$ be the individual BDDs corresponding to constraints in C_f. Let B be the multi-rooted BDD for the whole set C_f. Suppose we run:*

- *Algorithm 3 with (B, a, w) and store M_{TD}.*
- *Algorithm 3 with $(B_i, a, w(c_i))$ for each constraint c_i and store m mappings $\{M^1_{TD}, \cdots M^m_{TD}\}$. Then for each node $v \in B.V$, suppose the equivalent nodes of v also appear in $\mathcal{B}_v = (B_{I_{v1}}, \cdots, B_{I_{vk}})$ as $\mathcal{V}_v = (v_{J_{v1}}, \cdots, v_{J_{vk}})$, respectively.*

Then we have

$$M_{TD}[v] = \sum_{j=1}^{|I_v|} M^{I_{vj}}_{TD}[v_{J_{vj}}]$$

Proof. *We prove by induction on the multi-rooted BDD B.*
`Basis step`*: For each node $r \in B.V$ with in-degree 0, we have*

$$M_{TD}[v] = \sum_{j=1}^{|I_v|} w(c_{I_{vj}}) = \sum_{j=1}^{|I_v|} M^{I_{vj}}_{TD}[v_{J_{vj}}],$$

since v does not have parent nodes in B and all B_i's.
`Inductive step`*: For each node $v \in B.V$ that has at least one parent, let $par_T(v)$ be the set of parents of v with an edge labeled by `True` (-1) and $par_F(v)$ be the set of parents of v with an edge labeled by `False` (1). The equivalence of v can be either a root in some BDDs or non-root nodes in some other BDDs. Suppose the equivalence of v is the root in BDDs with constraint index $R_v = \{R_{v1}, \cdots, R_{vd}\}$.*

Consider the situation after Algorithm 3 terminates on MRBDD B, we have

$$M_{TD}[v] = \sum_{u \in par_T(v,B)} \frac{1 - a_i}{2} \cdot M_{TD}[u] + \sum_{u \in par_F(v,B)} \frac{1 + a_i}{2} \cdot M_{TD}[u] + \sum_{j=1}^{|R_v|} w(c_{vj})$$

By inductive hypothesis,

$$M_{TD}[u] = \sum_{j=1}^{|I_u|} M^{I_{uj}}_{TD}[v_{J_{uj}}].$$

for all $u \in par_T(v, B) \cup par_F(v, B)$.

Thus

$$
\begin{aligned}
M_{TD}[v] &= \sum_{u\in par_T(v,B)} \frac{1-a_i}{2}\cdot\sum_{j=1}^{|I_u|} M_{TD}^{I_{uj}}[v_{J_{uj}}] + \sum_{u\in par_F(v,B)} \frac{1+a_i}{2}\cdot\sum_{j=1}^{|I_u|} M_{TD}^{I_{uj}}[v_{J_{uj}}] + \sum_{j=1}^{|R_v|} w(c_{R_{vj}}) \\
&= \sum_{j=1}^{|I_v|}\Big(\sum_{u\in par_T(v,B_{I_{vj}})} \frac{1-a_i}{2}\cdot M_{TD}^{I_{uj}}[v_{I_{uj}}] + \sum_{u\in par_F(v,B_{I_{vj}})} \frac{1+a_i}{2}\cdot M_{TD}^{I_{uj}}[v_{I_{uj}}] + \mathbb{I}(I_{vj}\in R_v)\cdot w(c_{R_{vj}})\Big) \\
&= \sum_{j=1}^{|I_v|} M_{TD}^{I_{vj}}[v_{I_{vj}}]
\end{aligned}
$$

□

Now we are ready to prove Theorem 5.

