Nat Comput (2012) 11:431–448
DOI 10.1007/s11047-011-9296-3

Evolutionary mechanics: new engineering principles for the emergence of flexibility in a dynamic and uncertain world

James M. Whitacre · Philipp Rohlfshagen · Axel Bender · Xin Yao

Published online: 25 November 2011

Abstract Engineered systems are designed to deftly operate under predetermined conditions yet are notoriously fragile when unexpected perturbations arise. In contrast, biological systems operate in a highly flexible manner; learn quickly adequate responses to novel conditions, and evolve new routines and traits to remain competitive under persistent environmental change. A recent theory on the origins of biological flexibility has proposed that degeneracy—the existence of multi-functional components with partially overlapping functions—is a primary determinant of the robustness and adaptability found in evolved systems. While degeneracy's contribution to biological flexibility is well documented, there has been little investigation of degeneracy design principles for achieving flexibility in systems engineering. Actually, the conditions that can lead to degeneracy are routinely eliminated in engineering design. With the planning of transportation vehicle fleets taken as a case study, this article reports evidence that degeneracy improves the robustness and adaptability of a simulated fleet towards unpredicted changes in task requirements without incurring costs to fleet efficiency. We find that degeneracy supports faster rates of design adaptation and ultimately leads to better fleet designs. In investigating the limitations of degeneracy as a design principle, we consider decision-making difficulties that arise from degeneracy's influence on fleet complexity. While global decision-making becomes more challenging, we also find degeneracy accommodates rapid distributed decision-making leading to (near-optimal) robust system performance. Given the range of conditions where favorable short-term and long-term performance outcomes are observed, we propose that degeneracy may fundamentally alter the propensity for adaptation and is useful within different engineering and planning contexts.

Keywords Degeneracy · Evolvability · Robustness · Redundancy · Dynamic optimization · Complex systems engineering · Dynamic capabilities · Strategic planning

J. M. Whitacre (✉) · X. Yao
CERCIA, School of Computer Science, University of Birmingham, Birmingham B15 2TT, UK
e-mail: j.m.whitacre@cs.bham.ac.uk

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

P. Rohlfshagen
School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, UK
e-mail: prohlf@essex.ac.uk

A. Bender
Land Operations Division, Defence Science and Technology Organisation, Edinburgh, SA 5111, Australia
e-mail: axel.bender@dsto.defence.gov.au

1 Introduction

Engineering involves the design and assemblage of elements that work in specific ways to achieve a predictable purpose and function (Fromm 2006; Ottino 2004). Engineering, planning, and science in general have historically taken a reductionist approach to problem solving; aiming to decompose a complicated problem into more manageable and well-defined sub-problems that are largely separable or modular (cf. Sect. 3.1; Minai et al. 2006; Beckerman 2000). A reductionist problem decomposition reduces the degrees of freedom that are considered at any one time. It is methodical, conceptually intuitive, and it can help individuals understand the most relevant determinants of each sub-system's behaviour. If sub-systems truly represent

modular building blocks of the larger system, then there are (by definition) relatively few ways in which the sub-system will interact with its surroundings. This often permits the operating environment to be defined with precision and accuracy. When these conditions are met, engineers and planners have historically been able to systematically design components/subsystems with reliable functioning that translates into reliable performance at the system-level (Calvano and John 2003). The application of reductionist principles also results in a hierarchical decomposition of a system that contributes to system-level transparency and benefits global forms of decision-making, trouble-shooting, planning, and control.

While the reductionist paradigm is logically sound, it is only loosely followed in practice because many of the systems within which engineering (or reengineering) takes place cannot be neatly decomposed. This is partly due to the factors that have shaped these systems over time including distributed decision-making during development, multiple conflicting objectives, bounded rationality, historical contingency (path dependency), and environmental volatility. However, purist views of planning and engineering often prevail when attempting to understand failure in complex systems. When failure occurs, it is common and logical to highlight the precise points where a system has failed with the implicit assumption that performance could have been sustained had a relatively small set of modifications been made during the design stage. Because a narrative description of failure can be achieved through careful investigation, it is often assumed that the precise contingency of failure sufficiently captures the origins of a system's design flaws. That these failures might be symptomatic of wider issues related to innate system properties is sometimes suggested but has been notoriously difficult to address in practice (Bak and Paczuski 1995; Carlson and Doyle 2002; Kauffman 1990).

A further limitation to the classic reductionist paradigm is that it assumes the capacity to anticipate beforehand the conditions a system will experience and the precise manner in which the system should respond to those conditions (Minai et al. 2006), i.e., it assumes a stable environment or it requires precognition (Gribble 2001). While predicting plausible future conditions is often a useful exercise, several factors can limit prediction accuracy and lead to uncertainty (Calvano and John 2003). The origins of this uncertainty are varied, however it is a general rule of thumb that for complex systems operating in a dynamic environment, we are limited in our ability to develop standard operating procedures or contingency plans that can accurately account for the various plausible conditions that we might encounter (Mogul 2006; Polacek and Verma 2009; Lane and Boehm 2007). Thus, it becomes important that a system be adaptable to conditions that were not anticipated in its design stage. From a classic engineering and planning perspective, such a design goal is ambiguous and vague and it may seem that planning for the unexpected is an oxymoron.

In contrast to the traditional engineering approach to problem solving, the creation and maintenance of biological functions occurs in a non-reductionist manner that is exceptionally effective at accommodating and exploiting novel conditions. There are a number of systems engineering and organization science studies that have proposed ways to exploit properties associated with biological robustness and adaptability (Sheard and Mostashari 2008; Grisogono and Spaans 2008; Ilachinski 1996; Ryan 2007). While a better appreciation of these biological properties has been useful to decision makers and planners, many of these properties—including loose coupling, distributed robustness, and adaptability—are hard to apply because there is little understanding of their origins or mechanistic basis and few guidelines for their realization.

One of the key contributions of the work presented in this article is the description and validation of design principles that can be defined at a component level and that can lead to the emergence of system properties such as flexibility, distributed robustness and adaptability. By distilling out basic working principles that lead to biological robustness and adaptability, we will gain useful insights that can be used within engineering and planning contexts. In Sect. 2 we focus particularly on one design principle—degeneracy—that is suspected to play a fundamental role in assisting complex biological systems to cope with environmental novelty (Whitacre 2010a; Whitacre and Bender 2010a, b).

Because degeneracy design principles correspond with measurable properties that can be realized in an engineering context, we are able to validate our most important claims through simulations involving engineered systems that undergo design optimization. In Sect. 3, we motivate a strategic planning problem involving the development of a military vehicle fleet capability, and introduce a simulation environment for evaluating fleet robustness across a range of future plausible operating scenarios. We use evolutionary computation to simulate incremental adaptations that are aimed at improving the robustness of a fleet's design. In Sect. 4, we explore several robustness and adaptability properties of the fleets as they are exposed to different classes of environmental change and we find evidence supporting the hypothesis that degeneracy provides broad advantages within some classes of environments; most notably environments that are complex and occasionally unpredictable. Section 5 comments on the potential relevance of these findings and we make concluding remarks in Sect. 6.

Given the multi-disciplinary nature of our topic, the introduction in Sect. 2 relies on abstract terminology that

can be universally understood by biologists, engineers, planners, and decision-makers. Later on in the Case study and Discussion sections (Sects. 3 and 5), we establish links between these abstract concepts and their counterparts within particular disciplines.

2 Lessons from biology

2.1 Robustness

2.1.1 Functional redundancy

Engineered systems typically comprise elements designed for a single and well-specified purpose. However in biological systems there is no pre-assignment of a one-to-one mapping between elements and traits. Instead, at almost every scale in biology, structurally distinct elements (e.g., genes, proteins, complexes, pathways, cells) can be found that are interchangeable with one another or are otherwise compensatory in their contributions to system functions (Whitacre 2010a; Whitacre and Bender 2010a; Edelman and Gally 2001; Kurakin 2009). This many-to-one mapping between components and functions is referred to as *functional redundancy*.

Pure redundancy is a special case of many-to-one mapping and it is a commonly utilized design tool for improving the robustness of engineered systems. In particular, if there are many copies of an element that perform a particular service then the loss of one element can be compensated for by others; as can variations in the demands for that service. However, maintaining diversity amongst functionally similar elements can lead to additional types of stability. If elements are somewhat different but partially overlap in the functions they perform, they are likely to exhibit different vulnerabilities: a perturbation or attack on the system is less likely to present a threat to all elements at once. Alternatively, we might say that a system gains versatility in how a function can be performed because functionally redundant elements enhance the diversity of conditions under which a particular demand can be satisfied (Whitacre and Bender 2010a; Carpenter et al. 2001; Holling 1996).

2.1.2 Functional plasticity

Whether elements are identical (i.e., *purely redundant*) or only functionally redundant, the buffering just described always requires an excess of resources, and this is often viewed in engineering as a necessary but costly source of inefficiency (Carlson and Doyle 2002; Csete and Doyle 2002; Stelling et al. 2004). What is less appreciated however is that simple trade-offs between robustness and efficiency do not necessarily arise in biological systems. Not only are different components able to perform the same function (a many-to-one mapping), many of these components are also multi-functional (one-to-many mapping), with the function performed depending on the context; a behavior known as *functional plasticity* (Whitacre 2010a; Atamas and Bell 2009; Batada et al. 2007; Kurakin 2007). Functionally plastic elements that are excluded from participating in a particular function (e.g., due to demands for that service already being met) will switch to other functions (Kurakin 2010). Functional plasticity thus alters the tradeoff between efficiency and robustness because excess resources are shared across multiple tasks.

2.1.3 Degeneracy

Functional plasticity and functional redundancy are observed at all scales in biology and we have described simple and generic situations where these properties contribute to trait stability through local compensatory effects. In biological systems, it is common to observe functionally plastic components that appear functionally redundant in some contexts but functionally distinct in other contexts. The observation of both functional redundancy and functional diversity within the same components is referred to as degeneracy. In the literature, there is an extensive list of documented cases where degeneracy has been found to promote trait stability (Whitacre 2010a; Whitacre and Bender 2010a, b; Edelman and Gally 2001).

