Finding Resilient Solutions for Dynamic Multi-Objective Constraint Optimization Problems

Maxime Clement[1], Tenda Okimoto[2,3], Nicolas Schwind[3,4] and Katsumi Inoue[4,1]
[1]*Department of Informatics, The Graduate University for Advanced Studies, Tokyo, Japan*
[2]*Faculty of Maritime Sciences, Kobe University, Kobe, Japan*
[3]*Transdisciplinary Research Integration Center, Tokyo, Japan*
[4]*National Institute of Informatics, Tokyo, Japan*

Keywords: Systems Resilience, Dynamic Multi-Objective COP, Resistance, Functionality, Reactive Approach.

Abstract: *Systems Resilience* is a large-scale multi-disciplinary research that aims to identify general principles underlying the resilience of real world complex systems. Many conceptual frameworks have been proposed and discussed in the literature since Holling's seminal paper (1973). Schwind et al. (2013) recently adopted a computational point of view of Systems Resilience, and modeled a resilient system as a dynamic constraint optimization problem. However, many real world optimization problems involve multiple criteria that should be considered separately and optimized simultaneously. Also, it is important to provide an algorithm that can evaluate the resilience of a dynamic system. In this paper, a framework for *Dynamic Multi-Objective Constraint Optimization Problem* (DMO-COP) is introduced and two solution criteria for solving this problem are provided, namely *resistance* and *functionality*, which are properties of interest underlying the *resilience* for DMO-COPs. Also, as an initial step toward developing an efficient algorithm for finding resilient solutions of a DMO-COP, an algorithm called *Algorithm for Systems Resilience* (*ASR*), which computes every resistant and functional solution for DMO-COPs, is presented and evaluated with different types of dynamical changes.

1 INTRODUCTION

Many researchers of different fields have recognized the importance of a new research discipline concerning the *resilience* of real world complex systems (Holling, 1973; Bruneau, 2003; Walker et al., 2004; Longstaff et al., 2010). The concept of resilience has appeared in various disciplines, e.g., environmental science, materials science and sociology. The goal of resilience research is to provide a set of general principles for building resilient systems in various domains, such that the systems are resistant from large-scale perturbations caused by unexpected events and changes, and keep their functionality in the long run. Holling (1973) first introduced the concept of resilience as an important characteristic of a well-behaved ecological system, and defined it as the capacity of an ecosystem to respond to a perturbation or disturbance by resisting damage. He adopted a verbal, qualitative definition of ecological resilience, rather than a mathematical, quantitative one. "Resilience determines the persistence of relationships within a system and is a measure of the ability of these systems to absorb changes of state variables, driving variables, and parameters, and still persist." (Holling, 1973, page 17). Schwind et al.(2013) adopted a computational point of view of *Systems Resilience*, and modeled a resilient system as a dynamic constraint-based model (called SR-model), i.e., dynamic constraint optimization problem. They captured the notion of resilience using several factors, e.g., resistance, recoverability, functionality and stabilizability.

Capturing and evaluating the resilience of realistic dynamic systems often requires to (i) consider several objectives to optimize simultaneously from the point of view of the resilience factors, and (ii) develop an algorithm for solving this problem. This is the main purpose of this paper.

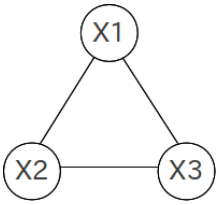
A *Multi-Objective Constraint Optimization Problem* (MO-COP) (Junker, 2006; Marinescu, 2010; Rollon and Larrosa, 2006) is the extension of a mono-objective COP (Dechter, 2003; Schiex et al., 1995). Solving a COP consists in finding an assignment of values to variables so that the sum of the resulting costs is minimized. A wide variety of Artificial Intelligence problems can be formalized as COPs,

Clement M., Okimoto T., Schwind N. and Inoue K..
Finding Resilient Solutions for Dynamic Multi-Objective Constraint Optimization Problems.
DOI: 10.5220/0005276305090516
In *Proceedings of the International Conference on Agents and Artificial Intelligence* (ICAART-2015), pages 509-516
ISBN: 978-989-758-074-1

e.g., resource allocation problem (Cabon et al., 1999), scheduling (Verfaillie et al., 1996) and combinatorial auctions (Sandholm, 1999). In an MO-COP, generally, since trade-offs exist among objectives, there does not exist an ideal assignment, which minimizes all objectives simultaneously. Thus, the "optimal" solution of an MO-COP is characterized using the concept of *Pareto optimality*. An assignment is a *Pareto optimal solution* if there does not exist another assignment that weakly improves all of the objectives. Solving an MO-COP is to find the *Pareto front* which is a set of cost vectors obtained by all Pareto optimal solutions. Most works on MO-COPs consist in developing efficient algorithms for solving static problems (Marinescu, 2010; Perny and Spanjaard, 2008; Rollon and Larrosa, 2006; Rollon and Larrosa, 2007). However, due to the dynamic nature of our environment, many real-world problems change through time.

In this paper, a framework for *Dynamic Multi-Objective Constraint Optimization Problem* (DMO-COP) is introduced, which is the extension of MO-COP and dynamic COP. Also, two solution criteria for solving this problem are provided, namely *resistance* and *functionality*, which are properties of interest underlying the resilience for DMO-COPs. Our model is defined by a sequence of MO-COPs representing the changes within a system that is subject to external fluctuations. The resistance is the ability to maintain some underlining costs under a certain threshold, such that the system satisfies certain hard constraint and does not suffer from irreversible damages. The functionality is the ability to maintain a guaranteed global quality for the configuration trajectory in a sequence. These two properties are central in the characterization of "robust" solution trajectories, which keep a certain quality level and "absorb" external fluctuations without suffering degradation. Indeed, these notions are consistent with the initial formulation of resilience from (Holling, 1973). An algorithm called *Algorithm for Systems Resilience* (*ASR*) for solving a DMO-COP is presented. This algorithm is based on the branch and bound search, which is widely used for COP and MO-COP algorithms, and it finds all resistant and functional solutions for DMO-COP. In the experiments, the performances of *ASR* are evaluated with different types of dynamical changes.