By the fact that the node labeled by constant `one` appears in every individual BDD B_i, we have

$$
\begin{aligned}
M_{TD}[B.one] &= \sum_{c_i\in C_f} M_{TD}^{i}[B_i.one] \quad \text{(Lemma 3)} \\
&= \sum_{c_i\in C_f} w(c_i)\cdot \texttt{COP}(P_a, c_i) \quad \text{(Lemma 2)} \\
&= F_{f,w}(a). \quad \text{(Corollary 1)}
\end{aligned}
$$

Complexity: Algorithm 3 traverse the multi-rooted BDD once. The topological sort also takes $O(S)$ time (e.g. Kahn's algorithm). In practice, since the topological order is indicated by the variable index, topological sort can be conveniently implemented by a priority queue with the variable index as the key, although this would make the complexity $O(SlogS)$. □

Proof of Theorem 6

To prove Theorem 6, we introduce the following lemma.

Lemma 4. *Suppose Algorithm 4 with inputs (B, a, w) terminates, where B is a multi-rooted BDD. Then for each node $v \in B.V$, let f_v be the sub-function corresponding to the sub-BDD generated by regarding v as the root. The following holds:*

$$M_{BU}[v] = \mathit{COP}(P_a, f_v),$$

where $P_a(x_i) = \frac{1-a_i}{2}$ for all $i \in \{1,\cdots,n\}$.

Proof. *We prove by structural induction on multi-rooted BDD B.*

`Basis step`*: For the node $B.one$, $M_{BU}[v] = 1$. For the node $B.zero$, $M_{BU}[v] = 0$. Since the sub-functions given by $B.one$ and $B.zero$ are the constant functions with value 1 and 0 respectively, the statement holds.*

`Inductive step`*: For each non-terminal node v, after Algorithm 4 terminates, we have*

$$M_{BU}[v] = \frac{1-a_i}{2}\cdot M_{BU}[v.T] + \frac{1+a_i}{2}\cdot M_{BU}[v.F].$$

By inductive hypothesis, we have

$$
\begin{aligned}
M_{BU}[v] &= \frac{1-a_i}{2}\cdot \mathit{COP}(P_a, f_{v.T}) + \frac{1+a_i}{2}\cdot \mathit{COP}(P_a, f_{v.F}) \\
&= P(x_i)\cdot \mathit{COP}(P_a, f_{v.T}) + (1-P(x_i))\cdot \mathit{COP}(P_a, f_{v.F}) \\
&= \mathit{COP}(P_a, f_v)
\end{aligned}
$$

□

Now we are ready to prove Theorem 6. When Algorithm 4 terminates, we have

$$
\begin{aligned}
&\sum_{c\in C_f} (M_{BU}[B.entry(c)]\cdot w(c)) \\
&= \sum_{c\in C_f} \texttt{COP}(P_a, f_{B.entry(c)})\cdot w(c) \quad \text{(Lemma 4)} \\
&= \sum_{c\in C_f} \texttt{COP}(P_a, c)\cdot w(c) \\
&= F_{f,w}(a) \quad \text{(Corollary 1)}
\end{aligned}
$$

Complexity: The analysis is similar with the proof of Theorem 3. □

Proof of Theorem 7

Proof Sketch. *In the proof of Theorem 5 and 6 we actually reveal the meaning of the values of mappings M_{TD} and M_{BU} on a node v. Roughly speaking, $M_{TD}[v]$ has connection with the probability of a BDD node v being reached from the root ($\mathcal{P}(B, a, v)$), while $M_{BU}[v]$ contains information of the circuit-output probability of the sub-function defined by the sub-BDD with root v ($\texttt{COP}(P_a, f_v)$). Intuitively, Algorithm 5 can be interpreted as applying the differentiation operation on each node. We prove this theorem from writing down the definition of gradient and by gradually massaging the representation of gradient to connect it with M_{TD} and M_{BU}.*

By Corollary 1, for a real point $a \in [-1, 1]^n$, we have

$$F_{f,w}(a) = \sum_{c \in C_f} w(c) \cdot \texttt{COP}(P_a, c)$$

Thus the gradient of $F_{f,w}$ for x_i at a real point a, denoted by $\frac{\partial F_{f,w}}{\partial x_i}(a)$, can be computed by:

$$\frac{\partial F_{f,w}}{\partial x_i}(a) = \sum_{c \in C_f} w(c) \cdot \big(\texttt{COP}(P_{a_{i+}}, c) - \texttt{COP}(P_{a_{i-}}, c)\big),$$

where $a_{i+} = (a_1, \cdots, a_i = 1, \cdots, a_n)$ and $a_{i-} = (a_1, \cdots, a_i = -1, \cdots, a_n)$. Let $X_i = (x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_n)$, $X_{i,1} = (x_1, \cdots, x_{i-1})$ and $X_{i,2} = (x_{i+1}, \cdots, x_n)$.

