

# Dynamic combinatorial optimisation problems: an analysis of the subset sum problem

Philipp Rohlfshagen · Xin Yao

© Springer-Verlag 2010

**Abstract** The field of evolutionary computation has traditionally focused on static optimisation problems. Recently, many new approaches have been proposed that adapt traditional evolutionary algorithms to the dynamic domain to deal with the task of tracking high-quality solutions as the search space changes over time. These novel algorithms are subsequently evaluated on a wide range of different optimisation problems, including well-specified benchmark generators. However, due to a lack of theoretical results, as well as a general lack of references to actual real-world scenarios, it is not entirely clear whether these benchmarks capture any of the characteristics found in NP-hard dynamic optimisation problems. In this paper, we extensively analyse the properties of the NP-hard (dynamic) subset sum problem. In particular, we highlight the correlation between the dynamic parameters of the problem and the resulting movement of the global optimum. It is shown by empirical means that the degree to which the global optimum moves in response to the underlying dynamics is correlated only in specific cases. Furthermore, the role of the representation used to encode the problem, as well as the impact of the formulation of the objective function on the dynamics are also discussed.

**Keywords** Evolutionary computation · Combinatorial optimisation · Dynamic optimisation · Subset sum problem · Fitness landscapes

---

P. Rohlfshagen (✉) · X. Yao  
School of Computer Science, University of Birmingham,  
Birmingham B15 2TT, UK  
e-mail: P.Rohlfshagen@cs.bham.ac.uk

X. Yao  
e-mail: X.Yao@cs.bham.ac.uk

## 1 Introduction

The number of publications in the field of dynamic evolutionary computation (an umbrella term we have chosen to denote the use of evolutionary algorithms to deal with dynamic optimisation problems, DOPs) has increased significantly in recent years. In particular, several monographs (Branke 2002; Morrison 2004; Weicker 2003), PhD theses (e.g. Liekens 2005; Wilke 1999) as well as edited volumes (Yang et al. 2007) have emerged that deal with this particular area of research. The majority of work is motivated by the presence of real-world problems that are inherently dynamic: solutions to such problems need to be re-optimised, as time goes by to ensure feasibility and satisfactory quality. Unfortunately, the majority of papers provide not much (if any) additional context regarding the nature and impact of the dynamics on the actual problem they are concerned with. Many authors have suggested new dynamic versions of well-studied static optimisation problems by defining arbitrary dynamics that are often tailored towards the algorithm to be tested.<sup>1</sup> Alternatively, authors make use of artificial benchmark problem generators, of which, however, only few exist.

The use of benchmark problems is vital in driving forward this relatively young field of research. However, there is a danger that the design of such benchmarks is largely based on the preconceptions about the dynamic domain that are not necessarily grounded in empirical or theoretical evidence. For instance, it has recently been shown theoretically how common assumptions about the dynamic domain may fail in certain cases (Rohlfshagen et al. 2009). It is thus not entirely clear what these benchmarks actually test, a point raised previously by Ursem et al. (2002) (see Sect. 2.3). This issue

---

<sup>1</sup> The update period, for instance, is often set to correspond to multiples of the population size of the algorithm considered.

is crucial as it is apparent that not every problem that somehow changes over time reflects the kind of dynamics one is ultimately interested in. The no free lunch theorem for dynamic optimisation implies that “if one algorithm outperforms another for certain kinds of cost function dynamics, then the reverse must be true on the set of all other cost function dynamics” (Wolpert and MacReady 1997, p. 70). Therefore, it is vital to identify, if possible, characteristics that are typical of difficult DOPs, especially those that occur in practical scenarios. This would allow one to design new algorithms that work well over a range of dynamics that are of interest and fail on those that are non-representative.

In this paper, we attempt to contribute towards a better understanding of how the dynamics of a problem affect the search space over time. We concentrate on the subset sum problem, a problem commonly found in many practical applications (e.g. cryptography; see Sect. 4), and present an extensive empirical study that makes, to some extent, use of the fitness landscape metaphor to show how the search space is transformed over time as the parameters of the problem are subject to change. This study is an extension of work carried out previously (Rohlfshagen and Yao 2008) and although our results cannot, without further investigation, be generalised to other dynamic combinatorial optimisation problems, the framework proposed in this paper demonstrates the use of techniques that are sufficiently general to be applicable to a wide range of other combinatorial optimisation problems (especially pseudo-boolean ones).

The remainder of this paper is structured as follows: first, we provide a brief overview of the field of evolutionary dynamic optimisation and review commonly used benchmark problems in Sect. 2. We then present an extended definition of DOPs in Sect. 3 with special emphasis on the underlying dynamics. Section 4 introduces the subset sum problem followed by a discussion of the dynamic subset sum problem and combinatorial fitness landscapes in Sect. 5. The experimental results are presented in Sect. 6 followed by conclusions and a brief discussion of future work in Sect. 7.

## 2 Evolutionary dynamic optimisation

### 2.1 A brief introduction

In general terms, “a DOP is a problem that changes over time”, that is, the specifications of the problem are time variant.<sup>2</sup> In numerous cases, this concept has been

<sup>2</sup> Here we use the terms *static*, *stationary* and *time invariant* interchangeably to refer to problems that do not change over time. Problems that do change over time, on the other hand, are described as *dynamic* or *time variant*.

translated directly into new DOPs where an otherwise static problem is made dynamic by some alteration over time and solved using a newly proposed method. A simple example of a DOP is the moving sphere function,  $f(\mathbf{x}) = g(|\mathbf{x}^* - \mathbf{x}|)$ , where the dynamics, with magnitude  $\theta$  and direction  $\mathbf{v}$ , are modeled as  $\mathbf{x}^*(t+1) = \mathbf{x}^*(t) + \theta\mathbf{v}$  (see Rossi et al. 2007 for details).

The general objective of dynamic optimisation is similar to that of the static case except that the parameters of the problem,  $\gamma$ , now depend on time  $t$  and hence the algorithm needs to continuously find the currently best solution (adapted from Bosman and Poutre 2007):

$$\max_{\mathbf{x}(t)} \left\{ \sum_{t=0}^{\text{end}} f(\gamma(t), \mathbf{x}(t)) \right\} \quad (1)$$

where  $f(\gamma(t), \mathbf{x}(t))$  is the fitness function and  $\mathbf{x}(t)$  is the candidate solution at time  $t$ . Without the loss of generality, we assume here that we maximise a fitness (as opposed to minimise a cost) and that time is discreet. It is clear that in order to optimise Eq. 1, the algorithm is required to not only locate the global optimum as quickly as possible, but also to track said optimum over time. The global optimum is defined as the element  $\mathbf{x}^* \in X$  such that  $f(\mathbf{x}^*) \geq f(\mathbf{x}), \forall \mathbf{x} \in X$ .

