Virtual Pairwise Consistency in Cost Function Networks

Pierre Montalbano, David Allouche, Simon De Givry, George Katsirelos, Tomáš Werner

Virtual Pairwise Consistency in Cost Function Networks *

Pierre Montalbano[1], David Allouche[1], Simon de Givry[1], George Katsirelos[2], and Tomáš Werner[3]

[1] Université Fédérale de Toulouse, ANITI, INRAE, UR 875, 31326 Toulouse, France
{pierre.montalbano,david.allouche,simon.de-givry}@inrae.fr
[2] Université Fédérale de Toulouse, ANITI, INRAE, MIA Paris, AgroParisTech, 75231 Paris, France gkatsi@gmail.com
[3] Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic werner@fel.cvut.cz

Abstract. In constraint satisfaction, pairwise consistency (PWC) is a well-known local consistency improving generalized arc consistency in theory but not often in practice. A popular approach to enforcing PWC enforces arc consistency on the dual encoding of the problem, allowing to reuse existing AC algorithms. In this paper, we explore the benefit of this simple approach in the optimization context of cost function networks and soft local consistencies. Using a dual encoding, we obtain an equivalent binary cost function network where enforcing virtual arc consistency achieves virtual PWC on the original problem. We experimentally observed that adding extra non-binary cost functions before the dual encoding results in even stronger bounds. Such supplementary cost functions may be produced by bounded variable elimination or by adding ternary zero-cost functions. Experiments on (probabilistic) graphical models, from the UAI 2022 competition benchmark, show a clear improvement when using our approach inside a branch-and-bound solver compared to the state-of-the-art.

Keywords: dual encoding · non-binary cost function network · soft local consistency · branch-and-bound · graphical model · discrete optimization.

1 Introduction

Cost Function Networks (CFNs) can represent many combinatorial problems in a compact way as a sum of local functions over discrete variables. They have been used in bioinformatics [27,1], resource allocation [2,5], and elsewhere [9]. They can model probabilistic graphical models such as Bayesian networks and Markov random fields [8], which find many applications in artificial intelligence [32,17,28]. We focus here on the minimization task, *a.k.a.* Weighted Constraint

* This research was funded by the grants ANR-18-EURE-0021 and ANR-19-P3IA-0004. It receives support from the Genotoul (Toulouse) Bioinformatic platform.

Satisfaction Problem (WCSP), where exact methods mostly rely on a branch-and-bound procedure. Its efficiency depends on the compromise between the quality of its lower bound and the time to construct it. Several directions have been studied, inspired by Arc Consistency (AC) in CSP [6]. Stronger *soft* local consistencies were rarely considered, except in [7,22,13]. Pairwise Consistency (PWC) is known as the strongest consistency that can be enforced without introducing new cost functions when computing the lower bound [35]. It was never implemented nor tested in a branch-and-bound WCSP solver. In constraint programming, PWC was compared to generalized AC for solving non-binary CSPs given in extension [26,29,33,34]. These approaches rely on a dual encoding into a binary CSP. We explore a similar idea in the CFN framework.

2 Background

2.1 Weighted Constraint Satisfaction Problem

A *Cost Function Network (CFN)* is a quadruplet (V, D, S, f) where V is a set of variable indices (or *variables* in short), $D = (D_i)_{i \in V}$ is the list of *finite domains* for all the variables, S is a set of subsets of V, and $f = (f_A)_{A \in S}$ is the list of all the cost functions (defined below). A *value* of variable $i \in V$ is denoted by $x_i \in D_i$. By $D_A = \prod_{i \in A} D_i$ we denote the Cartesian product of the domains of variables $A \subseteq V$, and by $x = (x_i)_{i \in A} \in D_A$ an *assignment* to variables A. For $B \subseteq A \subseteq V$, $x|_B = (x_i)_{i \in B}$ denotes the projection of $x \in D_A$ to variables B. A *cost function* f_A is a function of a set of variables A taking non-negative values or possibly infinity (representing forbidden assignments), *i.e.,* it is a function $D_A \to \mathbb{R}_+ \cup \{\infty\}$, where $A \subseteq V$ is the *scope* of the function and $|A|$ its *arity*. A nullary cost function $f_\emptyset$ with an empty scope is just a constant. We assume that the network is normalized (with one cost function per scope), $\emptyset \in S$ ($f_\emptyset$ will be used as a problem lower bound) and $\{i\} \in S \ \forall i \in V$ (the network contains all unary functions). Given a CFN (V, D, S, f), the *Weighted Constraint Satisfaction Problem (WCSP)* is to find a complete non-forbidden assignment $x \in D_V$ minimizing the function $\sum_{A \in S} f_A(x|_A)$. This problem is NP-hard.