Degeneracy has also been shown to create emergent system properties that further enhance trait stability through *distributed compensatory effects*. The origins of this additional and less intuitive form of robustness have been described in the *networked buffering hypothesis* (Whitacre and Bender 2010a). Using genome:proteome simulations, Whitacre and Bender found evidence that networked buffering can roughly double the overall robustness potential of a system for each of the perturbation classes tested. An important conclusion from those studies is that even small amounts of excess functional resources can have a multiplicative effect on system-level flexibility and robustness when degeneracy is prevalent in a system (Whitacre and Bender 2010a; b). This is of considerable relevance to this study because these distributed forms of robustness were also found to substantially enhance a system's ability to adapt to novel conditions.

2.2 Adaptation

2.2.1 Accessing novelty

In both biology and engineering, the discovery of an improved component design necessitates the exploration of

new design variants. Theoretically, degeneracy should enhance a system's access to design novelty because functionally redundant elements retain unique structural characteristics. Structural differences afford a multiplicity of design change options that can be tested, and thus provide more opportunities for new innovative designs to be discovered (Whitacre 2010a; Whitacre and Bender 2010b; Wagner 2008; Kirschner and Gerhart 1998).

2.2.2 *Transforming novelty into innovation*

The availability of distinct design options is an important prerequisite for innovation, however new opportunities often come with new challenges. To transform a local novelty into an exploited innovation, a system must be flexible (e.g., structurally, behaviorally) to accommodate and utilize a modified component effectively. For instance, design changes in a device sometimes require new specifications for interaction, communication, operating conditions, etc. However, a system must accommodate these new conditions without losing other important capabilities or sacrificing the performance of other core system processes. In other words, the propensity to innovate is enhanced in systems that are robust in many of their core functions yet flexible in how those functions are carried out (Kirschner and Gerhart 1998).

2.2.3 *Facilitating unexpected opportunities*

Because design novelty is not predictable, we propose that the flexibility needed to exploit design novelty cannot be entirely pre-specified based on the anticipation of future design changes. To support innovation, it appears that this robust yet flexible behavior would need to be a property that is pervasive throughout the system. Yet within the wider context of a system's development—where each incremental design change involves a boundedly rational and ultimately myopic decision—it also seems that this flexibility must be a property that, while intimately tied to the history of the system's development, can readily emerge without foresight. In contrast, if flexibility only arises in the places where we perceive future need, then a system's ability to accommodate novel design conditions will be limited by our foresight, e.g., our ability to predict plausible future environments.

We have established a theoretical basis explaining how degeneracy can support these prerequisites for adaptation in an efficient manner and we have recently accumulated some evidence from simulation studies that supports these conjectures. For instance, we have found evidence that degeneracy considerably enhances access to design novelty (Whitacre and Bender 2009, 2010b). We have also demonstrated that these novelties can be utilized as positive adaptations (Whitacre 2010b; Whitacre et al. 2010) and can sometimes afford further opportunities when presented with new environments (Whitacre 2010c). In attempting to understand how novelties are transformed into adaptations, we have shown in (Whitacre and Bender 2010a) that high levels of degeneracy lead to the emergence of pervasive flexibility in how a system can organize its resources and thus allows for a decoupling between the preservation of important functions and the accommodation of new ones. These experimental findings support each of our stated conjectures regarding how degeneracy provides a mechanistic basis for robustness and adaptability in biological systems (Whitacre 2010a; Whitacre and Bender 2010b, Edelman and Gally 2001).

2.3 Conflicts between robustness and adaptability

The requisite conditions that we have outlined above suggest a relationship between robustness and adaptability that we argue is rarely observed in engineered systems and may even be entirely absent in systems that are designed entirely using classic reductionist design principles. Some evidence supporting this view is found by comparing the relationships between robustness and adaptability in human-designed evolutionary optimization algorithms with that observed in natural evolution. In agreement with theories on neutral evolution (Kimura 1955, 1983; Ohta 2002), simulations of gene regulatory networks and models of other biological systems have indicated that increasing mutational robustness may increase a system's propensity to adapt (Whitacre and Bender 2010b; Ciliberti et al. 2007). Taking cues from the original theories, research into nature-inspired optimization has looked at designing mutational robustness into an optimization problem's representation and has almost exclusively done so through the introduction of gene redundancy, e.g., polyploidy. The result has been a negligible or sometimes negative influence on the adaptive capabilities of individuals and populations (Banzhaf 1994; Keller and Banzhaf 1996; Knowles and Watson 2003; Vassilev and Miller 2000; Yu and Miller 2001; Smith et al. 2002; Rothlauf and Goldberg 2003). More recently we have investigated simulations where evolution is given the option to create robustness through degeneracy. We have found evidence that this leads to the selective growth of degeneracy, substantial improvements in evolving robustness towards environmental volatility, and better adaptive capabilities in responding to environmental novelty (Whitacre 2010b; Whitacre et al. 2010). Thus, positive relationships between robustness and adaptability have historically been absent in evolutionary simulations but can be established when designed redundancy is replaced by the evolution of degeneracy.

3 Case study

In this article, we consider a simple but well-defined framework that allows us to study fully the mechanistic basis for robustness and adaptability in engineered systems. The case study that we investigate falls into the category of strategic planning problems.

3.1 Motivations for a strategic planning case study

In strategic planning problems, uncertainty arises from the long time horizons in which planning goals are defined and the properties of the systems and environments in which planning takes place. In particular, strategic planning almost invariably deals with the manipulation of multi-scaled complex systems, i.e., systems comprising many heterogeneous elements whose actions and interactions translate into emergent system functions or *capabilities* that are observable over several distinct timescales. A number of well-known contemporary issues are exemplars of strategic planning problems and include financial market regulation, strategies for responding to climate change and natural disasters, assistance to developing countries, defence planning, and strategic planning for nations and large organizations.

Uncertainty within strategic planning can be characterized in different ways (Grisogono and Ryan 2003; Davis et al. 2007; Davis 2002, 2005). For instance, uncertainty can be classified into "known unknowns" (conscious ignorance), "unknown knowns" (tacit knowledge) and "unknown unknowns" (meta ignorance) (Kerwin 1993). Different classes of uncertainty pose different challenges, and traditional approaches to planning have mostly been developed to address conscious ignorance, e.g., through prediction modelling. Tacit knowledge and meta-ignorance cannot be explicitly planned for and require the development of system properties that facilitate adaptation at different levels within a system. For instance, tacit knowledge is predominantly revealed during plan execution and exploiting this knowledge requires plans to be responsive (behaviourally adaptive). On the other hand, meta-ignorance often gives rise to shocks or surprises and knowledge about the "unknown unknowns" only reveals itself "after the fact". Dealing with meta-ignorance therefore requires adaptable design and behaviour that allow for innovation to germinate. Past experiences only provide very limited useful insights.

The existence of meta-ignorance means that we are never entirely certain what future events might be encountered or how current decisions will shape future events or ourselves (Lempert et al. 2003). When we don't know what we don't know, we are prevented from formulating our uncertainty based on the likelihood of possible future states; a common assumption used in robust optimization and control. Even the articulation of plausible states can be difficult due to the emergence of new phenomena. While many aspects of our world remain conserved over time (described in the literature as historical contingency, path dependency, or "frozen accidents" (Kauffman 1990; Kirschner and Gerhart 1998; Gould 1990; Gell-Mann 1995; Crutchfield and Van Nimwegen 2002), other aspects are not predictable due to, for instance, unanticipated regime shifts in the physical environment, paradigm shifts in knowledge and culture, and disruptive technological innovations that introduce new opportunities as well as new challenges. Under these circumstances, desirable solutions or strategies should not only be robust to expected variability; they should also have an innate propensity to adapt to novel conditions. More generally, it is important to understand what design principles, routines, and system properties will determine a system's capacity to function under high variability and facilitate adaptations to unexpected novelty. Because many long-term planning problems are appropriately modeled using semi-autonomous and multi-functional agents (each of which are also complex systems), this class of problems provides a suitable domain for exploring the biologically inspired principles we presented in the previous section.

3.2 Model overview

We now introduce a strategic planning problem involving investment decisions in a fleet of military field vehicles. Over short (operational) timescales, fleets must operate within a volatile environment; having the capacity to rapidly deal with both anticipated and unanticipated missions. Over longer timescales, environments change more dramatically and it is important that new fleet architectures can be developed that can adequately respond to these new realities. Our model is a realistic representation of a strategic resource planning problem. It captures the most important dynamics, namely: changes in the assignment of vehicles to tasks, changes in the composition of vehicles that make up a fleet, changes in the tasks that must be executed during fleet operations, and changes in the composition of tasks at a timescale comparable to the timescale for changes in fleet composition.

The model consists of a *fleet* of vehicles and each vehicle corresponds to a specific *vehicle type* that is capable of carrying out *two* specific *task types*. Each vehicle may devote its resources (e.g., time) to either or both tasks: a vehicle that is capable of performing tasks *A* and *B*, for instance, could devote 30% and 70% of its time to task *A* and *B* respectively. The fleet is exposed to a number of environmental conditions (referred to as *scenarios*), each of which specifies a set of demands for the

specific task types. It responds to each environment by re-distributing its resources using a local decision making process. In particular, each vehicle may distribute its own resources amongst the two tasks it can perform, based on global feedback about the current priority level for fulfilling each task. The problem can thus be stated as finding a suitable fleet composition that allows the fleet to respond to a volatile environment through the redistribution of its resources. In our experiments, this fleet composition (or architecture) is evolved, using a genetic algorithm, to maximize fleet robustness towards this environmental volatility. In order to compare and evaluate different fleet architectures, constraints are imposed on the evolutionary process so that degeneracy is allowed to emerge in some but not all cases.

3.2.1 Vehicle fleet and task types

The problem is formally defined as follows: There are a total of m task types and vehicles are constrained in what types of tasks they can perform. This constraint emulates restrictions that may arise naturally in real-world scenarios where specific tasks have specific hardware requirements that may either be in conflict with one another or simply too expensive to be combined within a single vehicle. In particular, we constrain task-type combinations by defining a symmetric neighborhood for any task type $0 < i \leq m$ as $N_r(i) = \{ j \,||i - j| \leq r, i \neq j\}$ where r is a predefined *radius*, set as $r = 2$ in all experiments. We assume the set of task types to have periodic boundaries such that the neighborhood wraps around the extreme values.