We believe that the computational design of resilient systems is a promising area of research, relevant for many applications like sensor networks. A sensor network is a resource allocation problem that can be formalized as a COP (Cabon et al., 1999). For example, consider a sensor network in a territory, where each sensor can sense a certain area in this territory. When we consider this problem with multiple criteria, e.g., data management, quality of observation data and electrical consumption, it can be formalized as an MO-COP (Okimoto et al., 2014). Additionally, when we consider some accidents, e.g., sensing error, breakdown and electricity failure, it can be represented by the dynamical change of constraint costs.



x_1	x_2	*cost*
a	a	5
a	b	7
b	a	10
b	b	12

x_2	x_3	*cost*
a	a	0
a	b	2
b	a	0
b	b	2

x_1	x_3	*cost*
a	a	1
a	b	1
b	a	0
b	b	3

Figure 1: Example of mono-objective COP.

The rest of the paper is organized as follows. In the next section, the formalizations of COP and MO-COP are briefly introduced. The following section presents our framework for DMO-COP and the computation of resistant and functional solutions. Also, an algorithm called *ASR* is presented. Afterwards, some empirical results are provided. Just before the concluding section, some related works are discussed.

2 PRELIMINARIES

2.1 COP

A *Constraint Optimization Problem* (COP) (Dechter, 2003; Schiex et al., 1995) consists in finding an assignment of values to variables so that the sum of the resulting costs is minimized. A COP is defined by a set of variables X, a set of constraint relations C, and a set of cost functions F. A variable x_i takes its value from a finite, discrete domain D_i. A constraint relation (i, j) means there exists a constraint relation between x_i and x_j.[1] For x_i and x_j, which have a constraint relation, the cost for an assignment $\{(x_i, d_i), (x_j, d_j)\}$ is defined by a cost function $f_{i,j} : D_i \times D_j \rightarrow \mathbb{R}^+$. For a value assignment to all variables A, let us denote $R(A) = \sum_{(i,j)\in C, \{(x_i,d_i),(x_j,d_j)\}\subseteq A} f_{i,j}(d_i, d_j)$, where $d_i \in D_i$ and $d_j \in D_j$. Then, an optimal assignment A^* is given

[1] In this paper, we assume that all constraints are binary for simplicity like many existing COP papers. Relaxing this assumption to general cases is relatively straightforward.

as $\arg\min_A R(A)$, i.e., A^* is an assignment that minimizes the sum of the value of all cost functions, and an optimal value is given by $R(A^*)$. A COP can be represented using a constraint graph, in which nodes correspond to variables and edges represent constraints.

Example 1 (COP)**.** Figure 1 shows a mono-objective COP with three variables x_1, x_2 and x_3. Each variable takes its value assignment from a discrete domain $\{a, b\}$. The figure shows three cost tables among three variables. The optimal solution of this problem is $\{(x_1, a), (x_2, a), (x_3, a)\}$, and the optimal value is 6.

2.2 MO-COP

A *Multi-Objective Constraint Optimization Problem* (MO-COP) (Junker, 2006; Marinescu, 2010; Rollon and Larrosa, 2006) is defined by a set of variables $X = \{x_1, x_2, \ldots, x_n\}$, multi-objective constraints $C = \{C^1, C^2, \ldots, C^m\}$, i.e., a set of sets of constraint relations, and multi-objective functions $F = \{F^1, F^2, \ldots, F^m\}$, i.e., a set of sets of objective functions. For an objective h $(1 \leq h \leq m)$, variables x_i and x_j, which have a constraint relation, the cost for an assignment $\{(x_i, d_i), (x_j, d_j)\}$ is defined by a cost function $f^h_{i,j} : D_i \times D_j \to \mathbb{R}^+$. For an objective h and a value assignment to all variables A, let us denote $R^h(A) = \sum_{(i,j) \in C^h, \{(x_i,d_i),(x_j,d_j)\} \subseteq A} f^h_{i,j}(d_i, d_j)$. Then, the sum of the values of all cost functions for m objectives is defined by a cost vector, denoted $R(A) = (R^1(A), R^2(A), \ldots, R^m(A))$. To find an assignment that minimizes all m objective functions simultaneously is ideal. However, in general, since trade-offs exist among objectives, there does not exist such an ideal assignment. Therefore, the "optimal" solution of an MO-COP is characterized by using the concept of *Pareto optimality*. An assignment is a *Pareto optimal solution* if there does not exist another assignment that weakly improves all of the objectives. Solving an MO-COP is to find the *Pareto front* which is a set of cost vectors obtained by all Pareto optimal solutions. In an MO-COP, the number of Pareto optimal solutions is often exponential in the number of variables, i.e., every possible assignment can be a Pareto optimal solution in the worst case. This problem can be also represented as a constraint graph.

Definition 1 (Dominance)**.** For an MO-COP and two cost vectors $R(A)$ and $R(A')$, we say that $R(A)$ dominates $R(A')$, denoted by $R(A) \prec R(A')$, iff $R(A)$ is partially less than $R(A')$, i.e., it holds (i) $R^h(A) \leq R^h(A')$ for all objectives h, and (ii) there exists at least one objective h', such that $R^{h'}(A) < R^{h'}(A')$.