Example 1. Let $V = \{1, 2, 3, 4, 5\}$, $S = \{\emptyset, \{1\}, \{1, 2, 3\}, \{1, 4\}, \{2\}, \{2, 3, 4\}, \{2, 3, 5\}, \{3\}, \{4\}, \{5\}\}$, $D_2 = D_3 = D_5 = \{a, b\}$, and $D_1 = D_4 = \{a, b, c\}$. The WCSP aims to minimize the objective function $f_\emptyset + f_1(x_1) + f_{123}(x_1, x_2, x_3) + f_{14}(x_1, x_4) + f_2(x_2) + f_{234}(x_2, x_3, x_4) + f_{235}(x_2, x_3, x_5) + f_3(x_3) + f_4(x_4) + f_5(x_5)$ (where we abbreviated $f_{\{1,2,3\}}$ by f_{123}, etc.) over all assignments $(x_1, x_2, x_3, x_4, x_5) \in D_V$.

2.2 Constraint Satisfaction Problem and Local Consistencies

If all cost functions in a CFN take only values 0 or ∞, the cost functions are called *constraints* and the WCSP reduces to the *Constraint Satisfaction Problem (CSP)*. In this case, we denote the cost functions by r_A rather than f_A, so the *Constraint Network (CN)* is defined by (V, D, S, r). The values 0 and ∞ act as the logical values *true* and *false*, respectively. For $u, v \in \{0, \infty\}$, we will denote

logical conjuction by $u \wedge v = u + v$ and the disjunction by $u \vee v = \min\{u, v\}$. As in CFN, we assume a CN contains all unary constraints, i.e., $\{i\} \in S \; \forall i \in V$.

For any $B \subseteq A \subseteq V$, we define the *projection* of a constraint r_A: $D_A \to \{0, \infty\}$ onto variables B to be the constraint $r_A|_B$: $D_B \to \{0, \infty\}$ given by

$$r_A|_B(x) = \bigvee_{x' \in D_A:\, x'|_B = x} r_A(x') \qquad \forall x \in D_B. \tag{1}$$

We say that a pair of constraints $\{r_A, r_B\}$ is *Pairwise Consistent (PWC)* if they admit the same set of assignments to their shared variables, i.e.,

$$r_A|_{A \cap B} = r_B|_{A \cap B} \tag{2}$$

where '=' denotes here equality of functions. A CN is PWC if all possible pairs of its constraints are PWC.[4] If we restrict PWC to pairs of constraints where one constraint is unary, we get (generalized) arc consistency (GAC, AC for binary CNs). PWC or GAC can be enforced on a CN P by iteratively forbidding assignments that violate (2). The minimal set of changes required to do this is unique and the resulting CN is called the PWC (or GAC) *closure* of P.

We say that a local consistency ψ' is *not weaker* than a local consistency ψ if for every CN instance for which the ψ-consistency closure is empty, the ψ'-consistency closure is also empty. We say that ψ and ψ' are *equally strong* if ψ' is not weaker than ψ and *vice versa*. We say that ψ' is *strictly* stronger than ψ if ψ' is not weaker than ψ but they are not equally strong. It can be shown that: (i) for binary CNs, AC is equally strong as PWC; (ii) for non-binary CNs, PWC is strictly stronger than GAC.

The PWC relation of constraints is clearly reflexive and symmetric. It is in general not transitive but it satisfies the following weaker condition:

Theorem 1. *[16] Let $C_1, \ldots, C_n \in S$ be such that for every $i = 1, \ldots, n$, we have $C_1 \cap C_n \subseteq C_i$. Let for every $i = 1, \ldots, n-1$, constraint r_{C_i} be PWC with $r_{C_{i+1}}$. Then r_{C_1} is PWC with r_{C_n}.*

Thus, enforcing PWC for some constraint pairs implies that the PWC condition holds also for some other pairs, which can simplify algorithms [16,30].