The design of a fleet of n vehicles is characterized by the matrix $V = \{0, 1\}^{n \times m}$: If vehicle i can carry out task-type j, then $V_{ij} = 1$. Otherwise, the matrix entry V_{ij} is 0. As any vehicle is capable of exactly two types of tasks, it follows that $\sum_{j=1}^{m} V_{ij} = 2, \ \forall i$. Furthermore, task combinations in any vehicle i are constrained such that for any j and k for which $V_{ij} = V_{ik} = 1$, it follows that $k \in N_r(j)$. The matrix V thus fully specifies a valid fleet architecture where each vehicle is constrained in (a) the number of task types it may execute and (b) the relationship between the task types (i.e., the radius). While this definition of a vehicle fleet's design may seem highly abstract, it actually is a realistic description of how vehicles are assigned to tasks, e.g., in military vehicle schedule planning problems. The only contrived aspect of our model is the constraint that vehicle types are capable of performing exactly two tasks. As our current research shows, weakening this constraint does not alter the findings and insights presented in the next section.

A second matrix $R = \{x \in \mathbb{Z} : 0 \leq x \leq 10\}^{n \times m}$ is used to indicate the degree to which each vehicle distributes its resources. Each element R_{ij} specifies how much of its resources vehicle i has assigned to task type j. Any entry R_{ij} may be non-zero only if vehicle i is able to perform task j (i.e., $V_{ij} = 1$). We assume that a vehicle allocates all of its resources such that $\sum_{j=1}^{m} R_{ij} = 10, \ \forall i$. The alteration of elements in R thus specifies the distribution of tasks across the fleet of vehicles.

3.2.2 Redundancy and degeneracy

The matrix V specifies the task types each vehicle in the fleet can carry out. We may thus use V to compute a degree of redundancy or degeneracy found in the fleet. Two vehicles are redundant with respect to each other if they are able to carry out precisely the same two task types. If two vehicles have exactly one task type in common, they are considered degenerate. In order to calculate redundancy and degeneracy in a fleet, we consider all $(n^2 - n)/2$ unique pair-wise comparisons of vehicles and count the number of vehicle pairs that have identical task capabilities, partially similar task capabilities and unique capabilities. We then normalize these measurements by the total number of pair-wise comparisons made. More specifically, vehicles i and j are redundant if $V_i \cdot V_j = 2$, degenerate if $V_i \cdot V_j = 1$ and unique with respect to one another if $V_i \cdot V_j = 0$. The fleet's degree of degeneracy is then given by:

$$|V|_{degen} = \frac{1}{n^2 - n} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \left[V_i \cdot V_j = 1 \right] \tag{1}$$

where $[\cdot]$ returns 1 if the containing statement is true and 0 otherwise.

3.2.3 Environments

At any moment in time, t, the fleet is exposed to a set of $\varepsilon = 10$ environmental scenarios $E_t = \left\{ e_1^t, e_2^t, \ldots, e_\varepsilon^t \right\}$ where $0 \leq e_{ij}^t \leq n$ specifies the number of times task type j needs to be executed in scenario e_i at time t. For each scenario e_i^t, the complete satisfaction of task demands requires a fully utilized fleet,[1] i.e., $\sum_{j=1}^{m} e_{ij}^t = 10n, \ \forall i, t$. The scenarios are generated as follows: The first scenario in the set—the "seed scenario", e_{seed}—is created by randomly sampling, with replacement, n tasks from the m task types. The remaining $\varepsilon - 1$ scenarios are generated from the seed using *two* methods as described below. We distinguish between a decomposable case, where correlations are restricted to occur between predetermined pairs of

[1] Assuming the architecture of the fleet is optimal with respect to the environment (i.e., the fleet *can* satisfy all demands given an optimal distribution of resources).

tasks, making the problem separable, and a non-decomposable case where correlations between the variation of any two task types is avoided (Fig. 1c).

3.2.3.1 Decomposable set of scenarios In order to generate the decomposable scenarios, the m task types are randomly grouped into $m/2$ pairings which are place into a set S. A random walk is then used to create new scenarios based on e_{seed}. At each step of the random walk, a single task i is selected and its demand incremented by 1. At the same time, the demand for the corresponding task j, i.e., the task type for which $(i, j) \in S$, is decremented by 1, ensuring that the total number of tasks required by the environment is always constant and equal to $10n$ (Fig. 1d). The correlated increase/decrease is allowed as long as either task demand is within the bounds of 0 and n. The *volatility* of E_t is subsequently defined as the length of the random walk which always starts from e_{seed}.

3.2.3.2 Non-decomposable set of scenarios In the non-decomposable case, each step of the random walk involves an entirely new and randomly selected pair of task types (i.e., there is no predetermined set S). This ensures that the number of correlated task types is minimised, making the problem non-decomposable.

3.2.3.3 Validation scenarios For selected experiments, fleet robustness evolves for one set of environmental scenarios ("training scenarios") and is then further evaluated on a new set of "validation scenarios". The validation sets are generated in the same manner as the scenarios used during evolution but with the following additional consideration: In each validation set, one scenario is selected at random from the set used during evolution and is then used as the seed to generate the new set of scenarios. For validation set 1 ("Validation I"), the generation of the remaining scenarios is further constrained such that variations between the original and validation set remain decomposable, i.e., the same task type correlations (as of set S) are used. For validation set 2 ("Validation II"), the remaining scenarios are generated without such restriction, i.e., they are non-decomposable. This implies that the scenarios within the second validation set are not only non-decomposable with respect to

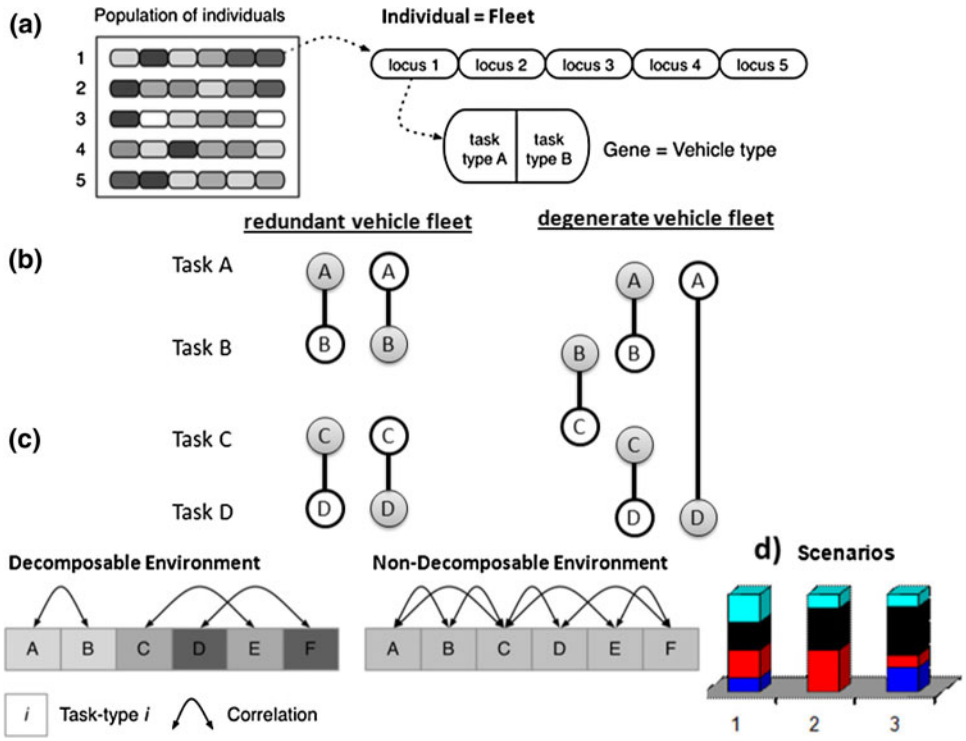

Fig. 1 **a** Fleet encoding in genetic algorithm. The genetic algorithm evolves a population of N individuals, each of which represents a complete fleet of vehicles: at each locus, an individual/fleet has a specific vehicle type as specified by two distinct task types. **b** A simple example highlighting the differences between sets of redundant and degenerate vehicles: given four vehicles (each vehicle represented as a pair of connected nodes) and four task types—*A*, *B*, *C*, *D*, the fleet on the *left-hand side* consists of two sets of identical vehicles. The fleet on the *right-hand side* consists of four unique vehicles with partial overlap between their functionalities (i.e., task types); in both cases, each task is represented to the same degree. **c** An illustration of task-type frequency correlations that characterize the variation in decomposable and non-decomposable environments. **d** Environmental scenarios differ from one another in the frequency of task types but not the total number of tasks (*vertical axis*)

one another but they are also non-decomposable with respect to the training scenarios.

3.2.4 Evaluating fitness

The fleet of vehicles is specified by the matrix V and the distribution of resources is given by the matrix R. The fitness of a fleet V at time t is determined by how well the fleet satisfies the demands imposed by the set of environments E_t. For every scenario $e_i^t \in E_t$, the fleet may compute a new matrix R_i^t. We denote as $\delta_j(R_i^t) = \sum_{k=1}^{n} R_{i\,kj}^t$ the sum of elements across column j in matrix R_i^t, which corresponds to the total amount of resources the fleet devotes to task type j. The fitness of the fleet with respect to environment e_i^t is then calculated as follows:

$$f_{e_i^t} = \sum_{j=1}^{m} \max\left\{0, e_{ij}^t - \delta_j(R_i^t)\right\}^2 \tag{2}$$

Fleets are exposed to ε scenarios at any one moment in time and the fitness of a fleet is simply the average over the individual fitness values:

$$F_{E_t} = \frac{1}{\varepsilon}\sum_{i=1}^{\varepsilon} f_{e_i^t} \tag{3}$$

It should be noted that fleet A is considered fitter than fleet B at time t when $F_{E_t}(A) < F_{E_t}(B)$.