Definition 2 (Pareto optimal solution)**.** For an MO-COP, an assignment A is said to be the Pareto optimal solution, iff there does not exist another assignment A', such that $R(A') \prec R(A)$.

Table 1: Example of bi-objective COP.

x_1	x_2	*cost*	x_2	x_3	*cost*	x_1	x_3	*cost*
a	a	(5,2)	a	a	(0,1)	a	a	(1,0)
a	b	(7,1)	a	b	(2,1)	a	b	(1,0)
b	a	(10,3)	b	a	(0,2)	b	a	(0,1)
b	b	(12,0)	b	b	(2,0)	b	b	(3,2)

Definition 3 (Pareto Front)**.** Given an MO-COP, the *Pareto front* is the set of cost vectors obtained by the set of Pareto optimal solutions.

Example 2 (MO-COP)**.** Table 1 shows a bi-objective COP, which is an extension of the COP in Figure 1. Each variable takes its value from a discrete domain $\{a, b\}$. The Pareto optimal solutions of this problem are $\{\{(x_1, a), (x_2, a), (x_3, a)\}, \{(x_1, a), (x_2, b), (x_3, b)\}\}$, and the Pareto front is $\{(6,3), (10,1)\}$.

3 DYNAMIC MO-COP

In this section, a framework for *Dynamic Multi-Objective Constraint Optimization Problem* (DMO-COP) is introduced and two solution criteria for solving this problem are provided: *resistance* and *functionality*. Furthermore, an algorithm called *Algorithm for Systems Resilience* (*ASR*) is presented.

3.1 Model

A framework of DMO-COP is defined by a sequence of MO-COPs as follows [2]:

$$\text{DMO-COP} = \langle \text{MO-COP}_0, \text{MO-COP}_1, \ldots, \text{MO-COP}_\text{k} \rangle,$$

where each index i $(0 \leq i \leq k)$ represents a time step. Solving a DMO-COP is finding the following sequence of Pareto front, denoted $\mathbb{PF}$, where each PF_i $(0 \leq i \leq k)$ represents the Pareto front of MO-COP$_i$.

$$\mathbb{PF} = \langle \text{PF}_0, \text{PF}_1, ..., \text{PF}_\text{k} \rangle.$$

Our focus is laid on a *reactive* approach, i.e., each problem MO-COP$_i$ in a DMO-COP can only be known at time step i $(0 \leq i \leq k)$, and we have no information about the problems for any time step j where $j > i$. For dynamic problems, there exist two approaches, namely proactive and reactive. In a proactive approach, all problems in a DMO-COP are known in advance. Since we know all changes among problems, one possible goal of this approach is to find

[2] Similar formalization, i.e., dynamic problem as a sequence of static problems, is provided in many previous works such as (Okimoto et al., 2014; Yeoh et al., 2011).

Table 2: Cost table of MO-COP$_1$.

x_1	x_2	*cost*	x_2	x_3	*cost*	x_1	x_3	*cost*
a	a	(5,2)	a	a	(0,1)	a	a	(1,0)
a	b	(7,1)	a	b	(2,1)	a	b	**(5,5)**
b	a	(10,3)	b	a	(0,2)	b	a	(0,1)
b	b	(12,0)	b	b	(2,0)	b	b	**(1,1)**

Table 3: Cost table of MO-COP$_2$.

x_1	x_2	*cost*	x_2	x_3	*cost*	x_1	x_3	*cost*
a	a	(5,2)	a	a	**(3,3)**	a	a	**(4,4)**
a	b	(7,1)	a	b	(2,1)	a	b	(5,5)
b	a	**(2,2)**	b	a	(0,2)	b	a	(0,1)
b	b	**(3,0)**	b	b	(2,0)	b	b	(1,1)

an optimal solution for a DMO-COP. On the other hand, in a reactive approach, since the new problem typically arises after solving the previous problem, it requires to solve each problem in a DMO-COP one by one. Thus, we need to find a sequence of Pareto front. In the following, the change of the constraint costs among problems in a DMO-COP is studied. [3]

Example 3. Consider a DMO-COP = $\langle$MO-COP$_0$, MO-COP$_1$, MO-COP$_2\rangle$. We use the same example represented in Example 2 and use it as the initial problem of this DMO-COP. The Pareto optimal solutions of MO-COP$_0$ are $\{(x_1,a),\ (x_2,a),\ (x_3,a)\}$ and $\{(x_1,a),\ (x_2,b),\ (x_3,b)\}$, and the Pareto front is $\{(6,3),\ (10,1)\}$ (see. Example 2). Table 2 shows the cost table of MO-COP$_1$. In Table 2, two constraints written in boldface are dynamically changed from the initial problem MO-COP$_0$, i.e., the cost vector of $\{(x_1,a),\ (x_3,b)\}$ and $\{(x_1,b),\ (x_3,b)\}$ are changed from $(1,0)$ to $(5,5)$ and from $(3,2)$ to $(1,1)$. The Pareto optimal solutions of MO-COP$_1$ are $\{(x_1,a),\ (x_2,a),\ (x_3,a)\}$ and $\{(x_1,b),\ (x_2,b),(x_3,b)\}$, and the Pareto front is $\{(6,3),\ (15,1)\}$. Table 3 represents the cost vector of MO-COP$_2$. In Table 3, four constraints written in boldface are additionally changed from MO-COP$_1$ i.e., $(2,2)$, $(3,0)$, $(3,3)$, and $(4,4)$. The Pareto optimal solutions of MO-COP$_2$ are $\{(x_1,b),\ (x_2,b),\ (x_3,a)\}$ and $\{(x_1,b),\ (x_2,b),(x_3,b)\}$, and the Pareto front is $\{(3,3),\ (6,1)\}$. Thus, the solution of this DMO-COP is $\mathbb{PF} = \{\{(6,3),(10,1)\},\{(6,3),(15,1)\},\{(3,3),(6,1)\}\}$.