2.3 Soft Local Consistencies

To solve a WCSP to optimality, most methods rely on a branch-and-bound algorithm. At each node, the solver computes a bound using either static memory-intensive bounds [11] or memory-light ones [6] better suited to dynamic variable orderings. We focus on the latter, called *Soft Arc Consistencies (SAC)*, because they reason on each non-unary cost function one by one, in a generalization of

[4] This corresponds to *full* PWC because unary constraints may appear in these pairs.

propagation in CSPs. In particular, *Virtual Arc Consistency (VAC)* is characterized by AC of a CN derived from the CFN. To any CFN $P = (V, D, S, f)$ we associate the CN $\text{Bool}(P) = (V, D, S, r)$ where $r_\emptyset = 0$ and

$$r_A(x) = \begin{cases} 0 & \text{if } f_A(x) = 0 \\ \infty & \text{if } f_A(x) > 0 \end{cases} \qquad \forall A \in S \setminus \{\emptyset\},\ x \in D_A. \tag{3}$$

Definition 1. *[6] A CFN P is VAC if the GAC closure of* $\text{Bool}(P)$ *is non-empty.*

Algorithms enforcing SACs apply a sequence of *Equivalence-Preserving Transformations (EPTs)* to the CFN. An EPT moves finite costs between two cost functions. That is, for some $A, B \in S$ we add a function φ_{AB}: $D_{A\cap B} \to \mathbb{R}$ with scope $A \cap B$ to cost function f_A and subtract it from function f_B:

$$f_A(x) := f_A(x) + \varphi_{AB}(x|_{A\cap B}) \qquad \forall x \in D_A, \tag{4a}$$
$$f_B(x) := f_B(x) - \varphi_{AB}(x|_{A\cap B}) \qquad \forall x \in D_B. \tag{4b}$$

This operation can be seen as moving a set of costs (stored as the values of the φ_{AB}) from f_B to f_A. The values of φ_{AB} cannot be arbitrary because we require the resulting cost functions to have non-negative values. EPTs preserve the WCSP objective function because the terms φ_{AB} and $-\varphi_{AB}$ cancel out in the sum $\sum_{A\in S} f_A(x|_A)$. CFNs (V, D, S, f) and (V, D, S, f') are *equivalent* if one can be obtained from the other by a sequence of EPTs. SAC algorithms aim to derive an equivalent CFN where $f'_\emptyset > f_\emptyset$.

2.4 Dual Encoding of a Cost Function Network

An encoding into a binary CFN is a way to get better bounds. The *dual encoding* of a CFN $P = (V, D, S, f)$ is a CFN $\text{Dual}(P) = (S \setminus \{\emptyset\}, \bar{D}, \bar{S}, \bar{f})$ where:

- The variables of the dual problem are the scopes $S \setminus \{\emptyset\}$ of P.
- The domain of variable $A \in S \setminus \{\emptyset\}$ of the dual problem is $\bar{D}_A = D_A$.
- The scopes are $\bar{S} = \{\emptyset\} \cup \{\,\{A\} \mid A \in S\,\} \cup \{\,\{A, B\} \mid A, B \in S,\ A \cap B \neq \emptyset\,\}$.
- The dual nullary cost function is unchanged: $\bar{f}_\emptyset = f_\emptyset$.
- The dual unary cost function with scope $\{A\} \in \bar{S}$ is the function $\bar{f}_A = f_A$.
- The dual binary cost function with scope $\{A, B\} \in \bar{S}$ is the *channeling constraint* $\bar{f}_{AB}$: $D_A \times D_B \to \{0, \infty\}$ with values:

$$\bar{f}_{AB}(y, y') = \begin{cases} 0 & \text{if } y_i = y'_i\ \forall i \in A \cap B \\ \infty & \text{otherwise} \end{cases} \qquad \forall y = (y_i)_{i\in A} \in D_A,\ y' = (y'_i)_{i\in B} \in D_B.$$

Example 2. Let P be the CFN described in Example 1, represented by the hypergraph in Fig. 1(a). Then, $\text{Dual}(P)$ has 9 dual variables, $y_1, y_{123}, y_{14}, \ldots, y_5$, and 16 binary channeling constraints, as shown by the constraint graph in Fig. 1(b). Using Theorem 1, a minimal dual graph can be produced with only 9 binary constraints (Fig. 1(c)).