### 2.2 Evolutionary approaches to dynamic optimisation

The principle idea that motivates the majority of work in dynamic evolutionary optimisation is the reuse of information uncovered in the past (and, to a lesser extent, the prediction of future dynamics). In other words, most evolutionary approaches to DOPs attempt to reduce the computational complexity of the dynamic problem by “transferring knowledge from the past” (Branke 2002). In typical black box optimisation, a lack of instance specific knowledge requires the search to start from uniformly random points in the search space. In the dynamic case, on the other hand, once a change occurs, the algorithm has already sampled a number of unique search points and it is generally hoped that this information may be used to reduce the computational complexity of tracking the global optimum. This, of course, requires that the successive search spaces are somehow correlated to one another. In scenarios, where the global optimum “moves”, on an average, relatively little, for instance, it may be beneficial to restrict the search to the vicinity of the old global optimum. It is also possible to extract additional statistical information from the search points sampled to identify exploitable features of the search space or to use such information to predict future dynamics. The desire to speed up the optimisation process is particularly relevant in the dynamic domain (i.e. possibly even more so than in the stationary case) where the time to re-locate the

global optimum may be severely restricted by the frequency of change. Many different algorithms have been proposed in recent years to deal explicitly with the dynamic domain and the key ideas are reviewed next.

Random restarts are amongst the simplest approaches to deal with DOPs<sup>3</sup> although this technique does not utilise any knowledge from the past. At the other extreme would be a “continuous” approach where the algorithm does not take any action at all. Here one has to be careful not to reuse outdated fitness values (i.e. fitness values obtained and stored before the fitness function changed). Furthermore, it is generally assumed that convergence of the population, a desirable feature in the stationary domain (unless premature), may severely limit the algorithm’s ability to explore the search space once a change has occurred. An intermediate approach is to increase the diversity of the population [e.g. via hyper-mutations, Cobb (1990) or random immigrants; Grefenstette (1992)], either throughout the execution of the algorithm, or whenever an environmental change has been detected. This preserves some of the information contained within the population yet also introduces additional diversity that may be required to re-locate the global optimum.

In many approaches, such as the ones mentioned above, knowledge about the search space is stored implicitly in the population of candidate solutions (somewhat acting as a memory) and numerous attempts have been made to extend this repertoire of points. A widespread technique is the use of additional memory, either implicit or explicit. In the former case, the memory is embedded in the encoding (i.e. representation of the problem), usually in the form of diploidy (e.g. Goldberg and Smith 1987) or polyploidy (e.g. Hadad and Eick 1997). This approach has been used successfully for small numbers of distinct states, but the space requirements and the memory’s loss of integrity over time seem to prevent this technique from scaling to larger numbers of states. Subsequently, more research has focused on explicit memory schemes which attempt to maintain a (diverse) register of previously good solutions that are not subject to modification by the algorithm’s genetic operators (e.g. Yang 2005, 2006). In cases where the environment returns to a point similar or identical to a previously visited one, solutions from the register may be reused efficiently. Empirical evidence seems to suggest that explicit memory may also be utilised successfully in random, nonrecurrent dynamics (Yang 2005) (probably as a source of diversity).

Another promising approach is the use of multiple populations, each of which keeps track of a promising part of the search space (e.g. optima encountered in the past).

<sup>3</sup> Assuming it is computationally tractable to detect when a change has occurred.

This concept is closely related to speciation in nature where an existing species may diverge as parts of the population adapt to different ecological niches. Alternatively, a central population could keep track of the overall search, while multiple smaller populations diverge to track additional regions of the search space (Branke et al. 2000). Finally, more recent approaches attempt to exploit the concepts of anticipation and prediction (e.g. Bosman 2005; van Hemert et al. 2001) where data about the problem is collected online and used to predict future dynamics. This is particularly, useful if the underlying dynamics of the problem is systematic (e.g. recurrent) or if the current solution affects the dynamics of the search space [i.e. there is time-linkage (Bosman and Poutre 2007)].

### 2.3 Benchmark problems

It is fair to say that the field of evolutionary dynamic optimisation is still in its infancy, evident by the relative lack of theoretical work, commonly accepted frameworks and the limited number of readily available benchmark problems. Here, we use the term benchmark to describe a framework that is able to simulate a wide range of different dynamics that may be replicated on demand. It is clear that such tools are essential in the development of new algorithms. The three most widely used benchmarks are due to Branke (2002, Moving Peaks), Morrison (2004, DF1) and Yang (2003, XoR). The former two are based on the continuous domain and model the search space as a “field of cones” (Morrison and DeJong 1999), where each cone may be controlled individually to model different ranges of dynamics. Although both benchmark generators are conceptually very similar, they are modelled in different ways and here we focus on DF1. In the two-dimensional case, the base function is given by

$$f(x, y) = \max_{i=1, \dots, N} \left[ h_i - r_i \cdot \sqrt{(x - x_i)^2 + (y - y_i)^2} \right] \quad (2)$$

where  $N$  is the number of cones, each at location  $(x_i, y_i)$ , and with height  $h_i$  and slope  $r_i$ . The initial morphology is randomly generated within the bounds specified by the user, and the dynamics are modelled using a logistic function  $y(t) = \alpha y(t-1)(1 - y(t-1))$  where  $\alpha \in [1, 4]$  is constant. The logistic function, the output of which is in  $[0, 1]$ , may generate different trajectories (depending on  $\alpha$ ) ranging from stationary to recurrent and chaotic. The motivation behind this approach is to create a problem, where the majority of changes are smooth and correlated with the occasional drastic change when the height of the peaks change in such a way that a local optimum surpasses the global one (Branke 1999). There are numerous other continuous problems that have been used in the evaluation of new algorithms, including the simple moving-sphere

function mentioned earlier. However, these latter problems have not gained the same widespread acceptance as DF1 or Moving Peaks.

XoR is the only widely accepted benchmark problem for the combinatorial domain and is hence of particular interest to this paper. XoR can generate a dynamic version of any static binary problem: given a static fitness function  $f(x)$ , where  $x \in \{0, 1\}^n$ , its dynamic equivalent is simply  $f(x(t) \oplus m(T))$  where  $\oplus$  is the bit-wise exclusive-or operator. The period index  $T = \lceil t/\tau \rceil$  is determined by the update period  $\tau$  which is usually measured as the number of generations between changes (i.e.  $1/\tau$  is the frequency of change). The vector  $m(T) \in \{0, 1\}^n$ , initially  $m(1) = \mathbf{0}$ , is a binary mask for period  $k$ , generated as follows:  $m(T) = m(T - 1) \oplus p(T)$  where  $p(T) \in \{0, 1\}^n$  is a randomly created template for period  $T$  that contains exactly  $\lfloor \rho n \rfloor$  ones. The value of  $\rho \in [0, 1]$  thus controls the magnitude of change which is specified as the Hamming distance between two binary points,  $d_X(x, y) = \sum_{i=1}^n |x_i - y_i|$ . It follows that the algorithm is required to invert  $\rho n$  bits to relocate the global optimum (assuming the global optimum was found previously). Finally, it should be noted that both  $\tau$  and  $\rho$  are usually kept constant throughout the execution of the algorithm.