Now, two solution criteria for DMO-COPs are provided, namely, resistance and functionality. A sequence of assignments $\mathbb{A} = \langle A_0, A_1, ..., A_j\rangle$ is called an *assignment trajectory*, where A_i is an assignment of MO-COP$_i$ $(0 \le i \le j)$. Let m be the number of objectives and $R^h(A_i)$ be the cost for objective h obtained by assignment A_i $(1 \le h \le m)$, and l, q be constant vectors.

Definition 4 (Resistance)**.** For an assignment trajectory $\mathbb{A}$ and a constant vector $l = (l^1, l^2, ..., l^m)$, $\mathbb{A}$ is said to be l-resistant, iff for all h $(1 \le h \le m)$,

$$R^h(A_i) \le l^h,\ (0 \le i \le |\mathbb{A}| - 1).$$

Definition 5 (Functionality)**.** For an assignment trajectory $\mathbb{A}$ and a constant vector $q = (q^1, q^2, ..., q^m)$, $\mathbb{A}$ is said to be q-functional, iff for all h $(1 \le h \le m)$ and for each $j \in \{0, \ldots, |\mathbb{A}| - 1\}$,

$$\frac{\sum_{i=0}^{j} R^h(A_i)}{j+1} \le q^h.$$

Resistance is the ability to maintain some underlining costs under a certain threshold, such that the system satisfies certain hard constraint and does not suffer from irreversible damages, i.e., the ability for a system to stay essentially unchanged despite the presence of disturbances. Functionality is the ability to maintain a guaranteed global quality for the assignment trajectory. While resistance requires to maintain a certain quality level at each problem in a DMO-COP, functionality requires to maintain this level in average, when looking over a certain horizon of time. Thus, functionality evaluates an assignment trajectory globally. The followings are two examples for them. We use the same example as in Example 3.

Example 4 (Resistance)**.** *The Pareto optimal solutions of the DMO-COP is* $\{(x_1,a),\ (x_2,a),\ (x_3,a)\}$ *and* $\{(x_1,a),\ (x_2,b),\ (x_3,b)\}$ *for MO-COP*$_0$, $\{(x_1,a),\ (x_2,a),\ (x_3,a)\}$ *and* $\{(x_1,b),\ (x_2,b),\ (x_3,b)\}$ *for MO-COP*$_1$, *and* $\{(x_1,b),\ (x_2,b),\ (x_3,a)\}$ *and* $\{(x_1,b),\ (x_2,b),\ (x_3,b)\}$ *for MO-COP*$_2$. *The corresponding Pareto front is* $PF_0 = \{(6,3),(10,1)\}$, $PF_1 = \{(6,3),(15,1)\}$, *and* $PF_2 = \{(3,3),(6,1)\}$, *respectively. Let* $l = (8,4)$ *be a constant vector. The assignment trajectory* $\mathbb{A} = \langle A_0, A_1, A_2\rangle$ *with* $A_0 = \{(x_1,a),(x_2,a),(x_3,a)\}$, $A_1 = \{(x_1,a),\ (x_2,a),\ (x_3,a)\}$, *and* $A_2 = \{(x_1,b),\ (x_2,b),\ (x_3,a)\}$ *is l-resistant, since* $R^1(A_0) = 6 < l^1\ (= 8)$, $R^2(A_0) = 3 < l^2\ (= 4)$, $R^1(A_1) = 6 < l^1$, $R^2(A_1) = 3 < l^2$, *and* $R^1(A_2) = 3 < l^1, R^2(A_2) = 3 < l^2$. *Similarly,* $\mathbb{A}' = \langle A_0, A_1, A_2'\rangle$ *is also l-resistant, where* A_0 *and* A_1 *are same as in* $\mathbb{A}$ *and* $A_2' = \{(x_1,b),\ (x_2,b),\ (x_3,b)\}$.

Example 5 (Functionality)**.** *Let* $q = (5,4)$ *be a constant vector. The assignment trajectory* $\mathbb{A} = \langle A_0,\ A_1,\ A_2\rangle$ *with* $A_0 = \{(x_1,a),\ (x_2,a),\ (x_3,a)\}$, $A_1 = \{(x_1,a),\ (x_2,a),\ (x_3,a)\}$, *and* $A_2 = \{(x_1,b),\ (x_2,b),\ (x_3,a)\}$ *is q-functional, since* $(6+6+3)/3 = 5 \le q^1\ (= 5)$

[3] Other changes, e.g., the number of variables, objectives and domain size, can be also considered. In this paper, the focus is laid on the dynamical change of constraint costs among problems. Similar assumption is also introduced in previous works (Okimoto et al., 2014; Yeoh et al., 2011).

and $(3+3+3)/3 = 3 < q^2\ (=4)$. *However, for* $\mathbb{A}' = \langle A_0, A_1, A_2' \rangle$ *where* A_0 *and* A_1 *are same as in* $\mathbb{A}$ *and* $A_2' = \{(x_1,b),\ (x_2,b),\ (x_3,b)\}$, $\mathbb{A}'$ *is not q-functional, since* $(6+6+6)/3 = 6 > q^1\ (=5)$.