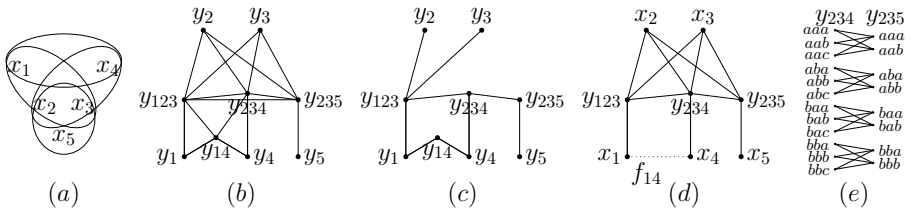

Fig. 1. (a) Hypergraph of a CFN, (b) its dual graph, (c) a minimal dual graph, (d) the partial dual graph used in the experiments, (e) a binary channeling constraint created by the dual encoding (an edge depicts a 0-cost assignment).

(a)

f_{123}	x_1	x_2	x_3	Cost
	a	a	a	0
	a	a	b	1
	b	a	a	1
	b	a	b	1
	c	a	a	1
	c	a	b	1

f_{234}	x_2	x_3	x_4	Cost
	a	a	a	1
	a	a	b	1
	a	a	c	1
f_1			x_1	
			b	2
			c	2

f_{14}	x_1	x_4	Cost
	a	a	2
	a	b	2
	a	c	2
f_2		x_2	
		a	0
		b	2

(b)

y_{123}	Cost	y_{234}	Cost	y_{14}	Cost
aaa	0	aaa	1	aa	2
aab	1	aab	1	ab	2
baa	1	aac	1	ac	2
bab	1	y_1		y_2	
caa	1	b	2	a	0
cab	1	c	2	b	1

Table 1. (a) Original CFN. (b) dual unary cost functions (missing tuples have 0 cost).

3 Virtual Pairwise Consistency

Following the idea of VAC, we introduce *Virtual Pairwise Consistency (VPWC)*, a stronger soft local consistency than VAC.

Definition 2. *A CFN P is VPWC if the PWC closure of* Bool(P) *is non-empty.*

Combining Definition 2 and previous results [16], we get that enforcing VPWC is possible using existing algorithms.

Theorem 2. *Let P be a CFN. P is VPWC if and only if* Dual(P) *is VAC.*

Proof. It is known that a CN has a non-empty PWC closure if and only if its dual has a non-empty AC closure [16]. Clearly, for any CFN P we have Dual(Bool(P)) = Bool(Dual(P)). Therefore, P is VPWC iff Dual(Bool(P)) = Bool(Dual(P)) has a non-empty AC closure, which means Dual(P) is VAC. □

Example 3. Following Ex.2, we give the costs for each cost function in Table 1(a). VAC on this problem derives a lower bound of 2, since $x_1 = a$ is not consistent with r_{14}. VAC on the dual (Table 1(b)) derives a lower bound of 3, because (a) all values in y_{14} compatible with $y_1 = a$ (*i.e.,* aa, ab, ac) have cost 2, and (b) all values compatible with $y_{123} = aaa$ in y_{234} (*i.e.,* aaa, aab, aac) have a cost of 1, therefore they do not support $y_2 = a$, making it inconsistent in y_{123}. This leads to a lower bound of 3.

The dual can help derive better lower bounds, as we show in the next section, but introduces a possibly large number of variables with large domains which may slow down search. We propose to first dualize the problem and get a first strong lower bound, then return to the primal. The following shows that this is always possible without introducing higher order cost functions[5].

[5] This is unsurprising because the strongest bound that can be derived using EPTs is obtained using a linear program which includes pairwise consistency constraints [35].