The fitness of a fleet depends on its composition and the subsequent construction of R. The resources are allocated using a local asynchronous decision making process that aims to optimize fitness: all vehicles are considered in turn and for each vehicle, its resource allocation is adjusted (using ordered asynchronous updating). We increment/decrement the value R_{ij} according to how this affects the fleet's fitness (global feedback). Once no further improvements are possible, the next vehicle is considered. This process is repeated until no further improvements can be made.

3.3 Genetic algorithm

To evaluate the merits and limitations of reductionist versus degenerate design principles, we define specific design options for a fleet such that certain structural properties can be maintained within the fleet architecture. The resource allocation of the fleet is computed on the fly (as tasks are introduced) but the fleet's composition (i.e., its vehicle types) needs to be optimized towards a particular E_t. In order to do so, we employ a genetic algorithm based on deterministic crowding. The fleet is represented as a vector of vehicle types (Fig. 1a). During the evolutionary process, two parents are randomly selected from a population of 30 fleets (without replacement) and subjected to uniform crossover with probability 1: the crossover operator selects, with probability 0.5, for each locus along the chromosome, whether the corresponding gene (vehicle type) will be allocated to offspring 1 or 2. Each offspring is mutated (see below) and then replaces the genotypically more similar parent if its fitness is at least as good.

The design options available to a fleet (i.e., redundancy or degeneracy) are controlled exclusively by the mutation operator: When reductionist design principles are enforced by the mutation operator, the fleet always maintains a fully decomposable architecture. This is illustrated in Fig. 1b where fleets comprise vehicles that are either identical (purely redundant) or entirely dissimilar in their functionality. Unsurprisingly, fleets with this architecture acquire much of their robustness through vehicle redundancy and consequently are referred to as *redundant fleets*. When reductionism is not enforced by the mutation operator, some changes to vehicle designs may result in a partial overlap between the capabilities of vehicles (see right panel of Fig. 1b). Fleets evolved under these conditions are referred to as *degenerate fleets*. In our experiments, the initial fleet is always fully redundant independent of the mutation operator used.

The mutation operator plays a central role in the process of finding the best fleet design in a strategic planning problem in which possible future scenarios display different uncertainty characteristics (decomposable vs. non-decomposable, see Sect. 2.2.3 above): it is used to restrict the fleet compositions that may evolve. Its design is thus crucial, especially as it is important to ensure that both fleet compositions, redundant and degenerate, are obtained due to selective pressure alone and not auxiliary effects such as differences in solution space size. The mutation operator has thus been designed with the following considerations in mind: (a) the search space is to be of the same size in all experiments; (b) in some experiments both redundancy and degeneracy can be selected for during the evolutionary process (as opposed to degeneracy emerging as a consequence of the specifications of the model).

The mutation operator replaces exactly one randomly chosen vehicle in the fleet with a new vehicle type. For each locus in the chromosome, replacement options are predetermined and limited to $m/2$ unique vehicle types. For experiments in which the fleet cannot evolve degenerate architectures, alleles for all loci are drawn from set S. It follows that a purely redundant fleet remains redundant after mutation. For experiments in which the fleet can evolve degenerate architectures, allele options are almost the same as before, except that for half the alleles available at each locus, one task is changed to a new task type, thus allowing a partial overlap in functions to be possible between genes in the chromosome.

In some of our experiments we compare the results obtained from evolving fleets (using the genetic algorithm)

with "un-evolved" fleets that have the same composition characteristics as the evolved fleets. In constructing an un-evolved fleet, we proceed with an initial fleet as defined at the beginning of our evolutionary simulations, i.e., a randomly generated redundant fleet. We then proceed to iteratively implement the mutation operators associated with each fleet class (redundant, degenerate) with the mutated fleet replacing the original if its redundancy or degeneracy measurement becomes more similar to that of the evolved fleets. This mutation/selection procedure is iterated until redundancy or degeneracy differences are less than 5%.

3.4 Evaluating robustness

Possibly the clearest definition of robustness is put forth in (Alderson and Doyle 2010) which states: "a [property] of a [system] is robust if it is [invariant] with respect to a [set of perturbations]." The square brackets are used to emphasize that measurements of robustness requires one to specify the system, the property, the set of perturbations, and some measure of invariance (e.g., relative to some norm). Within the context of the proposed case study, robustness is related to system fitness and is characterized as the maintenance of good fitness values within a set of distinct scenarios, e.g., an aggregation function of fitness values F_{E_t} (Eq. 3) with $F_{E_t} < T$ where T describes some satisfactory fitness threshold.

One drawback to such measurements arises when the threshold T is high. Then complete and immediate failure is unlikely. While many fleets will "survive" or appear robust to most conditions, they may still exhibit substantial differences in their ability to maintain competitive performance with important subsequent effects to their long-term survival or growth prospects, i.e., there is a decoupling between robustness over short and long timescales. On the other hand, with tolerance thresholds T too low, all fleets will fail to meet the fitness requirements and will appear equally poor. This poses a technical challenge to fleet optimization as it eliminates selectable differences if corrective steps are not taken, e.g., thresholds are continually changed over time to ensure distinctions are possible in comparing fleets that would otherwise be consistently below or above satisfactory levels. One alternative is to create a two-tiered selection criterion whereby fleets are first compared based on satisfying fitness thresholds, and when differences are not resolved by this comparison they are further compared by their average fitness over all scenarios. This is the approach that we take in the analysis of our experiments. When using this measurement in preliminary tests we found it was still necessary to tune the threshold (or tune scenario volatility) in order to resolve differences between fleets.

While our findings might change when scenarios are assigned different priorities or when the number of scenarios is small (<20), our tests indicate that reporting an average fitness generates qualitatively similar conclusions in comparison to the use of a tuned robustness measurement. Average fitness is not a commonly used measure of robustness, however it does provide some advantages in the current study. In particular, an average fitness allows for a direct observation of abrupt performance changes in a system that could otherwise be dismissed as minor differences attributable to the crossing of a threshold.

4 Results

4.1 Evolution in environments with decomposable volatility

In the first set of experiments, we optimize fleets of vehicles to effectively handle a set of anticipated future scenarios. Each scenario defines a set of tasks that the fleet must perform and scenarios differ from one another in the frequency of task requirements. We proceed by investigating problems where environmental uncertainty is highly constrained and where reductionism in fleet design is expected to be favored. In particular, fleets are evolved in environments where, unbeknownst to the fleet optimization algorithm, the differences between scenarios exactly match vehicle capabilities within a fleet of a specific structural decomposition (i.e., a particular redundant fleet). We call these "scenarios with decomposable variation" (see Sect. 3.2.3).

With environmental variations matching a decomposable architecture, any fleet with that same decomposition will find that these variations can be decomposed into a set of smaller independent sub-problems that each can be addressed by a single class of vehicle and thereby solved in isolation from other vehicle types. This also means that any changes in fleet design that improve performance for one scenario will not result in lowered performance in other scenarios, i.e., performance across scenarios is correlated and evolution proceeds within a unimodal fitness landscape. These features provide ideal conditions for reductionist engineering design and we see in Fig. 2a that such design approaches can generate maximally robust fleets. Notice that according to our definition of fleet fitness, F_{E_t} in Eq. 3 a fleet is considered more robust than another fleet when its robustness measure is *smaller*. "Maximum" robustness is achieved when our robustness measure approaches 0.

Allowing fleet architectures to deviate from a pure decomposition should not be beneficial for these types of predictable problems. While the environment varies in an entirely decomposable manner, available fleet responses do

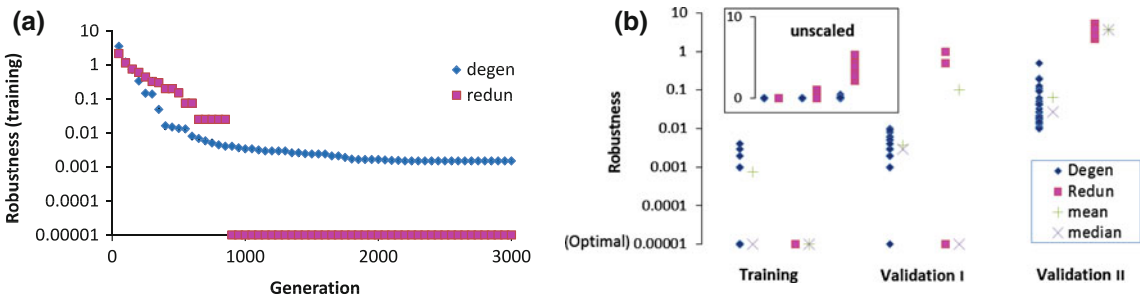

Fig. 2 **a** Robustness profiles of degenerate and redundant fleet designs evolved in environments with decomposable variation (*training scenarios*). **b** Robustness is then reevaluated within new *validation scenarios*. "Training" shows robustness values of all 30 fleets at the end of evolution against the training scenarios of "**a**". "Validation I" and "Validation II" show robustness values for the 30 evolved fleets against two different types of validation scenarios (described in main text). The *inset* shows the same results as the main panel but on a linear (not a logarithmic) robustness scale

not. During fleet evolution, we see in Fig. 2a that fleets with degeneracy initially evolve quickly and can learn how to respond to most variations within the environment, although evolution does not discover a maximally robust architecture in every problem instance (see "Training" results in Fig. 2b).

4.2 Robustness in environments with uncertainty

We next evaluate how the evolved fleets respond to various forms of environmental uncertainty. In particular, fleets evolved under the previous set of environments (now called *training scenarios*) are reevaluated in a new set of environments (*validation scenarios*). Uncertainty is characterized by the differences between training scenarios used during evolution and the new scenarios that are defined within the validation set. We present results for two validation tests that allow us to highlight some of the strengths and weaknesses that we have observed in the two fleet architectures.

4.2.1 Validation I

As described in Sect. 3.2.3, the first validation set is generated by selecting a scenario from the training set and using its position in scenario space as a seed for generating a new set of scenarios, via short random walks. The intuition behind this validation test is that actual future conditions are often partially captured by our expectations such that some expectations/scenarios are invalidated while other regions of scenario space become more relevant and are expanded upon. We impose the following additional constraints on validation set I: differences between scenarios must be decomposable when comparing scenarios within the validation set as well as across the training and validation sets.