3.2 Algorithm

An algorithm, *Algorithm for Systems Resilience* (*ASR*), for solving a DMO-COP is presented. This algorithm is based on the branch-and-bound search, which is widely used for COP and MO-COP algorithms, and it finds all resistant and functional solutions for DMO-COPs. Algorithm 1 shows the pseudo-code of *ASR*. The input is a DMO-COP that is a sequence of MO-COPs and constant vectors l and q. *ASR* outputs a set of sequences (all l-resistant and q-functional solutions). For each *MO-COP*$_i$ ($0 \leq i \leq k$) in the sequence, *ASR* computes a set of all l-resistant solutions denoted $\mathbb{RS}_i^l$ by ASR_{res} (line 6). *ASR* uses the $\otimes$-operator and combines the set of sequences $\mathbb{RS}$ and the cost vectors of $\mathbb{RS}_i^l$ obtained by ASR_{res} (line 10). For example, after the combination of $\mathbb{RS} = \{(6,3),(10,1)\}$ and $\mathbb{RS}_1^l = \{(6,3),(15,1)\}$, i.e., $\mathbb{RS} \otimes \mathbb{RS}_1^l$, there exist four sequences $\{\{(6,3)\},\{(6,3)\}\}$, $\{\{(6,3)\},\{(15,1)\}\}$, $\{\{(10,1)\},\{(6,3)\}\}$ and $\{\{(10,1)\},\{(15,1)\}\}$. For the initial problem, i.e., *MO-COP*$_0$, $\mathbb{RS}$ is equal to $\mathbb{RS}_0^l$ when $\mathbb{RS}_0^l \neq \emptyset$. Next, *ASR* checks the q-functionality of each sequence of $\mathbb{RS}$ (line 11). Finally, it provides a set of all l-resistant and q-functional solutions. Otherwise, it outputs the empty set (checked in line 7-9 and line 12-14). In case l and q are large enough (i.e., no restriction), all Pareto optimal solutions may become l-resistant and q-functional solutions in the worst case, and the size of $\mathbb{RS}$ finally becomes $|\mathbb{RS}_0| \times |\mathbb{RS}_1| \times ... \times |\mathbb{RS}_k|$.

Algorithm 1: *ASR*.

1: INPUT : DMO-COP = $\langle$MO-COP$_0$, MO-COP$_1$, ..., MO-COP$_k\rangle$ and two constant vectors $l = (l^1, l^2, ..., l^m)$, $q = (q^1, q^2, ..., q^m)$
2: OUTPUT : $\mathbb{RS}$ // set of sequences (all l-resistant and q-functional solutions)
3: $\mathbb{RS} \leftarrow \emptyset$
4: $l, q \leftarrow$ constant vectors
5: **for** each MO-COP$_i$ ($i = 0, ..., k$) **do**
6: $\quad \mathbb{RS}_i^l \leftarrow ASR_{res}$(MO-COP$_i$, l) // find all l-resistant solutions
7: $\quad$ **if** $\mathbb{RS}_i^l = \emptyset$ **then**
8: $\quad\quad$ **return** $\mathbb{RS} \leftarrow \emptyset$
9: $\quad$ **end if**
10: $\quad \mathbb{RS} \leftarrow \mathbb{RS} \otimes \mathbb{RS}_i^l$ // combine the current solution with previous solutions
11: $\quad \mathbb{RS}^q \leftarrow ASR_{fun}(\mathbb{RS}, q)$ // filter $\mathbb{RS}$ with q-functionality
12: $\quad$ **if** $\mathbb{RS}^q = \emptyset$ **then**
13: $\quad\quad$ **return** $\mathbb{RS} \leftarrow \emptyset$
14: $\quad$ **end if**
15: $\quad \mathbb{RS} \leftarrow \mathbb{RS}^q$
16: **end for**
17: **return** $\mathbb{RS}$

Algorithm 2: ASR_{res}.

1: INPUT : MO-COP$_i$ and l
2: OUTPUT : $\mathbb{RS}_i^l$ // a set of l-resistant solutions of MO-COP$_i$
3: *Root* : // the root of MO-COP$_i$
4: $AS \leftarrow \emptyset$ // an assignment of variables
5: *Cost* $\leftarrow$ null vector // cost vector obtained by *AS*
6: $\mathbb{RS}_i^l \leftarrow \emptyset$ // a set of pairs <cost vector, set of assignments>
// Launch solving from the root
7: $\mathbb{RS}_i^l \leftarrow first.solve(AS, Cost, \mathbb{RS}_i^l, l)$
8: **return** $\mathbb{RS}_i^l$

Algorithm 3: solve($AS, Cost, \mathbb{RS}_i^l, l$).