Theorem 3. *Let P be a CFN and let Q be a CFN equivalent to* Dual(P)*. Then there exists a CFN Q' equivalent to Q such that all binary constraints of Q' are hard and Q' has the same lower bound as Q.*

Proof (Sketch). The main observation is that every dual binary cost function (the channeling constraint) r_{ij} has a block structure (see Fig. 1(e)): there exists a partition $H_i = \{s_1, ..., s_m\}$ of the domain D_i and a partition $H_j = \{s'_1, ..., s'_m\}$ of D_j such that for each $x_i \in s_k$ and $x_j \in s'_l$ we have $r_{ij}(x_i, x_j) = 0$ whenever $k = l$ and $r_{ij}(x_i, x_j) = \infty$ whenever $k \neq l$. This implies that every EPT that moves cost into r_{ij} can be matched with another EPT that moves cost out of it without affecting the lower bound. □

We can now summarize the base version of our approach. Given a CFN P, we apply EPTs to its dual encoding Dual(P) (using a VAC algorithm) to obtain a CFN Q with an increased lower bound. Theorem 3 lets us obtain from Q another CFN Q' in which all channeling cost functions are constraints. We can thus undo the dual encoding, i.e., obtain a CFN P', equivalent to P, such that $Q' = \text{Dual}(P')$. If Q was VAC then, by Theorem 2, P' is VPWC.

4 Experimental Results on UAI 2022 Competition

We won a recent competition on probabilistic graphical models. [6] We present results on a set of 120 tuning instances where 63 have maximum arity of 3.

We evaluate three solvers: daoopt (version from UAI 2012 competition with 1-hour settings as given in [23]), cplex (version 20.1.0.0, forcing completeness with zero absolute and relative gaps, translating CFN to 0-1 LP by the tuple encoding [15]), and toulbar2 (version 1.2.0) using two state-of-the-art methods, Variable Neighborhood Search (VNS) [24] winner of UAI 2014 competition, [7] and Hybrid Best-First Search with VAC in preprocessing (VACpre-HBFS), including VAC integrality heuristics [31]. [8] We implemented VPWC in the latest version of toulbar2. It is either enforced in preprocessing (and then converted back to the primal, see Theorem 3) (VPWCpre-HBFS) or maintained during search (HBFS-VPWC). EDAC is always enforced [19,27], providing a default value ordering heuristic when no solution is found for solution-based heuristics [12]. The branching heuristic is *dom/wdeg* [4] combined with *last conflict* [20].

We use a slightly different binary encoding, a hybrid between the dual and *hidden variable encoding* [25].[9] We keep the original variables and the original binary cost functions unchanged, and only dualize the original non-binary cost functions. We add channeling constraints between those pairs of dual variables that are not redundant by Theorem 1 and with intersecting scopes strictly greater

[6] `https://uaicompetition.github.io/uci-2022`, see MPE and MMAP entries.

[7] `http://auai.org/uai2014/competition.shtml`, `http://miat.inrae.fr/toulbar2`

[8] Options *-A -P=1000 -T=1000 -vacint -vacthr -rasps -raspsini* in toulbar2-vacint.

[9] Called *double encoding* in [26], it allows more flexibility to enforce various levels of consistency from GAC to PWC depending on the selected channeling constraints.

instance	(n,d,e,a)	(n',e',a')	(n'',d'',e'')	VACpre-HBFS time *(gap)*	VPWCpre-HBFS time *(gap)*	HBFS-VPWC time *(gap)*
Grids21	(1600,2,4800,2)	(799,2810,4)	(1628,16,4675)	- *(42.4%)*	- *(3%)*	1216.83
Promedas12	(1766,2,1766,3)	(826,1884,4)	(1373,16,2223)	5.17	6.34	7.7
ProteinFold11	(400,2,1160,2)	(190,604,4)	(381,16,1005)	- *(16.1%)*	8.48	12.43
wcsp12	(311,4,5732,3)	(305,5887,3)	(12708,64,70959)	- *(49.9%)*	- *(19.3%)*	- *(54.8%)*

Table 2. UAI 2022 detailed results on a selection of four instances for HBFS methods. '-' means the instance is unsolved in 1h. (in parentheses, remaining optimality gap).

than 1. We also add channeling constraints between dual and primal variables.[10] Note that Theorem 2 and 3 remain valid. Moreover, we apply this encoding only partially, indeed for high-arity constraints, a full dual encoding might mean prohibitive amount of memory to store the dual domains. Hence, only non-binary cost functions of arity less than 10 and fewer than 2^{15} non-forbidden tuples are dualized. Those remaining are lazily propagated by VAC/EDAC when they have less than three unassigned variables in their scope. The memory used by each channeling constraint between a pair of dual variables is restricted to at most 1MB (arbitrarily chosen). Larger channeling constraints are ignored.