By the definition of $\texttt{COP}$ we have

$$\begin{aligned}
&\texttt{COP}(P_{a_{i+}}, c)\\
=& \sum_{b \in \{-1,1\}^{n-1}} c(X_i = b, x_i = 1) \cdot \prod_{b_j = -1} P_a(x_j) \cdot \prod_{b_j = 1} (1 - P_a(x_j))\\
=& \sum_{b_1 \in \{-1,1\}^{i-1}} \Big(\prod_{\substack{b_{1j} = -1 \\ x_j \in X_{i,1}}} P_a(x_j) \cdot \prod_{\substack{b_{1j} = 1 \\ x_j \in X_{i,1}}} (1 - P_a(x_j)) \Big) \cdot \sum_{b_2 \in \{-1,1\}^{n-i}} \Big(\prod_{\substack{b_{2j} = -1 \\ x_{i+j} \in X_2}} P_a(x_{i+j}) \cdot \prod_{\substack{b_{2j} = 1 \\ x_{i+j} \in X_2}} (1 - P_a(x_{i+j})) \cdot c(b_1, 1, b_2) \Big)
\end{aligned}$$

Thus

$$\begin{aligned}
&\frac{\partial F_{f,w}}{\partial x_i}(a)\\
&= \sum_{c \in C_f} w(c) \cdot \big(\texttt{COP}(P_{a_{i+}}, c) - \texttt{COP}(P_{a_{i-}}, c)\big)\\
&= \sum_{c \in C_f} w(c) \cdot \sum_{b_1 \in \{-1,1\}^{i-1}} \Big(\prod_{\substack{b_{1j} = -1 \\ x_j \in X_1}} P_a(x_j) \cdot \prod_{\substack{b_{1j} = 1 \\ x_j \in X_1}} (1 - P_a(x_j)) \Big) \cdot \Big(\sum_{b_2 \in \{-1,1\}^{n-i}} \prod_{\substack{b_{2j} = -1 \\ x_{i+j} \in X_2}} P_a(x_{i+j}) \cdot \prod_{\substack{b_{2j} = 1 \\ x_{i+j} \in X_2}} (1 - P_a(x_{i+j})) \cdot\\
&\big(c(b_1, 1, b_2) - c(b_1, -1, b_2)\big)\Big)
\end{aligned}$$

Note that the following part of the right hand side of the equation above,

$$\Big(\sum_{b_2 \in \{-1,1\}^{n-i}} \prod_{\substack{b_{2j} = -1 \\ x_{i+j} \in X_2}} P_a(x_{i+j}) \cdot \prod_{\substack{b_{2j} = 1 \\ x_{i+j} \in X_2}} (1 - P_a(x_{i+j})) \cdot \big(c(b_1, 1, b_2) - c(b_1, -1, b_2)\big)\Big)$$

is non-zero if and only if the value of $b_1 = (b_{11}, \cdots, b_{1,i-1})$ leads to a non-terminal node of BDD B_c labeled by x_i, where B_c is the BDD corresponding with constraint c.

Thus we are able to simplify the representation of gradient by getting rid of b_1 and the equation above can be written as

$$\begin{aligned}
&\frac{\partial F_{f,w}}{\partial x_i}(a)\\
=& \sum_{c \in C_f} w(c) \sum_{\substack{v \in B_c \\ i_v = i}} \mathcal{P}(B_c, a, v) \cdot \Big(\sum_{b_2 \in \{-1,1\}^{n-i}} \prod_{\substack{b_{2j} = -1 \\ x_{i+j} \in X_2}} P_a(x_{i+j}) \cdot \prod_{\substack{b_{2j} = 1 \\ x_{i+j} \in X_2}} (1 - P_a(x_{i+j})) \cdot \big(f_v(x_i = 1, X_{i,2} = b_2) - f_v(x_i = -1, X_{i,2} = b_2)\big)\Big)
\end{aligned}$$