1: INPUT : $< AS, Cost, \mathbb{RS}_i^l, l >$
2: OUTPUT : $\mathbb{RS}_i^l$
3: **for** each value v_1 of the variable domain **do**
4: $\quad AS \leftarrow v_1$
5: $\quad local\_cost \leftarrow$ null vector
$\quad$ // step 3.1: Compute local cost of the choice
6: $\quad$ **for** each constraint with an ancestor a **do**
7: $\quad\quad v_2 \leftarrow$ value of a in AS
8: $\quad\quad local\_cost \leftarrow local\_cost + cost(v_1, v_2)$
9: $\quad$ **end for**
10: $\quad new\_cost \leftarrow Cost + local\_cost$
$\quad$ // step 3.2: Bound checking
11: $\quad dominated \leftarrow$ **false**
12: $\quad$ **for** each objective h ($1 \leq h \leq m$) **do**
13: $\quad\quad$ **if** $r^h > l^h, r^h \in new\_cost$ **then**
14: $\quad\quad\quad AS \leftarrow (AS \setminus v_1)$
15: $\quad\quad\quad dominated \leftarrow$ **true**
16: $\quad\quad$ **end if**
17: $\quad$ **end for**
18: $\quad$ **if** *new_cost* is dominated by $\mathbb{RS}_i^l$ **then**
19: $\quad\quad AS \leftarrow (AS \setminus v_1)$
20: $\quad\quad dominated \leftarrow$ **true**
21: $\quad$ **end if**
22: $\quad$ **if** *dominated* **then**
23: $\quad\quad$ **continue**
24: $\quad$ **end if**
$\quad$ // step 3.3.1: New Pareto optimal solution
25: $\quad$ **if** AS is complete **then**
26: $\quad\quad E \leftarrow$ all elements of $\mathbb{RS}_i^l$ dominated by *new_cost*
27: $\quad\quad \mathbb{RS}_i^l \leftarrow \mathbb{RS}_i^l \setminus E$
28: $\quad\quad \mathbb{RS}_i^l \leftarrow \mathbb{RS}_i^l \cup \{(new\_cost, AS)\}$
29: $\quad\quad$ **continue**
30: $\quad$ **end if**
$\quad$ // step 3.3.2: Continue solving
31: $\quad \mathbb{RS}_i^l \leftarrow next.solve(AS, Cost, \mathbb{RS}_i^l, l)$
32: $\quad AS \leftarrow (AS \setminus v_1)$
33: **end for**
34: **return** $\mathbb{RS}_i^l$

Let us describe ASR_{res}. It finds all Pareto optimal solutions of each problem in a sequence, which are bounded by the parameter l. The pseudo-code of ASR_{res} is given in Algorithm 2 and 3. We assume that a variable ordering, i.e., pseudo-tree (Schiex et al., 1995), is given. The input is an *MO-COP*$_i$ ($0 \leq i \leq k$) and a constant vector l, and the output is the entire set of l-resistant solutions (lines 1 and 2 in Algorithm 2). ASR_{res} starts with an empty set of l-resistant solutions and a null cost vector, and solves the first variable according to the variable ordering (lines 3-7). It chooses

Algorithm 4: ASR_{fun}.

1: INPUT : $\mathbb{RS}, q$
2: OUTPUT : $\mathbb{RS}^q$ // set of the filtered sequences
3: $\mathbb{RS}^q \leftarrow \emptyset$
4: **for** each sequence $S_j \in \mathbb{RS}$ $(0 \leq j \leq |\mathbb{RS}|-1)$ **do**
5: **for** each h $(1 \leq h \leq m)$ **do**
6: **if** $\frac{\sum_{i=0}^{|S_j|-1} s_{i,j}^h}{|S_j|} > q^h$ // $s_{i,j} \in S_j$ **then**
7: **return** $\mathbb{RS}^q = \emptyset$
8: **end if**
9: **end for**
10: $\mathbb{RS}^q \leftarrow S_j$
11: **end for**
12: **return** $\mathbb{RS}^q$

a value for the variable and updates the cost vector according to the cost tables (step 3.1 in Algorithm 3). At this moment the obtained cost vector has to ensure the following two properties: (i) r^h (the cost for objective h) is bounded by the constant vector l and (ii) the cost vector is not dominated by another cost vector (i.e., current l-resistant solutions) in $\mathbb{RS}_i^l$. If one of the two properties is violated, ASR_{res} branches on the next value of the variable. When its domain is completely explored, the search branches to the previous variable and continues the solving (step 3.2 in Algorithm 3). When all assignments are formed, i.e., no variable left to be assigned, a new solution is added to $\mathbb{RS}_i^l$. All previous dominated solutions are removed from $\mathbb{RS}_i^l$ and the search continues with the next values of the variable (step 3.3.1 in Algorithm 3). In case r^h fulfills the two properties, it continues the solving with the next variable according to the variable ordering (step 3.3.2). The search stops when the whole search space has been covered by the branch-and-bound search. ASR_{res} finally outputs the set of l-resistant solutions (which are not dominated by other solutions) [4].

Let us describe ASR_{fun}. Algorithm 4 shows the pseudo-code of it. The input is a set of sequences obtained by ASR_{res} and a constant parameter q, and output is a set of l-resistant and q-functional solutions $\mathbb{RS}^q$ (lines 4-11). For each sequence S_j of $\mathbb{RS}$, it checks the q-functionality by using the equation given in Definition 5 (lines 5-9). *ASR* is a complete algorithm, i.e., it provides a set of all l-resistant and q-functional solutions if there exists. Otherwise, it outputs the empty set (lines 7-9 and 12-14 in Algorithm 1).

[4] ASR_{res} checks the dominance among the solutions (in lines 18, 22 and 26 in Algorithm 3) and provides the set of Pareto optimal solutions bounded by the parameter l.

4 SOME EXPERIMENTAL RESULTS

In this section, the performances of *ASR* are evaluated with different types of dynamical changes. In the experiments, we generate DMO-COPs that contain three MO-COPs as in Example 3. All the tests are made with 20 variables, the domain size of each variable is 2, the number of objectives is 2, and the cost of each constraint is a random value between 0 and 100. In DMO-COPs, we change a fixed proportion of constraint costs (called the change ratio) at each dynamic step. For the initial problem, we choose the constraint costs from [0:100]. Then, we create the next problem by changing the proportion of constraints costs defined by the change ratio. For example, in case the change ratio is 5%, we choose 5% of all constraints in the current problem and change their constraint costs by selecting the new costs from the range [100:200], but do not change the remaining constraint costs. Each data point in a graph represents the average of 50 problem instances.