These benchmarks allow full control over some important aspects of the dynamics (e.g. frequency and magnitude of change), but have been criticised by Ursem et al. (2002, p. 1), as “no research has been conducted to thoroughly evaluate how well they reflect characteristic dynamics of real-world problems.” This notion is supported by the fact that none of these benchmark problems consider (dynamic) constraints or the issue of time linkage. Furthermore, the dynamics considered are usually random and only little or no attention is paid to the characteristics of the dynamics or the attributes of the current configuration of the problem. Finally, as XoR does not actually alter the underlying search space in any way (the search points are rotated by some degree but the underlying function is static) it is limited in its applicability (Tinos and Yang 2007).

### 3 Dynamic optimisation problems

So far, we described a DOP simply as  $f(y(t), x(t))$ . In this section, we extend and elaborate on this definition to place more emphasis on the actual dynamics and to distinguish between the dynamics in the parameter space and their impact on the search space (fitness landscape). It should be noted that there are many different types of DOPs (in principle, every aspect of an optimisation problem may be time variant, including attributes, such as the domain and dimensionality of the decision variables) and a completely unified definition is beyond the scope of this paper. Instead,

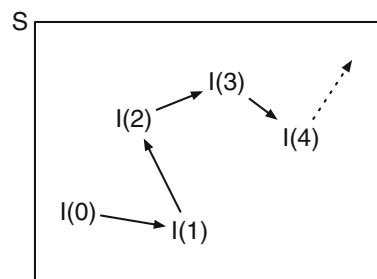


Fig. 1 Visualisation of a DOP: the dynamics  $\mathcal{T}$  describe a trajectory through the space of instances  $S$  over time

we concentrate on the most commonly considered type of DOP in the literature which views a DOP as a time-variant sequence of instances  $I(\cdot)$  of the same problem  $\Pi$ ,

$$I(T = 0) \longrightarrow I(T = 1) \longrightarrow I(T = 2) \longrightarrow \dots \tag{3}$$

where  $T$  is the period index as defined below. The distinction between a problem and instances of said problem is vital and to explain these concepts further, we extend the definition of combinatorial optimisation problems suggested by Garey and Johnson (1979): a combinatorial optimisation problem  $\Pi$  (either maximisation or minimisation) consists of the following three parts:

1. A set  $S$  of instances  $I$ .
2. A finite set of candidate solutions  $X$  for each  $I \in S$ .
3. A function  $f : S \times X \rightarrow \mathbb{R}$  that assigns a positive rational number  $f(I, x)$  to each instance  $I \in S$  and each candidate solution  $x \in X$ .

In addition, the majority of real-world problems have numerous constraints a solution must satisfy. The degree to which a solution  $x$  violates any of the  $k$  constraints is given by the function  $g : S \times X \rightarrow \mathbb{R}^k$ . Using this notation, we define a DOP as  $\Pi_D = (\Pi, \mathcal{T})$  with dynamic objective function  $f(I(T), x(t))$  where  $x(t) \in X$  is the candidate solution at time  $t$  for instance  $I(T)$  and  $I(T + 1) = \mathcal{T}(I(T), t)$  (also see Wolpert and MacReady (1997).

The dynamics may be viewed as a trajectory through the space of all instances over time (state space) and this trajectory is described by  $\mathcal{T}$ , sometimes called the meta dynamics. This concept is shown in Fig. 1. It follows that the problem  $\Pi_D$  is specified not only by  $S$ , but also by the set of (dynamic) instances  $S_D$ , where each  $\mathcal{T} \in S_D$  is a mapping  $\mathcal{T} : S \times \mathbb{N} \rightarrow S$  that consists of a set of difference equations (possibly including the identify function):

$$\mathcal{T}(I(T), T) = \begin{pmatrix} \mathcal{T}_1(I_1(T), T) \\ \vdots \\ \mathcal{T}_k(I_k(T), T) \end{pmatrix} \tag{4}$$

The dynamics encapsulated by  $\mathcal{T}$  generate a new instance at period  $T + 1$  given the previous instance:  $I(T + 1) = \mathcal{T}(I(T), T)$  where  $T$  represents an environmental index

such that  $T \cdot \tau \leq t < (T + 1) \cdot \tau$  and where  $1/\tau$  is the frequency of change (usually constant, but could also be time variant). The actual time  $t$  is assumed to be discrete and advances with every function evaluation. A particular instance of a DOP is then given by the tuple  $(\Pi, \mathcal{T}_\Pi, I(0))$  where  $I(0)$  is the initial instance of the dynamic problem.

The distinction between a problem  $\Pi$  and instances of said problem,  $S$ , is crucial. The problem  $\Pi$  is an unambiguous description (mathematical or otherwise) of a problem and includes the definition of the objective function. Each instance contains the information (parameters and constraints) to actually compute the quality of a solution (decision variables). In static optimisation, one might be given an arbitrary function such as  $f(x) = 2x^2, x \in [l, u]$ , and is asked to find the maximum/minimum. In the dynamic case, one is given a class of functions described by the more general form  $f(x; a, b) = ax^b$  and the algorithm has to follow the time-variant global optimum through the state space as  $a$  and  $b$  change over time.

Finally, it should be noted that the concept of dynamic optimisation is strongly related to the field of dynamical systems. In dynamical systems, one attempts to formulate a rule that describes the time-dependent trajectory of a point  $x$  in its ambient space. The behaviour of  $x$  is given by some difference equation  $h(x)$  (e.g. a logistic function) such that

$$x(t + 1) = h(x(t)) \tag{5}$$

The exact trajectory of  $x$  depends on  $x(0)$  and  $h(x)$ . In dynamic optimisation, the point  $x$  would correspond to  $\mathbf{x}^*$ . Relatively not much research has been carried out to date that examines how the trajectory of  $\mathbf{x}^*$  is correlated to the trajectory of instances in  $S$ . The remainder of this paper will examine this relationship in the case of the subset sum problem.

#### 4 The stationary subset sum problem

There are numerous variations of the classical subset sum problem, which is an NP-complete decision problem that may be solved in pseudo-polynomial time (see Garey and Johnson 1979). In this paper, we consider its NP-hard optimisation variant: given a set of positive weights  $W$  and a positive integer  $c$ , the task is to find a subset of weights the sum of which is as close as possible to  $c$ , without exceeding it. For instance, given  $W = \{2, 7, 9, 12, 17\}$  and  $c = 18$ , the optimal solution is  $\{2, 7, 9\}$ . There are a total of  $2^{|W|}$  subsets and it is natural to treat the subset sum problem as a pseudo-boolean optimisation problem, modelling the set  $W$  as a vector  $\mathbf{w}$ . The objective may then be stated as

$$\max_{\mathbf{x}} \left\{ \sum_{i=1}^n w_i x_i \right\} \text{ w.r.t } c - \sum_{i=1}^n w_i x_i \geq 0 \tag{6}$$

where  $x_i \in \{0, 1\}$  and  $1 \leq w_i \leq m$  for all  $i$  and  $0 < c \leq nm$ . The parameter  $m$  is a predefined limit on the weights. An instance of the subset sum problem is thus fully defined as  $I = (\mathbf{w}, c)$ .