Additional preprocessing is performed beforehand for all the HBFS methods in order to find better bounds. An initial upper bound is found by local search [21,3] and VAC-based heuristics [31]. To reduce the problem size and improve lower bounds, we apply bounded variable elimination with a min-fill ordering [10,18,14][11] and add ternary zero-cost functions on the *most-preferred triangles* (total memory space of extra ternary functions limited to 1MB).[12] It results in at most 6-ary (resp. zero-cost ternary) cost functions for 84 (resp. 81) instances, making our encoding applicable to 85 instances rather than 63. Finding a (quasi-)minimal dual graph (see Theorem 1) yielded 700.3 channeling constraints on average, a 4.5% savings compared to the complete dual graph.

The experiments were run on a single core of Intel Xeon E5-2683 2.1GHz processors with 1-hour CPU-time and 8GB memory limit. toulbar2 was able to solve optimally 86 instances using VACpre-HBFS or VNS. daoopt solved 92 instances and cplex 95 instances. Using our partial dual encoding with VPWC applied in preprocessing, VPWCpre-HBFS solved 95 instances, and when applied during search, HBFS-VPWC solved 99 instances, 15% above VACpre-HBFS, being the best exact method for this benchmark.

Table 2 shows for a selection of UAI 2022 instances their size in terms of number of variables n, maximum domain size d, number of cost functions e, maximum arity a of the original problem, after preprocessing it with bounded

[10] The resulting non-minimal graph for Example 2 is shown in Fig. 1(d).

[11] It is done only if the median degree in the original problem is less than 8, eliminating variables with a current degree less than or equal to the original median degree.

[12] With additional options *-i -pils -p=-8 -O=-3 -t=1*. A triangle is defined by three variables involved in three binary cost functions. The score of a triangle is given by the average cost in the three functions. Triangles with the largest score are selected first. This approach allows to simulate soft path inverse consistency [22].

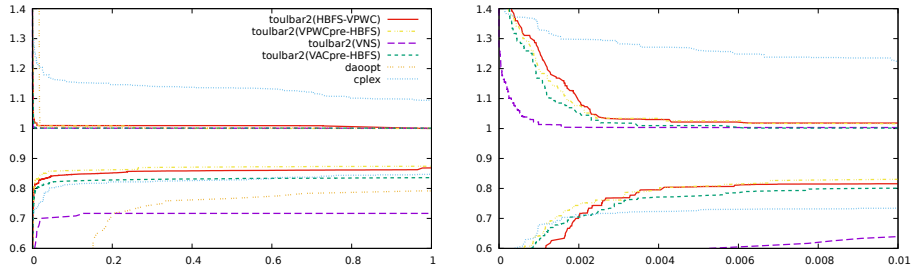

Fig. 2. Normalized lower and upper bounds (y-axis) as time passes (x-axis in hour, zoomed on the right fig.) for cplex, daoopt, and toulbar2 on UAI 2022 tuning benchmark.

variable elimination and adding triangles (n', d', e', a' with $d' = d$), and after applying our partial dual encoding (n'', d'', e'', a'' with $a'' = 2$). It gives also the CPU-time in seconds to solve an instance using HBFS methods or the remaining optimality gap if unsolved after 1 hour. On Grids21, only HBFS-VPWC solves the instance. Notice the large improvement on the optimality gap by VPWCpre compared to VACpre. On Promedas12 and ProteinFolding11, VPWCpre-HBFS develops 13 and 32250 nodes, respectively, and takes about the same time as HBFS-VPWC which develops 4 and 992 nodes, respectively. For wcsp12, the size of the encoding slows down the search too much, suggesting harder limits for our partial dual encoding.