Figure 2 and 3 show the average number of solutions and its runtime obtained by *ASR*. The l-ratio is provided by $l/(cost_{max} \times \#constraints)$, where $cost_{max}$ is the maximal cost value (i.e., 200). We vary the the change ratio from 0.05 to 0.5 and from 0.3 to 1.0 for l-ratio. In this experiment, we set the constant vector to $q = (q_{max}, q_{max})$, where $q_{max} = 3 \times (cost_{max} \times$

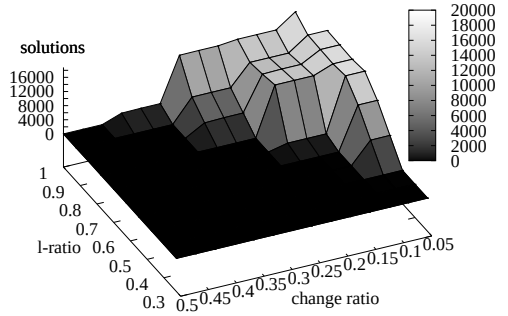


Figure 2: Number of solutions.

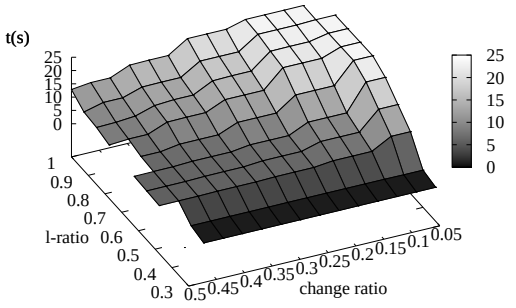
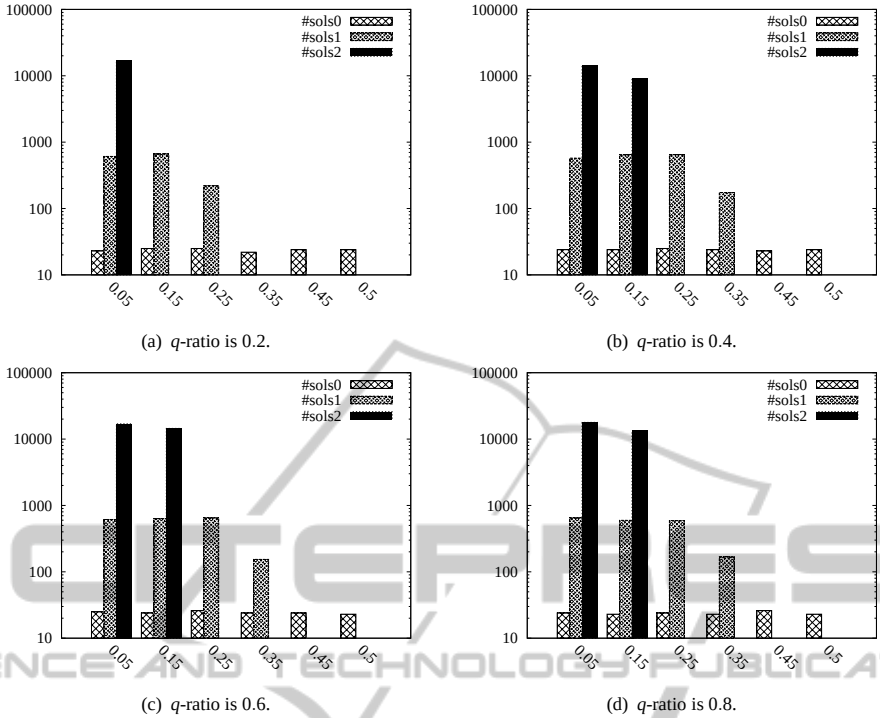


Figure 3: Runtime.



(a) q-ratio is 0.2. (b) q-ratio is 0.4.

(c) q-ratio is 0.6. (d) q-ratio is 0.8.

Figure 4: The average number of resistant and functional solutions of each problem in DMO-COPs.

#constraints) which is large enough, so that all l-resistant solutions become q-functional. In figure 2, we can observe that only small change ratio (i.e., from 0.05 to 0.15) for the l-ratio from 0.5 to 1.0 allows us to find resistant (and also functional) solutions for DMO-COPs. On the other hand, in case the l-ratio is small and the change ratio is large, *ASR* cannot find any solutions. In figure 3, we observed that the average runtime increases where the number of resistant (and also functional) solutions becomes large.

Figure 4 shows the results for the average number of obtained resistant and functional solutions of each problem in DMO-COP, i.e., MO-COP$_0$, MO-COP$_1$ and MO-COP$_2$, for varying the q-ratio. In this experiment, we set l-ratio to 0.8. The #*sols*0 represents the number of obtained resistant and functional solutions in MO-COP$_0$, #*sols*1 is for MO-COP$_1$, and #*sols*2 shows that for MO-COP$_2$. The x axis shows the change ratio and the y axis is the number of solutions obtained by *ASR*. For any q-ratio, we find solutions for the initial problem (i.e., MO-COP$_0$) in DMO-COPs. It is the problem where Pareto optimal solutions have the lowest cost vectors. Once this problem changes (with regard to the change ratio), the average cost vector increases and the functionality becomes harder to obtain. For small changes where the change ratio is 0.05, even a low q-ratio ($q = 0.2$) allows to find resistant and functional solutions (Figure 4(a)). However, for more drastic changes, e.g., the change ration is 0.15, we need a higher q-ratio in order to find solutions after the third problem. We then reach a point where the change ratios are too severe (i.e., 0.25-0.5) to find solutions for the third problem. We can increase the q-ratio but we cannot find resistant and functional solutions (Figure 4(b)- 4(d)).

In summary, these experimental results reveal that the performance of *ASR* is highly influenced by change ratio. *ASR* can obtain the resistant and functional solutions of a DMO-COP, when the dynamical changes are small (i.e. the change ratio is from 0.05 to 0.15). Otherwise, *ASR* outputs empty set quickly.