While redundant fleets are not guaranteed to respond optimally to every new scenario, the constraints imposed on the validation set should place these fleets at a considerable advantage. It is seen in Fig. 2b that redundant fleets often maintain optimal robustness; however, they also occasionally exhibit poor performance outcomes. We found bimodal validation responses were a common phenomena in redundant fleets. In particular, redundant fleets tended to provide highly effective responses towards scenarios that fit within their design limits but once these limits were exceeded, their performance degraded markedly.

In contrast, the degenerate fleets are not designed in a manner that is biased towards the validation set's decomposition. In many validation instances the fleet's robustness is degraded as its repertoire of responses fails to match the new requirements (Fig. 2b). However, performance losses were attenuated, i.e., degenerate fleets were less sensitive to environmental demands that deviated from optimal conditions than were redundant fleets.

4.2.2 Validation II

The next set of validation tests utilize similar conditions as before except that non-decomposable variation is now permitted in the set of validation scenarios. More precisely, differences between scenarios are decomposable when comparing scenarios within the training set, but not decomposable for comparisons within the validation set or for comparisons across sets. This is a harder validation test to satisfy, firstly because the validation scenarios do not meet the assumptions implicit in the training data. Secondly, decomposition requirements restrict scenarios to compact regions of scenario space and in removing this restriction, the scenarios can now be more distinct from one another, even if their distance from the seed position remains unchanged.

Under these validation conditions, we see that the robustness of redundant fleets considerably degrades. Degenerate fleets also suffer degradation in robustness, however they display substantially better responses towards this type of novelty compared to redundant fleets.

Our findings vary somewhat depending on the parameter settings that are used to generate the validation tests. For both validation sets, as the distance between the training seed and validation seed grows, robustness decays in both types of fleets although this is more rapid for the redundant fleets. As the volatility is reduced in the training set, the robustness advantage of degenerate systems also decreases; in the limit where there is only a single member in the set of training scenarios, both fleet classes always evolve to optimal robustness.

4.3 Evolution in environments with non-decomposable volatility

In strategic planning, it is unlikely that anticipated future states (e.g., training scenarios) or the actual future states (e.g., validation scenarios) will involve compartmentalized variations. In the context of our model, this means that task requirements are more likely to change such that the covariance in task frequencies cannot be decomposed into a set of isolated relationships or sub-problems. The second set of validation scenarios evaluated in Fig. 2b were formulated to capture this type of "non-decomposable uncertainty" (see Sect. 3.2.3) as they allow correlations to arise between the frequencies of any pair of task types.

For these reasons, we focus our attention on the robustness and design adaptation characteristics for fleets that *evolve* under Validation II type non-decomposable environments. By tracking fleet robustness during the course of evolution, we now see in Fig. 3a that degenerate fleets are becoming substantially more robust and do so more quickly than fleets evolved using reductionist design principles.

In our view, there are two factors that primarily contribute to these observed differences: (i) evolvability of the fleet architecture and (ii) differences in the robustness potential of the system. Conceptually, evolvability is about

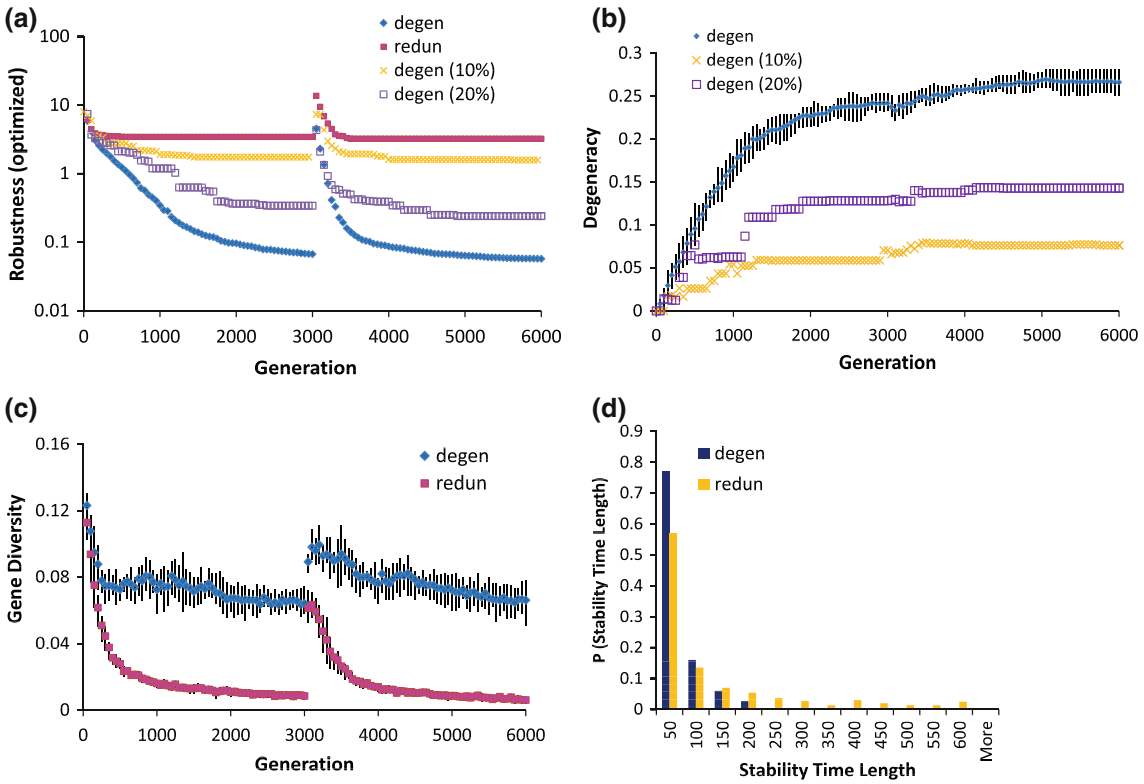

Fig. 3 **a** Robustness profiles of degenerate and redundant fleet designs evolved in environments with non-decomposable variation and exposed to a shock (i.e., an entirely new set of scenarios) at generation 3000. Characteristic results are also shown for experiments where only a predefined percentage of vehicles can be designed to no longer conform to the initial redundant fleet architecture and thus allow for some restricted level of partially overlapping capabilities within the fleet. **b** Degeneracy is measured (for fleets where it can emerge) during the course of evolution. **c** Gene diversity during the evolution of "**a**"; measured as the proportion of vehicles that have changed in pair-wise comparisons of fleets within the population. **d** histogram for the number of offspring sampled before an improvement is found (time length). Sampling is restricted to the evolution of fleets throughout the first 3000 generations of "**a**"

discovering design improvements. In Fig. 3d, we record the probability distribution for the time it is taking the optimization algorithm to find new improvements to the fleet design. As can be seen, degenerate architectures are allowing for a more rapid discovery of design improvements than redundant architectures. A further analysis of fleet evolution confirms that design adaptation rates predominantly account for this divergence in robustness values. In Fig. 3b we track degeneracy levels within the fleet and find that, when permitted, degeneracy readily emerges in the fleet architecture. We will show that the growth of degeneracy during evolution is a key factor in both the robustness and design adaptation of these fleets.

4.4 Adaptation to novel conditions

One of the claimed advantages of degeneracy is that it supports a system in dealing with novelty; not only in system design but also in environmental conditions. To investigate this claim, we expose the fleets to an entirely new set of scenarios and allow the fleet designs to re-evolve (Fig. 3a, generations 3000–6000). When high levels of degeneracy have already evolved within the fleet architecture, we see that the speed of design adaptation increases. When only redundancy is permitted, design adaptation after shock exposure at generation 3000 appears marginally worse. In our analysis of these results, we determined that the degraded adaptation rates in redundant fleet (re)evolution is an artifact of the optimization procedure and can readily be explained based on the following considerations: (i) we use a population-based optimization algorithm, i.e., many fleet designs are being modified and tested concurrently, (ii) the population of fleets is converging over time, i.e., differences between fleets or *diversity* get lost (Fig. 3c), and (iii) it is generally true, in both biology and population-based optimization, that a loss of diversity negatively affects a population's adaptive response to abrupt environmental change. We can confirm the population diversity effect by initializing populations at low diversity or more simply by reducing the population size, which results in redundant fleet robustness profiles that are indistinguishable when comparing evolution in old (generations 0–3000) and new (generations 3000–6000) environments.

In contrast, degenerate fleets evolve improved designs more rapidly in the new environment (Fig. 3a) and this effect is not eliminated by altering the population size. However, when we restrict the total amount of degeneracy that is allowed to arise in the fleet (Fig. 3b), the fitness profiles before and after shock exposure become increasingly similar (Fig. 3a). Together, these findings support the hypothesis that degeneracy improves fleet design evolvability under novel conditions.

4.5 Innate and evolved differences in the robustness potential of degenerate fleets

In our introductory remarks, we proposed that degeneracy may allow for pervasive flexibility in the organization of system resources and thereby afford general advantages for a system's robustness. To evaluate this proposal, in Fig. 3d we introduce fleets that are not evolved but instead are artificially constructed (see Sect. 3.2) so that they have levels of degeneracy and redundancy that are similar to their evolved counterparts from the experiments in Fig. 3a. Figure 4 reports the robustness of these (un-evolved) fleets against randomly generated scenarios. We see that innate advantages in robustness are associated with degenerate fleet architectures. We contend that this finding is not intuitively expected based on a reductionist view of vehicle capabilities given that: both degenerate and redundant fleets have the same number of vehicles (and the fleet design spaces are constrained to identical sizes), fleets are presented with the same scenarios, and vehicles have access to the same task capabilities.