The subset sum problem is a frequently encountered problem with many practical applications. For instance, solutions to the subset sum problem may be used to design better lower bounds for scheduling problems or to tighten the constraints of 0–1 integer programs (see Kellerer et al. 2004). The subset sum problem also appears in many other combinatorial optimisation problems such as bin packing or cryptography. The subset sum problem has a single constraint that needs to be satisfied for a solution to be valid. Constraints may be dealt with in numerous different ways such as penalty functions, repair algorithms and decoders. Penalty functions represent the easiest approach and may be added easily to any existing algorithm. We make use of the penalty function suggested by Wong (2004) who models the problem as a minimisation one (i.e. minimise the gap between a subset of weights and the constraint): given a binary vector  $\mathbf{x} \in \{0, 1\}^n$ , the fitness of  $\mathbf{x}$  is defined as

$$f(\mathbf{x}) = g(\mathbf{x}) \left( c - \sum_{i=1}^n x_i w_i \right) + (1 - g(\mathbf{x})) \sum_{i=1}^n x_i w_i \tag{7}$$

where  $g(\mathbf{x}) = 1$  if the solution is feasible and 0 otherwise. It is important to note that other techniques such as repair algorithms (or even other penalty functions) will give rise to different outcomes in the dynamic variant of the subset sum problem as discussed in Sect. 5.1.

#### 5 The dynamic subset sum problem

The objective of this work is to identify how the trajectory of instances through the state space  $S$  is reflected by the movement of  $\mathbf{x}^*$  through  $X$ . In Jin and Branke (2005), Branke states:

For most real-world problems, however, it is hoped that changes are rather smooth and, thus, a lot can be gained by transferring knowledge from the past (Jin and Branke 2005, p. 310)

The majority of studies assumes that there is a significant correlation between the position of the global optimum before and after a change. However, evidence of such a relationship is generally missing. Intuition suggests that changes in the parameter space may have an arbitrary impact

on the position of the global optimum and it is indeed easy to show how small changes may have an arbitrarily large impact on the position of the global optimum: given  $\mathbf{w} = (4, 5, 6, 16)$  and  $c = 17$ , for instance, the global optimum is 1,110. If the value of  $c$  changes to 16, the global optimum becomes 0001. Similarly, if the integers change to  $(4, 5, 7, 17)$ , the optimum also changes by the maximum possible amount. Increasing the value of the fourth item arbitrarily; on the other hand, does not affect the global optimum at all. These examples illustrate that changes in the parameters may not be correlated to the movement of the global optimum and similar observations have been made elsewhere (Yamasaki et al. 2002). However, it is equally trivial to present counter examples and it is thus of interest to determine whether a correlation exists on an average and to determine the properties that are responsible for this.

To test this property, it is necessary to define a metric space  $S$  which depends on the definition of a problem instance as well as a distance measure. The space of instances  $S$  for the subset sum problem as defined in Eq. 6 is of size  $m^n \cdot mn$ . However,  $S$  contains many instances that are trivial in the sense that the constraint is less than the smallest weight or larger than the sum of all weights. These instances make up a significant proportion of  $S$  and hence we redefine each instance, replacing the absolute constraint  $c$  with the constraint-to-weight ratio  $\delta = c/\omega$  where  $\omega = \sum_{i=1}^n w_i$ . An instance of the subset sum problem is then specified by  $(\mathbf{w}, \delta)$  eliminating the presence of trivial instances and the space  $S$  may then be visualised as a two-dimensional space where one dimension corresponds to the values of  $\mathbf{w}$  and the other to values of  $\delta$ .

The distances between successive global optima is readily specified by the Hamming distance. Defining a distance measure  $d_I(I_a, I_b)$  for two instances  $a$  and  $b$ , on the other hand, is far more difficult. Each instance is defined as the pair  $(\mathbf{w}, \delta)$  and while it is possible to define a distance measure for either component, it is difficult to define a joint distance measure for both without imposing a value on the importance of either component. We thus treat each component separately and concentrate on changes to  $\mathbf{w}$  only and changes to  $\delta$  only. The distances may then be defined simply as:

$$d_{\mathbf{w}}(\mathbf{w}_a, \mathbf{w}_b) = \sum_{i=1}^n |w_{ai} - w_{bi}| \quad (8)$$

and

$$d_{\delta}(\delta_a, \delta_b) = |\delta_a - \delta_b| \quad (9)$$

Finally, as mentioned in the previous section, it should be noted that the distances between instances and global optima strictly depend on the definition of the problem as well as the representation and genetic operators used by the algorithm to solve the problem. If, for instance, a different

representation were to be used (in place of binary), the distances between successive global optima would be affected. It is, thus, necessary to briefly discuss the role of the representation in the combinatorial domain which is the subject of the next section.

## 5.1 Combinatorial fitness landscapes

Any continuous function naturally specifies a fitness landscape given a range of values in the continuous domain. In the combinatorial case, a fitness landscape may only be specified with regards to a specific algorithm which determines both the representation of solutions as well as their neighbourhood relation. A fitness landscape in the combinatorial domain is specified by the tuple  $\mathcal{L} = (X, f, d)$  where  $X$  is the search space (set of solutions),  $f$  the objective (fitness) function and  $d$  the distance function for the points in  $X$ . The distance measure  $d$ , which determines the proximity of points in the  $n$ -dimensional search space, is often defined as the smallest possible distance (i.e. 1 bit in the binary case), although in practical terms, this depends on the variation (genetic) operators  $\mu$  of the algorithm. It is thus evident that the representation used by an algorithm to encode solutions to the problem is crucial to the structure of the resulting fitness landscape. The XoR benchmark circumvents this algorithm-dependency to allow precise control over the magnitude of change by preserving the underlying structure of the search space.

Rothlauf (2002) provides a comprehensive description of representations and defines two functions that map from genotypes to phenotypes and from phenotypes to fitness values. If a representation is used by an algorithm, the genotype (representation) is mapped onto a phenotype (actual solution) by the function  $f_g : X_g \rightarrow X_p$ . The phenotype is subsequently mapped onto a real value by the function  $f_p : X_p \rightarrow \mathbb{R}$ . The objective function may thus be specified as the composite mapping of  $f_p$  and  $f_g : f = f_p \circ f_g = f_p(f_g(\mathbf{x}_g))$ . Thus, in practical terms, a combinatorial fitness landscape often corresponds to the tuple  $\mathcal{L} = (X, f_p \circ f_g, \mu)$  and it follows that the dynamics, acting on the parameters of the problem, are “observed” by the algorithm through the composite mapping  $f_p \circ f_g$  where the spatial arrangement of the search space depends on  $\mu$ . The representation thus not only determines the structure of the fitness landscape, but may also have a significant impact on the transition from one landscape to another. In other words, the representation influences the relationship between the actual dynamics (in the parameter space) and the observed dynamics (in the fitness landscape). The same reasoning applies to other aspects (e.g. penalty function) and hence the results presented in subsequent sections have to be viewed with regard to the specific framework used.

**Table 1** Expected distances (to one decimal place) for elements  $x \in X$  with  $|x|_1 = a$  (left-most column) to elements with  $|x|_1 = b$  (top row) for  $n = 12$ .