Note that the term

$$\sum_{b_2 \in \{-1,1\}^{n-i}} \prod_{\substack{b_{2j} = -1 \\ x_{i+j} \in X_2}} P_a(x_{i+j}) \cdot \prod_{\substack{b_{2j} = 1 \\ x_{i+j} \in X_2}} (1 - P_a(x_{i+j})) \cdot f_v(x_i = -1, X_{i,2} = b_2)$$

equals $\texttt{COP}(P_a, v.T)$.

Similarly,

$$\sum_{b_2 \in \{-1,1\}^{n-i}} \prod_{\substack{b_{2j}=-1 \\ x_{i+j} \in X_2}} P_a(x_{i+j}) \cdot \prod_{\substack{b_{2j}=1 \\ x_{i+j} \in X_2}} (1 - P_a(x_{i+j})) \cdot f_v(x_i = 1, X_{2,i} = b_2)$$

equals $\texttt{COP}(P_a, v.F)$.

Thus we further simplify the equation for computing gradient:

$$\frac{\partial F_{f,w}}{\partial x_i}(a) = \sum_{c_i \in C_f} \sum_{\substack{v \in B_i \\ i_v = i}} w(c)\mathcal{P}(B_i, a, v) \cdot \Big(\texttt{COP}(P_a, v.F) - \texttt{COP}(P_a, v.T)\Big)$$

Next we need to involve results given by Algorithm 5 and the multi-rooted BDD B instead of using separate BDDs for each constraints. Suppose the call to Algorithm 3 and 4 in line 3-4 of Algorithm 5 terminate. Then we have,

$$\begin{aligned}
&\sum_{c_i \in C_f} \sum_{\substack{v \in B_i \\ i_v = i}} w(c)\mathcal{P}(B_i, a, v) \\
=&\sum_{c_i \in C_f} \sum_{\substack{v \in B_i \\ i_v = i}} M^i_{TD}[v] \quad \text{(Lemma 2)} \\
=&\sum_{\substack{v \in B \\ i_v = i}} M_{TD}[v]
\end{aligned}$$

On the other hand, note that $\texttt{COP}(P_a, v)$ remains the same despite of whether v is in an individual BDD of a specific constraint or in the multi-rooted BDD. Thus,

$$\begin{aligned}
&\frac{\partial F_{f,w}}{\partial x_i}(a) \\
=&\sum_{\substack{v \in B \\ i_v = i}} M_{TD}[v] \cdot \Big(\texttt{COP}(P_a, v.F) - \texttt{COP}(P_a, v.T)\Big) \\
=&\sum_{\substack{v \in B \\ i_v = i}} M_{TD}[v] \cdot \big(M_{BU}[v.F] - M_{BU}[v.T]\big) \quad \text{(Lemma 4)}
\end{aligned}$$

The right hand side of the equation above is exactly what line 5-7 of Algorithm 5 computes.

Complexity: Algorithm 5 invokes Algorithm 3 and 4, with complexity both $O(S)$ by Theorem 5 and 6. Line 5-8 of Algorithm 5 can also be done in $O(S)$. In fact, line 5-8 of Algorithm 5 can be combined with Algorithm 4 once the top-down traverse is done so that the MRBDD is traversed twice instead of three times. Thus the complexity of Algorithm 5 is $O(S)$.

□

Generation of Benchmark 1

In the following we will give a detailed description of how we generate the random hybrid Boolean benchmarks used in Section 5 as **Benchmark 1**. All the instances generated are satisfiable. Let the number of variables be n, the CNF density (number of disjunctive clauses$/n$) be r_C, the XOR density (number of XOR constraints$/n$) be r_X, the density of pseudo Boolean/cardinality constraints be r_P, the threshold of pseudo-Boolean constraints and cardinality constraints be δ, let the density of variables in a cardinality/Pseudo-Boolean be r_V. We generate the following four families of benchmarks. All benchmarks can be found in the .zip file.