We report in Fig. 2 the average normalized lower and upper bounds as time passes (computed as in [31]). Here VNS provides the best upper bounds in limited time whereas HBFS-VPWC is slightly slower than VPWCpre-HBFS, VACpre-HBFS, and VNS, but still faster than daoopt and cplex. Both VPWCpre-HBFS and HBFS-VPWC offer the best average lower bounds in less than 1 hour. HBFS-VPWC found 117 best solutions, VPWCpre-HBFS 112, VACpre-HBFS 106, VNS 105, daoopt 99, and cplex 95. VNS found 2 single-best solutions (wcsp11, wcsp12). For the competition, we combined VNS and HBFS-VPWC sequentially.[13]

5 Conclusion

We have defined virtual pairwise consistency and shown how it can efficiently be used in preprocessing or during search by applying the existing VAC algorithm to a dual encoding of the problem. In the future we will explore the benefit of other binary encodings [34] and adapt the VAC algorithm to the specific constraints of the encoding as it is done in CSPs [29,33]. Finding good heuristics to exploit a partial dual encoding in conjunction with bounded variable elimination and zero-cost function addition is also an interesting question.

[13] See toulbar2-ipr results on the UAI 2022 Tuning Leader Board. Multiple runs of VNS with increasing floating-point precision were done with a total amount of time of $\frac{1}{2}$h. The remaining time is allocated to HBFS-VPWC. Each search procedure gives its best solution found to the next search procedure. On UAI 2022 tuning instances, this approach found 119 best solutions, ranking first among our 7 tested methods.

References

1. Allouche, D., Davies, J., de Givry, S., Katsirelos, G., Schiex, T., Traoré, S., André, I., Barbe, S., Prestwich, S., O'Sullivan, B.: Computational protein design as an optimization problem. Artificial Intelligence **212**, 59–79 (2014)
2. Bensana, E., Lemaître, M., Verfaillie, G.: Earth observation satellite management. Constraints **4**(3), 293–299 (1999)
3. Beuvin, F., de Givry, S., Schiex, T., Verel, S., Simoncini, D.: Iterated local search with partition crossover for computational protein design. Proteins: Structure, Function, and Bioinformatics (2021)
4. Boussemart, F., Hemery, F., Lecoutre, C., Sais, L.: Boosting systematic search by weighting constraints. In: ECAI. vol. 16, p. 146 (2004)
5. Cabon, B., de Givry, S., Lobjois, L., Schiex, T., Warners, J.: Radio link frequency assignment. Constraints Journal **4**, 79–89 (1999)
6. Cooper, M., de Givry, S., Sanchez, M., Schiex, T., Zytnicki, M., Werner, T.: Soft arc consistency revisited. Artificial Intelligence **174**(7–8), 449–478 (2010)
7. Cooper, M.C.: High-order consistency in Valued Constraint Satisfaction. Constraints **10**, 283–305 (2005)
8. Cooper, M.C., de Givry, S., Schiex, T.: Graphical models: Queries, complexity, algorithms (tutorial). In: 37th International Symposium on Theoretical Aspects of Computer Science (STACS-20). LIPIcs, vol. 154, pp. 4:1–4:22. Montpellier, France (2020)
9. Cooper, M.C., de Givry, S., Schiex, T.: Valued Constraint Satisfaction Problems, pp. 185–207. Springer International Publishing (2020)
10. Dechter, R.: Bucket elimination: A unifying framework for reasoning. Artificial Intelligence **113**(1–2), 41–85 (1999)
11. Dechter, R., Rish, I.: Mini-buckets: A general scheme for bounded inference. Journal of the ACM (JACM) **50**(2), 107–153 (2003)
12. Demirovic, E., Chu, G., Stuckey, P.J.: Solution-based phase saving for CP: A value-selection heuristic to simulate local search behavior in complete solvers. In: Proc. of CP-18. pp. 99–108. Lille, France (2018)
13. Dlask, T., Werner, T., de Givry, S.: Bounds on weighted CSPs using constraint propagation and super-reparametrizations. In: Proc. of CP-21. Montpellier, France (2021)
14. Favier, A., de Givry, S., Legarra, A., Schiex, T.: Pairwise decomposition for combinatorial optimization in graphical models. In: Proc. of IJCAI-11. Barcelona, Spain (2011), video demonstration at http://www.inra.fr/mia/T/degivry/Favier11.mov
15. Hurley, B., O'Sullivan, B., Allouche, D., Katsirelos, G., Schiex, T., Zytnicki, M., de Givry, S.: Multi-Language Evaluation of Exact Solvers in Graphical Model Discrete Optimization. Constraints **21**(3), 413–434 (2016)
16. Janssen, P., Jégou, P., Nouguier, B., Vilarem, M.C.: A filtering process for general constraint-satisfaction problems: achieving pairwise-consistency using an associated binary representation. In: IEEE International Workshop on Tools for Artificial Intelligence. pp. 420–421. IEEE Computer Society (1989)
17. Kappes, J.H., Andres, B., Hamprecht, F.A., Schnörr, C., Nowozin, S., Batra, D., Kim, S., Kausler, B.X., Kröger, T., Lellmann, J., Komodakis, N., Savchynskyy, B., Rother, C.: A comparative study of modern inference techniques for structured discrete energy minimization problems. Intl. J. of Computer Vision **115**(2), 155–184 (2015)