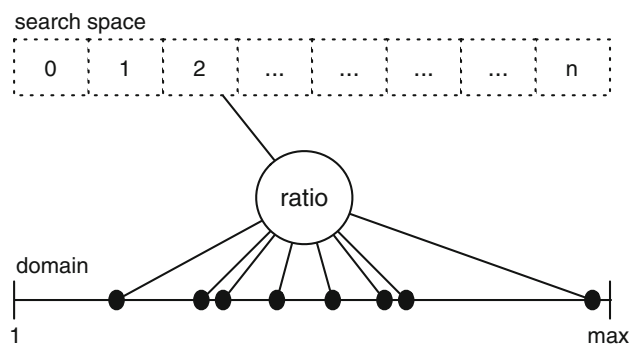
$ x _1$	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0	12.0
1	1.0	2.0	2.7	3.5	4.3	5.2	6.0	6.8	7.7	8.5	9.3	10.2	11.0
2	2.0	2.7	3.4	4.0	4.7	5.3	6.0	6.7	7.3	8.0	8.7	9.3	10.0
3	3.0	3.5	4.0	4.5	5.0	5.5	6.0	6.5	7.0	7.5	8.0	8.5	9.0
4	4.0	4.3	4.7	5.0	5.3	5.7	6.0	6.3	6.7	7.0	7.3	7.7	8.0
5	5.0	5.2	5.3	5.5	5.7	5.8	6.0	6.2	6.3	6.5	6.7	6.8	7.0
6	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0
7	7.0	6.8	6.7	6.5	6.3	6.2	6.0	5.8	5.7	5.5	5.3	5.2	5.0
8	8.0	7.7	7.3	7.0	6.7	6.3	6.0	5.7	5.3	5.0	4.7	4.3	4.0
9	9.0	8.5	8.0	7.5	7.0	6.5	6.0	5.5	5.0	4.5	4.0	3.5	3.0
10	10.0	9.3	8.7	8.0	7.3	6.7	6.0	5.3	4.7	4.0	3.4	2.7	2.0
11	11.0	10.2	9.3	8.5	7.7	6.8	6.0	5.2	4.3	3.5	2.7	2.0	1.0
12	12.0	11.0	10.0	9.0	8.0	7.0	6.0	5.0	4.0	3.0	2.0	1.0	0.0

The diagonal (top-left to bottom-right) shows the expected distance for elements within each group  
 The groups 1 and 12 only contain a single element

### 5.2 Search space analysis

In this work, we are interested in the correlation between the distances of successive instances and their global optima. For the sake of analysis, we divide the search space  $X$  into  $n + 1$  groups such that group  $i$  contains all bit strings with  $|x|_1 = i$  where  $|x|_1 = \sum_{k=1}^n x_k$ . The size of each group is given by the binomial distribution with respect to  $n$  and  $1/2$ . Table 1 shows the expected Hamming distances for elements from group  $i$  to group  $j$  (inter-distances) as well as the expected distances for elements from the same group (i.e.  $i = j$ ; intra-distances) for  $n = 12$ .

This division of the search space may be utilised in the analysis of the properties of the dynamic subset sum problem. Figure 2 illustrates the relationship between the weights  $w$ , the ratio  $\delta$  and the position of the resulting global optimum  $x^*$  in the partitioned search space. The partition of the search space reveals several trends: there is a correlation between the inter-distances and the intra-distances except for the case of  $n/2$ . The correlation is more pronounced for values of  $|x|_1$  close to 0 and 12. Interestingly, the correlation is positive in one direction and negative in the other. For instance, consider the case of  $|x|_1 = 3$ . The inter-distances to groups 2 and 1 are 4 and 3.5, respectively (i.e. the inter-distances and intra-distances are negatively correlated). On the other hand, the inter-distances between group 3 and groups 4 and 5 are 5 and 5.5, respectively (i.e. the inter-distances are positively correlated to the intra-distances). This highlights an important property, namely that the direction of change, an aspect not generally considered in the literature, may have a significant impact on the observed dynamics.



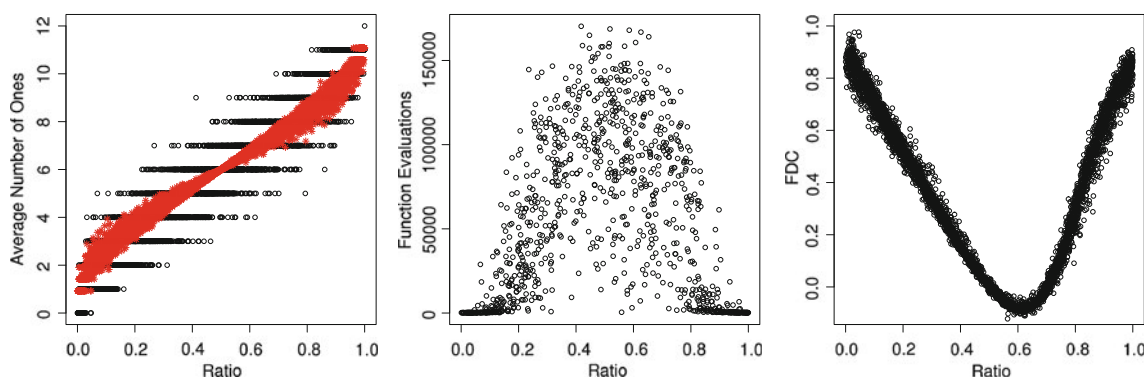
**Fig. 2** The relationship between the weights  $w$ , the ratio  $\delta$  and the position of the resulting global optimum  $x^*$  in the partitioned search space

## 6 Experimental studies

The results are presented in three parts: first, a general analysis of the subset sum problem highlights important properties that affect the structure of the fitness landscape and the position of the global optimum. These properties are subsequently used to first analyse the impact of a dynamic ratio  $\delta$  (Sect. 6.2) followed by an analysis of a dynamic weight vector  $w$  (Sect. 6.3).

### 6.1 Preliminary analysis of the subset sum problem

To better understand the properties of the dynamic subset sum problem, it is useful to first identify properties of the problem that have a significant impact on the structure of the search space. For this and all remaining experiments, we generated 5,000 sets of integers of size  $n = 12$  where



**Fig. 3** Preliminary analysis of the subset sum problem: (left) the constraint-to-weight ratio has a significant impact on the expected size of the optimal solution both on the global optima (horizontal lines) as well as the elements located at the boundary of the feasible space (fuzzy diagonal). Instances with a ratio close to 0.5 are (center), on an average, more difficult to solve for a simple genetic algorithm. The ratio (right) has a direct impact on the fitness distance correlation of the instance

**Table 2** Probability that the global optimum has a specific value for  $|x^*_1|_1$  (top row) given ratios  $\delta \in \{1/12, \dots, 11/12\}$  (left-most column)