5 RELATED WORKS

Various algorithms have been developed for MO-COPs (Marinescu, 2010; Perny and Spanjaard, 2008; Rollon and Larrosa, 2006; Rollon and Larrosa, 2007). Compared to these sophisticate MO-COP algorithms, *ASR* solves a DMO-COP and finds a subset of the Pareto front, i.e., a set of resistant and functional solutions. Furthermore, there exist several works on dynamic constraint satisfaction problem (Dechter and Dechter, 1988; Faltings and Gonzalez, 2002). Compared to these works, there exists few works on COPs with “multiple criteria” in a “dynamic environment”, as far as we know.

Schwind et al.(2013) introduced the topic of systems resilience, and defined a resilient system as a dynamic constraint-based model called SR-model. They captured the notion of resilience for dynamic systems using several factors, i.e., resistance, recoverability, functionality and stabilizability. In our work, as an initial step toward developing an efficient algorithm for finding resilient solutions of a DMO-COPs, we focus on two properties, namely resistance and functionality, which are properties of interest underlying the resilience for DMO-COPs. Compared to (Schwind et al., 2013), this paper provides an algorithm (*ASR*) which can computes resistant and functional solutions for DMO-COPs, while (Schwind et al., 2013) does not show any computational algorithms for these two properties, and they use the framework for dynamic (singe-objective) COPs. Both properties are related to an important concept underlying resilience. Indeed, these properties are faithful with the initial definition of resilience proposed by Holling (1973), as to "determine the persistence of relationships within a system and is a measure of the ability of these systems to absorb changes of state variables, driving variables, and parameters, and still persist." In contrast, Bruneau's (Bruneau, 2003) definition of resilience corresponds to the minimization of a triangular area representing the degradation of a system over time. This definition has been formalized under the name "recoverability" for Dynamic COP in (Schwind et al., 2013). We will investigate it in future work.

6 CONCLUSION

The contribution of this paper is mainly twofold:

- A framework for Dynamic Multi-Objective Constraint Optimization Problem (DMO-COP) has been introduced. Also, two solution criteria have been imported from Schwind et al. (2013) and extended to DMO-COPs, namely, resistance and functionality, which are properties of interest underlying the *resilience* for DMO-COPs.
- An algorithm called *ASR* for solving a DMO-COP has been presented and evaluated. *ASR* aims at computing every resistant and functional solution for DMO-COPs.

As a perspective for further research, we intend to apply our approach to some real-world problems, especially dynamic sensor network and scheduling problems, and will develop algorithms that are specialized to these application problems (by modifying *ASR*).

REFERENCES

Bruneau, M. (2003). A Framework to Quantitatively Assess and Enhance the Seismic Resilience of Communities. In *Earthquake Spectra*, volume 19.

Cabon, B., Givry, S., Lobjois, L., Schiex, T., and Warners, J. (1999). Radio link frequency assignment. *Constraints*, 4(1):79–89.

Dechter, R. (2003). *Constraint Processing*. Morgan Kaufmann Publishers.

Dechter, R. and Dechter, A. (1988). Belief maintenance in dynamic constraint networks. In *AAAI*, pages 37–42.

Faltings, B. and Gonzalez, S. (2002). Open constraint satisfaction. In *CP*, pages 356–370.

Holling, C. (1973). Resilience and stability of ecological systems. *Annual Review of Ecology and Systematics*, 4:1–23.

Junker, U. (2006). Preference-based inconsistency proving: When the failure of the best is sufficient. In *ECAI*, pages 118–122.

Longstaff, P. H., Armstrong, N. J., Perrin, K., Parker, W. M., and Hidek, M. A. (2010). Building resilient communities: A preliminary framework for assessment. *Homeland Security Affairs*, 6(3).

Marinescu, R. (2010). Best-first vs. depth-first and/or search for multi-objective constraint optimization. In *ICTAI*, pages 439–446.

Okimoto, T., Ribeiro, T., Clement, M., and Inoue, K. (2014). Modeling and algorithm for dynamic multi-objective weighted constraint satisfaction problem. In *ICAART*, pages 420–427.

Perny, P. and Spanjaard, O. (2008). Near admissible algorithms for multiobjective search. In *ECAI*, pages 490–494.

Rollon, E. and Larrosa, J. (2006). Bucket elimination for multiobjective optimization problems. *Journal of Heuristics*, 12(4-5):307–328.

Rollon, E. and Larrosa, J. (2007). Multi-objective russian doll search. In *AAAI*, pages 249–254.

Sandholm, T. (1999). An algorithm for optimal winner determination in combinatorial auctions. In *IJCAI*, pages 542–547.

Schiex, T., Fargier, H., and Verfaillie, G. (1995). Valued constraint satisfaction problems: Hard and easy problems. In *IJCAI*, pages 631–639.

Schwind, N., Okimoto, T., Inoue, K., Chan, H., Ribeiro, T., Minami, K., and Maruyama, H. (2013). Systems resilience: a challenge problem for dynamic constraint-based agent systems. In *AAMAS*, pages 785–788.

Verfaillie, G., Lemaître, M., and Schiex, T. (1996). Russian doll search for solving constraint optimization problems. In *AAAI/IAAI, Vol. 1*, pages 181–187.

Walker, B. H., Holling, C. S., Carpenter, S. C., and Kinzig, A. (2004). Resilience, adaptability and transformability. *Ecology and Society*, 9(2):5.

Yeoh, W., Varakantham, P., Sun, X., and Koenig, S. (2011). Incremental DCOP search algorithms for solving dynamic DCOPs. In *AAMAS*, pages 1069–1070.