It is interesting to ask whether this innate robustness advantage can explain entirely our previous results or whether the manner in which degeneracy evolves and becomes integrated within the fleet architecture is also important, as is proposed in the networked buffering hypothesis (Whitacre and Bender 2010a). To explore this question, in Fig. 4 we show robustness results of the final evolved fleets from Fig. 3a but now robustness is reevaluated against randomly generated scenarios. Here we see that, when degeneracy evolves, a fleet's robustness to novel conditions becomes further enhanced, indicating that the organizational properties of degeneracy play a significant role in a system's robustness and capacity to adapt. Importantly, it suggests that the flexibility is tied to the

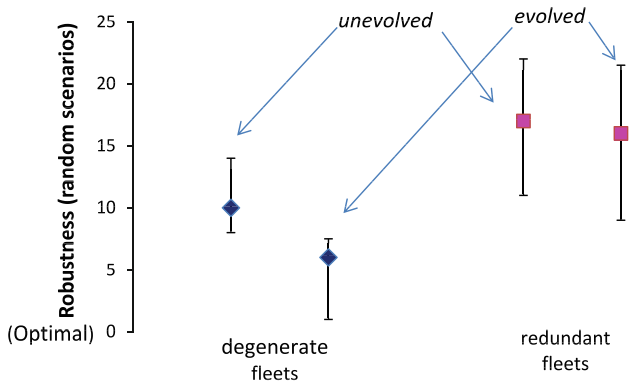

Fig. 4 Comparisons of the robustness of evolved and un-evolved fleets towards randomly generated scenarios. Un-evolved degenerate fleets were constructed with levels of degeneracy as large as evolved fleets

history of the system's evolution yet remains pervasive and can emerge without foresight, thus satisfying our third stated requirement for supporting innovation. In contrast, the capacity to deal with novel conditions is unaffected by evolution in redundant fleets and remains generally poor for both evolved and randomly constructed fleet designs.

4.6 Costs in the design and redesign of fleets

So far, we have found that degenerate fleet designs can be broadly robust as well as highly adaptable; enabling these fleets to exploit new design options and to respond more effectively to novel environments. Only when environments are stable and characterized to favor reductionist principles did we find redundant architectures to slightly outperform degenerate architectures. While these preliminary findings are promising, there are additional factors that should be considered when weighing the pros and cons of degeneracy design principles.

Depending on how the planning problem is defined, it may be of interest to know how the fleet composition changes in response to a new set of environmental demands. For instance, if we considered the initial conditions in our experiments to represent a pre-existing fleet, then it would be important to know the number of vehicles that are being replaced, as this would clearly influence the investment costs.

We would expect that, without having explicit cost objectives in place, degenerate fleets would diverge rapidly from their original conditions. Interestingly, we find in Fig. 5 that divergence is not substantially greater in comparison to reductionist design conditions and that more favorable cost-benefit trade-offs appear to exist for evolution under non-decomposable environments, *cēterīs paribus*.

For example, in decomposable environments an optimally redesigned redundant fleet was typically found by replacing 15% of the vehicles while degenerate fleets achieved near optimal performance when approximately 20% of the fleet is redesigned (Fig. 5a). In complex environments, no fleet typically is able to discover an optimal redesign, however, the increased propensity to discover adaptive design options is shown to confer degenerate fleets with a performance advantage that becomes more pronounced as larger investments are considered (Fig. 5b).

However, such conclusions do not factor in additional costs from the development of new vehicle designs or reduced costs that may come from economies of scale

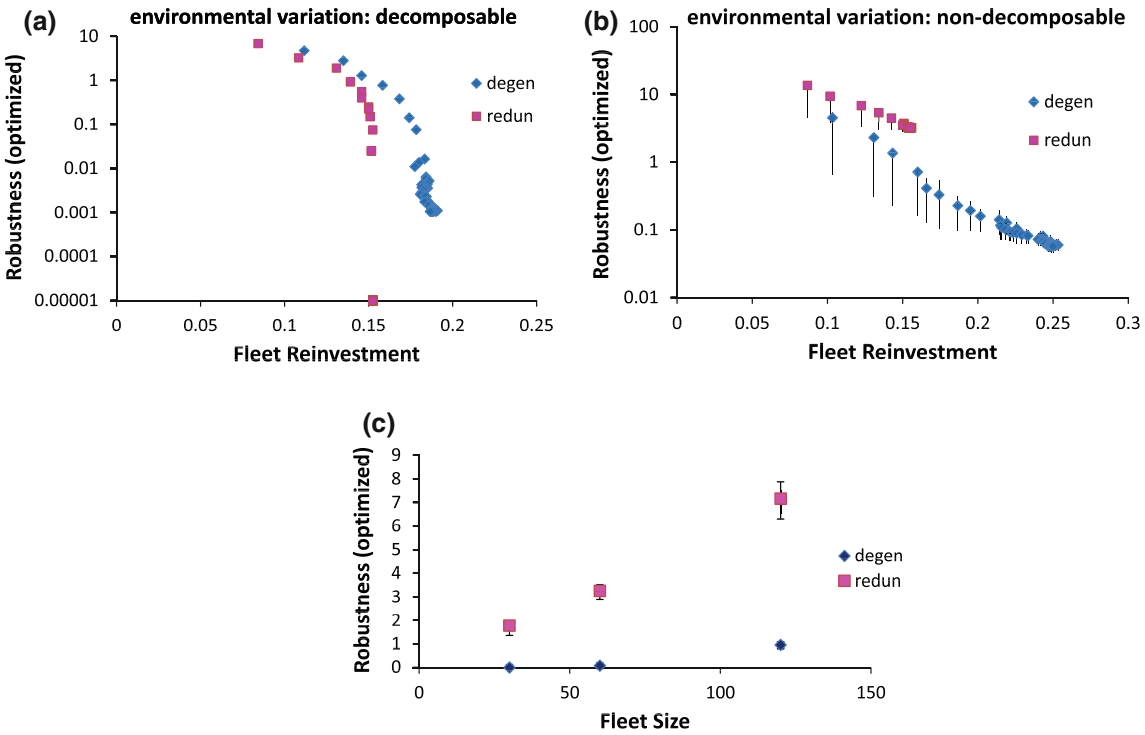

Fig. 5 **a** Robustness of evolved fleets plotted against the proportion of vehicles that have changed when comparing an evolved fleet with its original design (at gen = 0). Evolution takes place within a decomposable environment. **b** Same as "**a**" but with evolution taking place in non-decomposable environments. **c** Comparisons of the evolved fleet robustness for degenerate and redundant architectures at different fleet sizes. In these experiments the fleet size, the number of task types T, random walk size, and maximum generations are all increased by the same proportion

during the manufacturing and purchase of redundant vehicles. While such costs depend on the planning context, these factors will influence the advantages and disadvantages from degeneracy and warrant further exploration. Having noted the potential costs from degeneracy, one should not immediately conclude that fleet scaling places degeneracy design principles at a disadvantage. For instance, in Fig. 5 we consider fleet evolution (under non-decomposable environments) at different sizes of our model. Here we see that the robustness advantage from degeneracy has a non-linear dependence on fleet size. Moreover, because degeneracy is a relational property, modifying the design of a small number of vehicles can have disproportionate influence on degeneracy levels within a fleet. For instance, in one of the data sets in Fig. 3a only 20% of the fleet is permitted to deviate from its initially redundant architecture yet we observe considerable improvements in degeneracy and fleet robustness. In short, the careful redesign of a small subset of the fleet could provide considerable advantages, particularly for large fleet sizes.

4.7 Multi-scaling effects

In our fleet fitness evaluation, it was necessary to determine how vehicle resources can be matched to the task requirements for each scenario. While decision support tools might view this as a simulation (Davis 2002), in operations research this simulation would be formulated as a resource assignment problem. Regardless of how the fitness evaluation is viewed conceptually, architectural properties of the fleet are likely to have non-trivial consequences in the assignment of vehicles to tasks.

As formally described in Sect. 3, we approximate task assignment by a local decision-making heuristic. While this heuristic simplifies our experiments, it also maintains a rough analogy to how these decisions are made in reality, i.e., by vehicle owners (management) instead of optimization algorithms. The "owner" of a vehicle distributes the vehicle's resources (time) over suitable tasks. However, as other owners make their own decisions, the relative importance of remaining unfulfilled tasks can change and owners may decide to readjust their vehicle's tasks, thus (indirectly) adapting to the decisions of others. In the following, we investigate how a fleet's architecture influences the amount of time required to complete this task assignment procedure and similarly, how placing restrictions on the time allotted to the procedure, i.e., changing the *maximum simulation runtime*, influences the performance results for different fleet architectures.

In Fig. 6a, we evolve fleets with different settings for the maximum simulation runtime and record how this alters the final robustness achieved after evolution. In the limiting case where vehicle task assignments are never updated, a fleet has no capacity to respond to changes in the environment and performance is entirely genetically determined, i.e., vehicles are not functionally plastic. In this case the problem collapses to one that is equivalent to optimizing for a single scenario (i.e., the mean task requirements of the scenario set) and the two types of fleets evolve to the same poor performance. When fitness evaluation is extended to allow short simulation times (i.e., a few decision adjustments), degenerate architectures display modestly better matches between vehicles and tasks and this continues to improve as simulation time is increased and robustness converges to near optimal values (Fig. 6a).

The experiments reported earlier (in Sects. 4.1–4.6) did not impose restrictions on the simulation runtime, i.e., task assignment heuristics were run until no decisions could be readjusted. To evaluate the task assignment efforts that took place in these experiments, Fig. 6c plots the frequency distribution for the number of decision adjustments that occur as a fleet responds to a scenario. As can be seen the discovery of a stable assignment can take considerably longer for degenerate fleets than for redundant fleets.

However, we also have seen in Fig. 6a that constraining the assignment runtime had only modest repercussions to performance when the fleets are forced to evolve under these constraints. We have found two factors that contribute to these seemingly contradictory findings. Firstly, the largest fitness improvements that result from decision adjustments predominantly occur during the early stages of a simulation. This is shown in Fig. 6b where fitness changes are recorded as decision adjustments are made. While some fleet-scenario combinations potentially require many decision adjustments, the fleet still provides decent performance if the simulation is terminated early. We also speculate that a second contributing factor could be that evolution under restricted runtimes influences how the architecture evolves. With simulation runtime restricted, evolution may prefer fleet architectures that find decent assignments more quickly.