$\delta/ x _1$	1	2	3	4	5	6	7	8	9	10	11	<b>E</b>
1/12	0.251	0.489	0.203	0.055	0.002							2.07
2/12	0.014	0.291	0.505	0.151	0.034	0.005						2.92
3/12		0.087	0.341	0.383	0.160	0.029						3.70
4/12		0.002	0.096	0.434	0.336	0.123	0.006	0.002				4.51
5/12				0.171	0.442	0.323	0.060	0.004				5.28
6/12				0.022	0.235	0.477	0.213	0.051	0.002			6.04
7/12				0.007	0.081	0.275	0.445	0.188	0.004			6.74
8/12					0.012	0.096	0.372	0.412	0.101	0.007		7.52
9/12					0.002	0.026	0.142	0.433	0.333	0.063		8.26
10/12						0.002	0.038	0.187	0.459	0.312	0.002	9.05
11/12							0.004	0.038	0.174	0.530	0.253	9.99

The right-most column shows the expected value for  $|x^*_1|_1$  for each of the ratios considered

each element is generated uniformly at random from the interval  $[1, 4096]$ . We concentrate on rather small instances with  $n = 12$  to allow for an exhaustive analysis of the search space (which contains  $2^{12}$  points). For each set, we consider ratios  $\delta \in \{1/12, 2/12, \dots, 11/12\}$ . This gives rise to a total of 55,000 subset sum instances and for the sake of simplicity we ensure, by means of exhaustive sampling, that each instance has exactly a single global optimum (if a set of integers generates multiple global optima for any given ratio, the set is replaced with another randomly generate set of integers).

As shown in Fig. 3, one of the most prominent attributes of a subset sum instance is the ratio  $\delta$  which has a direct impact on the number of elements that are included in the global optimum (i.e. the value of  $|x|_1$ ). In fact, if the variance of  $w$  is 0, then the expected value of  $|x|_1$  is  $\delta n$ . The probability distribution for the general case (i.e. non-zero variance) is shown in Table 2. The spread is due to the variance of the integers contained within the set as large(r) weights

influence the value of  $c$ , allowing for multiple small(er) weights to form a subset that satisfies the constraint.

Figure 3 also shows how the ratio affects the difficulty of an instance: using a simple generational genetic algorithm (GA) with population size 50, elitism of 1, uniform crossover (crossover rate 0.8), bit-wise mutation (mutation probability  $1/n$ ) and tournament selection, the number of function evaluations required to locate the global optimum increases as the ratio of an instance approaches 0.5. Finally, the third observation is the relationship between the ratio and the fitness distance correlation (FDC) of an instance. The FDC is a measure of the correlation of a search point's quality (fitness) and its distance to the nearest global optimum. In Rohlfshagen and Yao (2008), we used a GA to evolve subset sum instances with specific FDCs and showed how differences between instances are reflected by the movement of their global optima only in cases where the FDC is close to 1. These cases correspond to instances with ratios close to 0 or 1.

**Table 3** The expected group membership (top row) of a point  $\mathbf{x}$  for a change in ratio of 1/12 (left-most column): if, for instance, a point is situated in group 1 given a ratio of 1/12, the expected group membership of that point for a new ratio of  $1/12 + 1/12 = 2/12$  is 2.71

$\Delta\delta/ \mathbf{x} _1$	1	2	3	4	5	6	7	8	9	10	11
1/12–2/12	2.71	2.87	3.15	3.39	3.95						
2/12–3/12	4.05	3.58	3.69	3.88	4.11	4.54					
3/12–4/12		4.50	4.40	4.49	4.61	4.72	4.33				
4/12–5/12		5.40	5.29	5.26	5.26	5.30	5.17	6.00			
5/12–6/12			6.16	6.00	6.01	6.00	5.99	5.82	7.00		
6/12–7/12			6.75	6.62	6.72	6.76	6.77	6.73	6.25		
7/12–8/12				7.19	7.42	7.53	7.56	7.49	7.38		
8/12–9/12					7.68	8.11	8.27	8.31	8.36	7.57	
9/12–10/12					7.88	8.74	8.96	9.09	9.20	9.14	
10/12–11/12						9.22	9.35	9.81	10.02	10.15	9.82

This data should be compared to the right-most column of Table 2 which shows the expected group membership over all instances

**Table 4** The expected Hamming distances between successive global optima for all combinations of ratios considered

$\delta$	1/12	2/12	3/12	4/12	5/12	6/12	7/12	8/12	9/12	10/12	11/12
1/12		3.33	4.06	4.70	5.38	6.03	6.63	7.28	7.89	8.59	<b>9.02</b>
2/12	3.33		4.37	4.96	5.43	5.94	6.56	6.99	7.49	<b>8.34</b>	8.50
3/12	4.06	4.37		5.17	5.57	6.00	6.35	6.72	<b>8.17</b>	7.51	7.88
4/12	4.70	4.96	5.17		5.68	5.90	6.21	<b>8.37</b>	6.77	7.02	7.24
5/12	5.38	5.43	5.57	5.68		5.94	<b>8.65</b>	6.21	6.41	6.55	6.64
6/12	6.03	5.94	6.00	5.90	5.94		5.96	5.92	5.98	5.96	5.99
7/12	6.63	6.56	6.35	6.21	<b>8.65</b>	5.96		5.73	5.56	5.43	5.34
8/12	7.28	6.99	6.72	<b>8.37</b>	6.21	5.92	5.73		5.15	4.95	4.73
9/12	7.89	7.49	<b>8.17</b>	6.77	6.41	5.98	5.56	5.15		4.41	4.10
10/12	8.59	<b>8.34</b>	7.51	7.02	6.55	5.96	5.43	4.95	4.41		3.40
11/12	<b>9.02</b>	8.50	7.88	7.24	6.64	5.99	5.34	4.73	4.10	3.40	

### 6.2 Dynamic $\delta$

It has been shown in the previous section that the ratio has a direct impact on the position of the global optimum in the partitioned search space. The second factor that influences the group membership of  $\mathbf{x}^*$  is the variance of  $\mathbf{w}$ : the higher the variance, the further away the global optimum from the group  $[\delta n]$ . This is an important property as it implies that the position of  $\mathbf{x}^*$  for  $\delta(T + 1)$  is not independent of the position of  $\mathbf{x}^*$  for  $\delta(T)$ . In other words, if the ratio changes from 1/12 to 2/12, the probability that  $\mathbf{x}^*(T + 1)$  belongs to a certain group depends implicitly on the group membership of  $\mathbf{x}^*(T)$  and Table 3 confirms this: it is evident from the data that the expected group membership at time  $T + 1$ , given an increase in ratio of 1/12, varies depending on the group membership at time  $T$ . This implies that the impact of the dynamics depends on the position in the space  $S$  at time  $T$ .

Table 4 shows the expected distances between successive global optima for all possible transitions of ratios. The differences in ratios are positively correlated to the distances between optima in cases where the ratio is close to 0 or 1. Furthermore, in cases where two ratios add up to 1 (diagonal, shown in bold), the distance tends to deviate from the trend as in these cases, it is likely that global optima are further apart: If one subset satisfies the constraint imposed by ratio  $\delta$ , the remaining numbers satisfy the constraint imposed by ratio  $1 - \delta$  with high probability. If none of the elements from either set form a subset of the other, then the global optima are the maximum possible distance apart.