CNF-XORs. We follow the setting of generating formulas with random 3CNF-XOR and 5-CNF-XOR in (Dudek, Meel, and Vardi 2016), where the phase-transition of CNF-XOR formulas is studied. For each variable number $n \in \{50, 100, 150\}$, for $(r_C, r_X) \in \{(1, 0.2), (2, 0.2), (3, 0.2), (1, 0.4), (2, 0.4), (1, 0.6)\}$, generate 10 random instance with $r_C \cdot n$ 3-CNF constraints, $r_X \cdot n$ XOR instances with the probability of each variable appears in a constraint $\frac{1}{2}$. For each variable number $n \in \{50, 100, 150\}$, for $(r_C, r_X) \in \{(5, 0.2), (5, 0.4), (5, 0.6), (10, 0.2), (10, 0.4), (15, 0.2)\}$, generate 10 random instance with $r_C \cdot n$ 3-CNF constraints, $r_X \cdot n$ XOR constraints with the probability of each variable appears in a constraint $\frac{1}{2}$. Total number of instances in this family: 360.

XORs-1CARD. We follow the setting of generating random XOR with 1 cardinality constraint in (Pote, Joshi, and Meel 2019) where the phase-transition of random XOR + 1 cardinality constraint is studied. For each variable number

$n \in \{50, 100, 150\}$, $r_X \in \{0.2, 0.3, 0.4\}$, $\delta \in \{0.2, 0.3, 0.4\}$, we generate 10 random instance with $r_X \cdot n$ XOR constraints with the probability of each variable appears in a constraint $\frac{1}{2}$, as well as 1 global cardinality constraint $\sum_{i=1}^{n} x_i \leq \delta \cdot n$. Total number of instances in this family: 270.

CARDs. Instances in this family are composed of random cardinality constraints. For each variable number $n \in \{50, 100, 150\}$, $r_P \in \{0.5, 0.6, 0.7\}$, $r_V \in \{0.2, 0.3, 0.4, 0.5\}$, we generate 10 instances with each consisting $r_P \cdot n$ cardinality constraints. Each cardinality constraint contains $r_V \cdot n$ variables randomly sampled from all n variables. The direction is either "$\geq$" or "$\leq$" with probability $\frac{1}{2}$ and the right-hand-side threshold is set to be $\frac{r_V \cdot n}{2}$. Total number of instances in this family: 360.

PBs. Instances in this family are composed of random pseudo-Boolean constraints. We generate two types of PBs instances according to how the coefficients of a PB constraint are obtained.

1) For each appearance of a variable x_i in a constraint, we sample an integer from $\{1, \cdots, n\}$ uniformly at random to be the coefficient of x_i. The coefficients of the same variable can be different in different constraints. For each variable number $n \in \{50, 100, 150\}$, $r_P \in \{0.5, 0.6, 0.7\}$, $r_V \in \{0.2, 0.3, 0.4, 0.5\}$, we generate 10 instances with each consisting $r_P \cdot n$ PB constraints. Each PB constraint contains $r_V \cdot n$ variables sampled uniformly at random from all n variables. The direction is either "$\geq$" or "$\leq$" with probability $\frac{1}{2}$. The right-hand-side threshold is set to be half the sum of coefficients in the left-hand-side.

2) For each variable x_i, we sample an integer from $\{1, \cdots, n\}$ uniformly at random to be the coefficient of x_i. The coefficient of the same variable will be the same in different constraints. For each variable number $n \in \{50, 100, 150\}$, $r_P \in \{0.5, 0.6, 0.7\}$, $r_V \in \{0.2, 0.3, 0.4, 0.5\}$, we generate 10 instances with each consisting $r_P \cdot n$ PB constraints. Each PB constraint contains $r_V \cdot n$ variables sampled uniformly at random from all n variables. The direction is either "$\geq$" or "$\leq$" with probability $\frac{1}{2}$. The right-hand-side threshold is set to be half the sum of coefficients in the left-hand-side.

Total number of instances in this family: 720.