To investigate this conjecture, we evolve fleets with simulation runtime restricted to at most ten decision adjustments. We then remove this restriction and record the total number of decision adjustments as the fleets respond (without runtime restriction) to new scenarios (Fig. 6d). Compared to the results in Fig. 6c, we see a large reduction in runtime distributions indicating that fleets have been designed to be more efficient in task assignment. In addition higher quality vehicle assignments are found more quickly for degenerate fleets optimized under these constrained conditions (Fig. 6b).

In summary, an exploration of the underlying resource assignment problem indicates that important interactions exist in fleet performance characteristics as they are

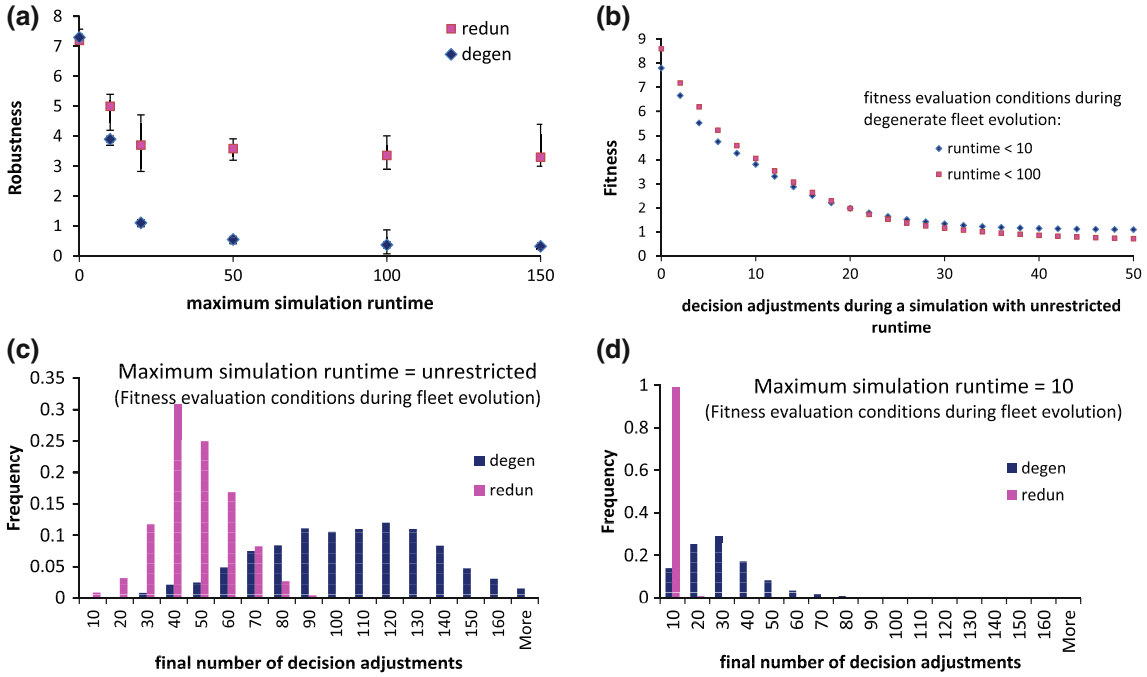

Fig. 6 **a** Robustness of fleets that have been evolved with different restrictions on maximum simulation runtime. **b** Fitness of a fleet is evaluated in a single scenario where fitness is recorded during a simulation, i.e., as vehicle decision adjustments are made. The results are shown for degenerate fleets that have evolved in conditions where the maximum simulation runtime is 100 and 10 readjustments. **c** Actual runtime distribution for fleets evolved under unrestricted runtime conditions. **d** Actual runtime distribution for fleets evolved under a maximum simulation runtime of 10, but where the distribution is being evaluated with these restrictions removed

observed at different timescales of our planning problem. However, in the context of the distributed decision-making heuristics that were implemented, we have generally found that degeneracy based design can result in high robustness without incurring large implementation costs. Importantly however, satisfactory cross-scale relationships depend on whether short-timescale objectives (i.e., simulation runtime) are being accounted for during fleet planning.

5 Discussion

Fleet degeneracy transforms a trivial resource allocation problem that, in principle, can be solved analytically into a problem with parameter interdependencies that is not analytically tractable for large systems. From a traditional operations research perspective, degeneracy should thus be avoided because it increases the difficulty of finding a globally optimal assignment of resources. However, despite increased complexity in centralized resource control, our results in Sect. 4.7 indicate that simple distributed decision-making heuristics can provide excellent resource allocation assignments in degenerate fleets. The superior fleet design easily lends itself to a more effective allocation of resources compared with the globally best solutions for fleets designed based on reductionist principles.

By exploring this conflict between design and operational objectives (Sects. 4.6 and 4.7), our study has identified conditions that support but also limit the benefits that arise from degeneracy. These conditions appear to be in agreement with observations of distributed control in biological systems where trait robustness generally emerges from many localized and interrelated decisions (Edelman and Gally 2001). How these local actions translate to system traits is not easily understood by breaking down the system into its respective parts. The lack of centralized control thus has put into question whether biological principles are compatible with engineering and sociotechnical systems where centrally defined global objectives play an important role in assessing and integrating actions across an organization. However, and in contrast to common intuition, we have found some evidence to suggest that difficulties in centralized control do not preclude the possibility of coherent system behaviors which are locally determined yet also driven by globally defined objectives.

Reductionist design principles should be well-suited for addressing planning problems that can be readily separated

into sub-problems and where variations in problem conditions are bounded and predictable. On the other hand, we propose that degeneracy should be better suited for environments that are not easily decomposed and that are characterized by unpredictable novelty.

5.1 Implications for complex systems engineering

As a design principle, degeneracy could be applicable to several domains, in particular in circumstances where

(1) distributed decision-making is achievable (e.g., in markets, open-source software, some web 2.0 technologies, human organizations and teams with a flat management structure);
(2) agent motivations can be aligned with system objectives; and,
(3) agents are functionally versatile while also following protocols for reliable agent–agent collaboration.

When these conditions are met, designing elements with partially overlapping capabilities can dramatically enhance system resilience to new and unanticipated conditions as we have shown. On the other hand, if there is a need to maintain centralized decision-making and system-level transparency, or in cases where historical bias favoring reductionism is deeply ingrained, implementing degeneracy design principles is likely to prove difficult and possibly unwise.

There is a growing awareness of the role played by degeneracy in biological evolution. However, these insights have not yet been taken into account in the application of nature-inspired principles to solve real-world problems. We propose that our findings can shed new light on a diverse range of research efforts that have taken insights from biology and complexity science to address uncertainties arising in systems engineering and planning (Sheard and Mostashari 2008; Ilachinski 1996; Ryan 2007; Levchuk et al. 2002). Degeneracy is a system property that:

(1) based on both empirical evidence and theoretical arguments, can facilitate the realization of other important properties in complex systems, including multi-scaled complexity (Minai et al. 2006; Tononi et al. 1999), resilience under far from equilibrium dynamics (Holling 1996), robustness within outcome spaces (Kuras and White 2005), the establishment of "solution rich configuration spaces", and evolvability (Minai et al. 2006; Kirschner and Gerhart 1998; Fricke and Schulz 2005);
(2) depends on the presence of other basic system properties that were implemented in our experiments such as distributed decision-making (Kuras and White 2005), feedback (Grisogono and Ryan 2003; Grisogono 2006), modularity/encapsulation (Fricke and Schulz 2005), protocols/interfaces/tagging (Csete and Doyle 2002; Grisogono and Ryan 2003), weak/loose coupling (Fricke and Schulz 2005; Csermely 2006), exploration (Kirschner and Gerhart 1998), agent agility (Grisogono and Spaans 2008; Grisogono 2006), and goal alignment within a multi-scaled system (Minai et al. 2006; Grisogono 2006).

5.2 Reengineering in complex systems

Our introductory remarks regarding system's engineering intentionally emphasized distinctions between designed and evolved systems, however engineering in practice can take on features of both. Engineering is not simply drawing up a plan and implementing it. Human-constructed systems typically coevolve with their environment during planning, execution, and even decommissioning activities. While biological terminology is not common to this domain, several attributes related to degeneracy are found in socio-technical systems and are sometimes intentionally developed during planning.

For instance, functional redundancy is a common feature in commercial/social/internal services when a sustained function under variable conditions is critical to system performance or safety, e.g., communications and flight control for commercial aircraft. Functional redundancy is rationally justified in these cases based upon its relationship to service reliability (see Sect. 1) which is analogous to the principles underpinning economic portfolio theory, i.e., risks are mostly independent while returns are additive. However if nothing else, this study and other recent studies have shown that a significant amount of the robustness derived from degeneracy has origins that extend beyond simple diversity or portfolio effects and is conferred instead as an emergent system property. For instance, previous work has found evidence that the enhanced robustness effects from degeneracy originate from distributed compensatory pathways or so called *networked buffers* (Whitacre and Bender 2010a).

Moreover, it is not simply reliability in function but also the relationship between robustness and innovation that makes degeneracy especially important to systems engineering. To better understand this relationship between degeneracy, robustness, and innovation, we have taken an interdisciplinary approach that combines system's thinking, biological understanding and experimentation using agent-based simulations. In ongoing research, we are using this approach to explore how degeneracy might facilitate successful organizational change within an organization as well as successful reengineering for systems that display functional interdependencies in the mapping between agent actions and system capabilities.

6 Conclusions

In this article, we have investigated the properties of degenerate buffering mechanisms that are prevalent in biological systems. In comparison to their engineering counterparts, these buffering mechanisms were found to afford robustness to a wider range (both in type and magnitude) of perturbations and do so more efficiently due to the manner in which these buffers interact and cooperate. While seemingly paradoxical, we also hypothesized how the same mechanisms that confer trait stability can also facilitate system adaptation (changes to traits) under novel conditions. With the design of transportation fleets taken as a case study, we reported evidence supporting this hypothesis, demonstrating that the incremental growth of degeneracy results in fundamentally different robustness and design adaptation properties within fleets.