Similar to the observation made in the previous section, the correlation may be positive or negative depending on the change in ratio. The data, thus shows that there are positive and negative correlations as well as a lack of correlation altogether. Nevertheless looking at the overall

expected distances for a change in  $\delta$  across all ratios, we find a positive trend of

4.91, 5.52, 5.59, 6.25, 6.44, 7.09, 7.38, 8.04, 8.54, 9.02

for respective distances of  $1/12, 2/12, \dots, 11/12$ .

### 6.3 Dynamic $\mathbf{w}$

The previous section has shown that there are cases where dependencies between successive global optima exist. This section looks at the dynamic component  $\mathbf{w}$ . The distance between two vectors  $\mathbf{w}$  has been defined in Eq. 8 and given that every element in  $\mathbf{w}$  is within the range  $[1, 4,096]$ , the maximally different instance that may be generated from any given instance in  $S$  has a distance of  $n \cdot 2,048$  (although greater distances of up to exist). We determined that the distances between the members of the sample of 5,000 instances used in this study range from 2,228 to 34,886. We thus divide the range  $34,886 - 2,228 = 32,658$  into 10 intervals which correspond to different magnitudes of change (i.e. the first interval corresponds to a small magnitude of change, etc). We then group all unique pairs of instances in the test set according to their distance  $d_w$ . If, for example, two instances have a difference of  $d_w = 6,000$ , they are classified as having a distance 2 ( $3,265.8 < 6,000 < 6,531.6$ ). We then compute the distance between their global optima to establish whether or not a correlation exists.

The resulting data is shown in Table 5 and it is evident that, as before, there is a positive correlation between the actual and observed magnitude of change for instances with a ratio close to 0 and 1. This correlation weakens as the ratio approaches 6/12. This data are in agreement with the study carried out by Rohlfshagen and Yao (2008). These results may be explained by what we call the permutation effect: the weights of the subset sum problem are specified as a set and their order is irrelevant. The commonly used binary encoding, on the other hand, makes use of a fixed and pre-determined indexing and hence implements the problem as a vector. This implies that identical cases of the subset sum problem may have different solutions. Given a value of  $c = 10$  and  $W = \{1, 2, 4, 6\}$  for instance, the order of weights determines the global optimum: 0011 if  $\mathbf{w} = (1, 2, 4, 6)$  and 1,100 if  $\mathbf{w} = (6, 4, 2, 1)$ . It is thus possible to have a Hamming distance between global optima of up to  $2 \cdot \min\{|\mathbf{x}^*|_1, n - |\mathbf{x}^*|_1\}$  by simply re-arranging the indices<sup>4</sup> and this is maximised when the expected value of  $|\mathbf{x}|_1$  is  $n/2$  (i.e. when  $\delta$  is close to 6/12). Similarly, looking at a ratio of 1/12, for example, instances

**Table 5** The expected Hamming distance between successive global optima for different distances (magnitudes of change) between instances (top row; see text) and ratios

	1	2	3	4	5	6	7	8	9	10
1/12	2.61	2.74	2.84	3.21	3.45	3.72	4.20	4.67	5.14	5.98
2/12	3.87	3.97	4.03	4.21	4.31	4.54	4.87	5.08	5.26	4.98
3/12	4.83	4.86	4.87	5.07	5.16	4.89	5.25	5.29	5.62	5.83
4/12	5.50	5.46	5.49	5.45	5.48	5.65	5.66	5.67	5.75	5.48
5/12	5.80	5.87	5.91	5.91	5.90	5.95	5.92	5.93	5.97	5.92
6/12	5.98	6.00	6.01	5.97	5.96	6.00	5.97	5.99	6.04	5.91
7/12	5.86	5.82	5.89	6.00	6.01	5.99	5.94	5.96	5.96	5.83
8/12	5.46	5.48	5.55	5.60	5.71	5.73	5.64	5.71	5.76	5.92
9/12	4.78	4.87	4.84	4.81	4.88	4.91	5.27	5.40	5.62	5.72
10/12	4.00	3.97	3.98	4.14	4.28	4.35	4.50	4.85	5.01	5.08
11/12	2.76	2.77	2.76	2.74	2.94	3.38	4.04	4.04	4.55	5.38
E	4.68	4.71	4.74	4.83	4.92	5.01	5.21	5.33	5.52	5.64

with a high variance in  $\mathbf{w}$  are located in groups with a higher number of ones than those with a lower variance. The expected distance amongst elements in such groups is generally higher (see Table 1) and since the higher variance implies that permutations yield higher differences when compared to one another, there is a positive correlation between the two. This reasoning is confirmed by repeating the experiments with a different distance measure for the instances. We tried three different measures: in the first case, the vectors are sorted before the distance is measured according to Eq. 8. In the second and third case, the Hausdorff distance and modified Hausdorff distance for sets was used. In these cases, the correlation evident in Table 5 disappears. This underlines the importance of the objective function and representation used to model the problem of interest.

## 7 Conclusions

In this paper, we have extended our previous work (Rohlfshagen and Yao 2008) on the dynamic subset sum problem to gain a better understanding of how the dynamics, affecting the parameters of the problem, are materialised in the fitness landscape (the movement of the global optimum). Following a review of the field, we proposed an extended definition for the most commonly considered class of DOPs with clear emphasis on the differences between a problem and instances of the problem (parameters). The majority of the paper focused on the dynamic variant of the subset sum problem. We focused on binary encodings, a natural choice for the subset sum problem, and made use of a penalty function to deal with the constraints of the problem. We first analysed the

<sup>4</sup> This approach has actually been suggested to create dynamic benchmark problems in the combinatorial domain with controllable distances; see (Younes et al. 2005).

stationary subset sum problem and then examined how the actual magnitude of change (changes to the parameters) correlates to the observed magnitude of change (distances between successive global optima). The dynamics were divided into two categories: those affecting only the constraint  $c$  (here modelled as the constraint-to-weight ratio  $\delta$ ) and those affecting the weight vector  $w$ . It has been shown that the rate of change of either  $\delta$  or  $w$  is reflected in the rate of change of the global optimum if one of the instances involved has a ratio close to the minimum or maximum possible value (i.e. 0 or 1). Furthermore, the direction of change is also important as it determines whether this correlation is positive or negative. When considering all possible cases (the expected behaviour), it has been shown that there is a positive correlation between the distances amongst instances in  $S$  and the distances between successive global optima.

These results have several important implications on the current state of research in the field of evolutionary dynamic optimisation. It has been shown that the observed dynamics differ significantly depending on the current position in the state space  $S$  (e.g. a certain value of  $\delta$ ). Furthermore, the direction of change (e.g. increase/decrease  $\delta$ ) is another significant factor that determines the trajectory of  $x^*$  in  $X$ . These properties are usually ignored in the literature. Nevertheless, it has also been shown that the expected case, averaged over all possible scenarios, shows a positive correlation between the actual and observed magnitude of change. Intuitively, one may use these results to argue that local properties are not as relevant as global properties. However, this is not the case, as the correlation breaks down particularly in those cases that are the most difficult to solve (for a simple genetic algorithm). This seems to imply that in the most difficult cases, it might not be possible to transfer knowledge from one instance in time to the next. This underlines the importance to clearly specify the type of trajectories through  $S$  that are of interest to a particular study.