18. Larrosa, J.: Boosting search with variable elimination. In: Principles and Practice of Constraint Programming - CP 2000. LNCS, vol. 1894, pp. 291–305. Singapore (Sep 2000)
19. Larrosa, J., Heras, F.: Resolution in Max-SAT and its relation to local consistency in weighted CSPs. In: Proc. of IJCAI'05. pp. 193–198. Edinburgh, Scotland (2005)
20. Lecoutre, C., Saïs, L., Tabary, S., Vidal, V.: Reasoning from last conflict(s) in constraint programming. Artificial Intelligence **173**, 1592,1614 (2009)
21. Neveu, B., Trombettoni, G., Glover, F.: ID Walk: A Candidate List Strategy with a Simple Diversification Device. In: Proc. of CP. pp. 423–437 (2004)
22. Nguyen, H., Bessiere, C., de Givry, S., Schiex, T.: Triangle-based Consistencies for Cost Function Networks. Constraints **22**(2), 230–264 (2017)
23. Otten, L., Ihler, A., Kask, K., Dechter, R.: Winning the pascal 2011 map challenge with enhanced AND/OR branch-and-bound. In: DISCML'12 Workshop, at NIPS'12. Lake Tahoe, NV (2012)
24. Ouali, A., Allouche, D., de Givry, S., Loudni, S., Lebbah, Y., Loukil, L., Boizumault, P.: Variable neighborhood search for graphical model energy minimization. Artificial Intelligence **278**(103194), 22p. (2020)
25. Rossi, F., Petrie, C.J., Dhar, V.: On the equivalence of constraint satisfaction problems. In: ECAI. vol. 90, pp. 550–556 (1990)
26. Samaras, N., Stergiou, K.: Binary encodings of non-binary constraint satisfaction problems: Algorithms and experimental results. Journal of Artificial Intelligence Research **24**, 641–684 (2005)
27. Sánchez, M., de Givry, S., Schiex, T.: Mendelian error detection in complex pedigrees using weighted constraint satisfaction techniques. Constraints **13**(1), 130–154 (2008)
28. Savchynskyy, B.: Discrete graphical models – an optimization perspective. Foundations and Trends in Computer Graphics and Vision **11**(3-4), 160–429 (2019)
29. Schneider, A., Choueiry, B.Y.: PW-CT: Extending compact-table to enforce pairwise consistency on table constraints. In: International Conference on Principles and Practice of Constraint Programming. pp. 345–361. Springer (2018)
30. Schneider, A., Choueiry, B.Y.: PW-CT: Extending compact-table to enforce pairwise consistency on table constraints. In: International Conference on Principles and Practice of Constraint Programming. pp. 345–361. Springer (2018)
31. Trösser, F., de Givry, S., Katsirelos, G.: Relaxation-aware heuristics for exact optimization in graphical models. In: Proc. of CP-AI-OR'2020. pp. 475–491. Vienna, Austria (2020)
32. Wainwright, M.J., Jordan, M.I., et al.: Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning **1**(1–2), 1–305 (2008)
33. Wang, R., Yap, R.H.: Arc consistency revisited. In: International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research. pp. 599–615. Springer (2019)
34. Wang, R., Yap, R.H.: Bipartite encoding: a new binary encoding for solving non-binary CSPs. In: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence. pp. 1184–1191 (2021)
35. Werner, T.: Revisiting the linear programming relaxation approach to Gibbs energy minimization and weighted constraint satisfaction. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(8), 1474–1488 (August 2010)