The theories that underpin these ideas are conceptually straightforward (Whitacre 2010a; Edelman and Gally 2001) yet also operationally useful as they position degeneracy as a mechanistic facilitator of robustness and adaptability that, in principle, can be applied outside biological contexts (Whitacre 2010a). In looking to tackle real-world problems using inspiration from biology, it is important to determine whether sufficient parallels exist between the problem and the biological system of interest. Here we have proposed that the investment decisions and subsequent operation of complex engineered systems consisting of versatile semi-autonomous agents provide a general domain where these requisite conditions are met and where degeneracy design principles could prove advantageous. There are a number of systems that can be characterized in this manner, with strategic planning for field vehicle fleets provided as one illustrative example.

Other suitable domains may include particular applications of agile manufacturing (Frei and Whitacre, 2011) and applications of swarm robotics. There are however additional challenges that arise in these other domains because humans play a smaller role in the communication and control of agent–agent interactions. For instance, the design and management of protocols for agent–agent communication and co-regulation were ignored in our case study due to the central role of humans in managing military vehicle operations but are important issues in agile manufacturing, swarm robotics and similar systems.

Our future research will continue to investigate the influence of degeneracy in systems that comprise both humans and hardware assets, however we will increasingly focus on the social dimension of such problems. For instance, some of our future studies will investigate how degeneracy in the skill mix of military and emergency response teams can influence team flexibility when such teams have to deal with surprises and novel threats. Team flexibility may become particularly important in situations where individuals are unexpectedly required to take on new roles in a crisis.

Acknowledgments This work was partially supported by a DSTO grant on "Fleet Designs for Robustness and Adaptiveness" and an EPSRC grant (No. EP/E058884/1) on "Evolutionary Algorithms for Dynamic Optimisation Problems: Design, Analysis and Applications."

References

Alderson DL, Doyle JC (2010) Contrasting views of complexity and their implications for network-centric infrastructures. IEEE Trans Syst Man Cybern A 40:839–852

Atamas S, Bell J (2009) Degeneracy-driven self-structuring dynamics in selective repertoires. Bull Math Biol 71:1349–1365

Bak P, Paczuski M (1995) Complexity, contingency, and criticality. Proc Natl Acad Sci USA 92:6689–6696

Banzhaf W (1994) Genotype–phenotype-mapping and neutral variation—a case study in genetic programming. In: Parallel problem solving from nature—PPSN III, vol 866. Springer, Berlin, pp 322–332

Batada NN et al (2007) Still stratus not altocumulus: further evidence against the date/party hub distinction. PLoS Biol 5:e154

Beckerman L (2000) Application of complex systems science to systems engineering. Syst Eng 3:96–102

Calvano C, John P (2003) Systems engineering in an age of complexity. Syst Eng 7:25–34

Carlson JM, Doyle J (2002) Complexity and robustness. Proc Natl Acad Sci USA 99:2538–2545

Carpenter S et al (2001) From metaphor to measurement: resilience of what to what? Ecosystems 4:765–781

Ciliberti S et al (2007) Innovation and robustness in complex regulatory gene networks. Proc Natl Acad Sci USA 104:13591–13596

Crutchfield JP, Van Nimwegen E (2002) The evolutionary unfolding of complexity. In: DIMACS workshop, Princeton

Csermely P (2006) Weak links: stabilizers of complex systems from proteins to social networks. Springer, Berlin

Csete ME, Doyle JC (2002) Reverse engineering of biological complexity. Science 295:1664–1669

Davis P (2006) Strategic planning amidst massive uncertainty in complex adaptive systems: the case of defense planning. In: Minai AA, Bar-Yam Y (eds) Unifying Themes in Complex Systems, Berlin, Heidelberg: Springer, 201–214

Davis PK (2005) New paradigms and new challenges. In: Proceedings of the conference on winter simulation, Orlando, pp 1067–1076

Davis PK et al (2007) Enhancing Strategic planning with massive scenario generation: theory and experiments. RAND Technical Report

Edelman GM, Gally JA (2001) Degeneracy and complexity in biological systems. Proc Natl Acad Sci USA 98:13763–13768

Frei R, Whitacre J (2011) Degeneracy and networked buffering: principles for supporting emergent evolvability in agile manufacturing systems. Nat Comput 1–14. doi:10.1007/s11047-011-9295-4

Fricke E, Schulz A (2005) Design for changeability (DfC): principles to enable changes in systems throughout their entire lifecycle. Syst Eng 8:342–359

Fromm J (2006) On engineering and emergence. Arxiv preprint nlin/0601002

Gell-Mann M (1995) What is complexity. Complexity 1:1–9

Gould S (1990) Wonderful life: the burgess shale and the nature of history. WW Norton, New York

Gribble SD (2001) Robustness in complex systems. In: The eighth workshop on Hot Topics in Operating Systems (HotOS VIII), Elmau

Grisogono A-M (2006) The implications of complex adaptive systems theory for C2. In: Proceedings of 2006 command and control research and technology symposium (CCRTS), Washington

Grisogono AM, Ryan A (2003) Designing complex adaptive systems for defence. In: Systems engineering test and evaluation conference, Canberra

Grisogono A-M, Spaans M (2008) Adaptive use of networks to generate an adaptive task force. In: Proceedings of 13th international command and control research and technology symposium (ICCRTS), Washington

Holling C (1996) Engineering resilience versus ecological resilience. In: Engineering within ecological constraints. National Academy Press, Washington, pp 31–43

Ilachinski A (1996) Land warfare and complexity, part II: an assessment of the applicability of nonlinear dynamics and complex systems theory to the study of land warfare. Center for naval analyses, Alexandria

Kauffman SA (1990) Requirements for evolvability in complex systems: orderly components and frozen dynamics. Phys D 42:135–152

Keller RE, Banzhaf W (1996) Genetic programming using genotype-phenotype mapping from linear genomes into linear phenotypes. In: Presented at the proceedings of the first annual conference on genetic programming stanford, California

Kerwin A (1993) None too solid: medical ignorance. Sci Commun 15:166–185

Kimura M (1955) Solution of a Process of random genetic drift with a continuous model. Proc Natl Acad Sci USA 41:144–150

Kimura M (1983) The neutral theory of molecular evolution. Cambridge University Press, Cambridge

Kirschner M, Gerhart J (1998) Evolvability. Proc Natl Acad Sci USA 95:8420–8427

Knowles JD, Watson RA (2003) On the utility of redundant encodings in mutation-based evolutionary search. In: Lecture notes in computer science, vol 2439. Springer, Berlin, pp 88–98

Kurakin A (2007) Self-organization versus Watchmaker: ambiguity of molecular recognition and design charts of cellular circuitry. J Mol Recognit 20:205–214

Kurakin A (2009) Scale-free flow of life: on the biology, economics, and physics of the cell. Theor Biol Med Model 6:6

Kurakin A (2010) Order without design. Theor Biol Med Model 7:12

Kuras ML, White BE (2005) Engineering enterprises using complex-system engineering. In: INCOSE symposium, Rochester, pp 10–14

Lane J, Boehm B (2007) System of systems lead system integrators: where do they spend their time and what makes them more or less efficient? Syst Eng 11:81–91

Lempert RJ et al (2003) Shaping the next one hundred years: new methods for quantitative, long-term policy analysis. RAND Technical Report

Levchuk G et al (2002) Normative design of organizations-part II: organizational structure. IEEE Trans Syst Man Cybern A 32:360–375

Minai A et al (2006) Complex engineered systems: a new paradigm. In: Complex engineered systems. Springer, Berlin, pp 1–21

Mogul J (2006) Emergent (mis) behavior vs. complex software systems. In: European conference on computer systems, Leuven, pp 293–304

Ohta T (2002) Near-neutrality in evolution of genes and gene regulation. Proc Natl Acad Sci USA 99:16134–16137

Ottino J (2004) Engineering complex systems. Nature 427:399

Polacek G, Verma D (2009) Requirements engineering for complex adaptive systems: principles vs. rules. In: 7th annual conference on systems engineering research 2009 (CSER 2009), Loughborough University

Rothlauf F, Goldberg DE (2003) Redundant representations in evolutionary computation. Evol Comput 11:381–415

Ryan A (2007) A multidisciplinary approach to complex systems design. Department of Mathematics (thesis), University of Adelaide, Adelaide

Sheard SA, Mostashari A (2008) Principles of complex systems for systems engineering. Syst Eng 12:295–311

Smith T et al (2002) Neutrality and ruggedness in robot landscapes. In: Congress on evolutionary computation, Honolulu, pp 1348–1353

Stelling J et al (2004) Robustness of cellular functions. Cell 118:675–685

Tononi G et al (1999) Measures of degeneracy and redundancy in biological networks. Proc Natl Acad Sci USA 96:3257–3262

Vassilev VK, Miller JF (2000) The advantages of landscape neutrality in digital circuit evolution. In: Evolvable systems: from biology to hardware. Springer, Berlin

Wagner A (2008) Robustness and evolvability: a paradox resolved. Proc R Soc Lond B Biol Sci 275:91–100

Whitacre JM (2010a) Degeneracy: a link between evolvability, robustness and complexity in biological systems. Theor Biol Med Model 7:6

Whitacre JM (2010b) Evolution-inspired approaches for engineering emergent robustness in an uncertain dynamic world. In: Artificial life XII, Odense, pp 559–561

Whitacre JM (2010c) Genetic and environment-induced innovation: complementary pathways to adaptive change that are facilitated by degeneracy in multi-agent systems. In: Artificial life XII, Odense, pp 431–433

Whitacre JM, Bender A (2009) Degenerate neutrality creates evolvable fitness landscapes. In: WorldComp-2009, Las Vegas

Whitacre JM, Bender A (2010a) Networked buffering: a basic mechanism for distributed robustness in complex adaptive systems. Theor Biol Med Model 7:20

Whitacre JM, Bender A (2010b) Degeneracy: a design principle for achieving robustness and evolvability. J Theor Biol 263:143–153

Whitacre JM et al (2010) The role of degenerate robustness in the evolvability of multi-agent systems in dynamic environments. In: PPSN XI, Krakow, pp 284–293

Yu T, Miller JF (2001) Neutrality and the evolvability of boolean function landscape. In: Proceedings of the 4th European conference on genetic programming, Lake Como, pp 204–217