There are many extensions to the work presented here that could be addressed in the near future to advance our understanding of dynamic combinatorial optimisation problems. In particular, the role of the representation has been discussed but no experimental work has been carried out to extend the existing work on this subject (Ursem et al. 2002; Branke et al. 2005, 2006). In addition, it would also be interesting to investigate the impact of different penalty functions, especially those that allow the algorithm to explore the infeasible region. Finally, the instances considered here are of small size ( $n = 12$ ) and it is not guaranteed that results scale. In the near future, we intend to look at larger, more realistic instances, using some state-of-the-art deterministic algorithm to obtain (near-)optimal solutions.

**Acknowledgments** This work was supported by EPSRC Grant no. EP/E058884/1 entitled “Evolutionary Algorithms for Dynamic Optimisation Problems: Design, Analysis and Applications”. The authors are grateful for useful discussions with Trung Thanh Nguyen regarding this work.

## References

- Bosman PAN (2005) Learning, anticipation and time-deception in evolutionary online dynamic optimization. In: Proceedings of the 2005 workshops on genetic and evolutionary computation, pp 39–47
- Bosman PAN, Poutrè HL (2007) Learning and anticipation in online dynamic optimization with evolutionary algorithms: the stochastic case. In: Proceedings of the 2007 genetic and evolutionary computation conference, pp 1165–1172
- Branke J (1999) Memory enhanced evolutionary algorithms for changing optimization problems. In: Proceedings of the 1999 IEEE congress on evolutionary computation, vol 3. IEEE, pp 1875–1882
- Branke J (2002) Evolutionary optimization in dynamic environments. Kluwer, Dordrecht
- Branke J, Kau T, Schmidt C, Schmeck H (2000) A multi-population approach to dynamic optimization problems. *Adapt Comput Des Manuf*
- Branke J, Salihoglu E, Uyar S (2005) Towards an analysis of dynamic environments. In: Hans-Georg Beyer et al (eds) Genetic and evolutionary computation conference. ACM, pp 1433–1439
- Branke J, Orbayi M, Uyar S (2006) The role of representations in dynamic knapsack problem. In: Rothlauf F (ed) *EvoWorkshops 2006*. Springer, Berlin, pp 764–775
- Cobb HG (1990) An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependant nonstationary environments. Technical Report, Naval Research Laboratory, Washington, USA
- Garey MR, Johnson DS (1979) *Computers and intractability: a guide to the theory of NP-completeness*. W. H. Freeman and Company, San Francisco
- Goldberg DE, Smith RE (1987) Nonstationary function optimization using genetic algorithms with dominance and diploidy. In: Grefenstette JJ (ed) *Second international conference on genetic algorithms*. Lawrence Erlbaum Associates, New Jersey, pp 59–68
- Grefenstette JJ (1992) Genetic algorithms for changing environments. In: Manner R, Manderick B (eds) *Proceedings of the second international conference on parallel problem solving from nature 2*. Elsevier, Amsterdam, pp 137–144
- Hadad BS, Eick CF (1997) Supporting polyploidy in genetic algorithms using dominance vectors. In: Proceedings of the sixth international conference on evolutionary programming, vol 1213. Springer, Berlin, pp 223–234
- Jin Y, Branke J (2005) Evolutionary optimization in uncertain environment—a survey. *IEEE Trans Evol Comput* 9(3):303–317
- Kellerer H, Pferschy U, Pisinger D (2004) *Knapsack problems*. Springer, Berlin
- Liekens AML (2005) Evolution of finite populations in dynamic environments. PhD thesis, Technische Universiteit Eindhoven
- Morrison RW (2004) *Designing evolutionary algorithms for dynamic environments*. Springer, Berlin
- Morrison RW, DeJong KA (1999) A test problem generator for non-stationary environments. In: *Congress on evolutionary computation*, vol 3. IEEE, pp 2047–2053
- Rohlfshagen P, Yao X (2008) Attributes of dynamic combinatorial optimisation. In: *Lecture notes in computer science*, vol 5361. Springer, Berlin, pp 442–451

- Rohlfshagen P, Lehre PK, Yao X (2009) Dynamic evolutionary optimisation: an analysis of frequency and magnitude of change. In: Proceedings of the 2009 genetic and evolutionary computation conference, pp 1713–1720
- Rossi C, Barrientos A, del Cerro J (2007) Two adaptive mutation operators for optima tracking in dynamic optimization problems with evolution strategies. In: Proceedings of the 9th annual conference on genetic and evolutionary computation, pp 697–704
- Rothlauf F (2002) Representations for genetic and evolutionary algorithms. Springer, Berlin
- Tinos R, Yang S (2007) Continuous dynamic problem generators for evolutionary algorithms. In: Proceedings of the 2007 IEEE congress on evolutionary computation, pp 236–243
- Ursem RK, Krink T, Jensen MT, Michalewicz Z (2002) Analysis and modeling of control tasks in dynamic systems. *IEEE Trans Evol Comput* 6(4):378–389
- van Hemert JI, Van Hoyweghen C, Lukschandl E, Verbeeck K (2001) A “futurist” approach to dynamic environments. In: Branke J, Bäck T (eds) Proceedings of the workshop on evolutionary algorithms for dynamic optimization problems at the genetic and evolutionary computation conference, pp 35–38
- Weicker K (2003) Evolutionary algorithms and dynamic optimization problems. Der Andere Verlag
- Wilke CO (1999) Evolutionary dynamics in time-dependent environments. PhD thesis, Ruhr-Universität Bochum
- Wolpert DH, MacReady WG (1997) No free lunch theorems for optimization. *IEEE Trans Evol Comput* 1(1):67–82
- Wong RL (2004) A genetic algorithm for subset sum problem. *Neurocomputing* 57:463–468
- Yamasaki K, Kitakazeand K, Sekiguchi M (2002) Dynamic optimization by evolutionary algorithms applied to financial time series. In: Proceedings of the 2002 Congress on evolutionary computation, pp 2017–2022
- Yang S (2003) Non-stationary problem optimization using the primal-dual genetic algorithms. In: Sarker R, Reynolds R, Abbass H, Tan K-C, McKay R, Essam D, Gedeon T (eds) Proceedings of the 2003 IEEE congress on evolutionary computation, vol 3. pp 2246–2253
- Yang S (2005) Memory-based immigrants for genetic algorithms in dynamic environments. In: Proceedings of the 2005 genetic and evolutionary computation conference, vol 2. ACM Press, pp 1115–1122
- Yang S (2006) Associative memory scheme for genetic algorithms in dynamic environments. In: Applications of evolutionary computing, vol 3907. Lecture Notes in Computer Science, Springer, Berlin, pp 788–799
- Yang S, Ong Y-S, Jin Y (eds) (2007) Evolutionary computation in dynamic and uncertain environments. Springer, Berlin
- Younes A, Calamai P, Basir O (2005) Generalized benchmark generation for dynamic combinatorial problems. In: Proceedings of the 2005 workshops on genetic and evolutionary computation, pp 25–31