Research in Engineering Design (1999)11:172–192

Research in
Engineering
Design

A-Design: An Agent-Based Approach to Conceptual Design in a Dynamic Environment

Matthew I. Campbell[1], Jonathan Cagan[1] and Kenneth Kotovsky[2]

[1]Computational Design Laboratory, Department of Mechanical Engineering; [2]Department of Psychology, Carnegie Mellon University, Pittsburgh, PA, USA

Abstract. *This paper provides an introduction to a new design methodology known as A-Design, which combines aspects of multi-objective optimization, multi-agent systems, and automated design synthesis. The A-Design theory is founded on the notion that engineering design occurs in interaction with an ever-changing environment, and therefore computer tools developed to aid in the design process should be adaptive to these changes. In this paper, A-Design is introduced along with some simple test problems to demonstrate the capabilities of different aspects of the theory. The theory of A-Design is then shown as the basis for a design tool that adaptively creates electro-mechanical configuration designs for changing user preferences.*

Keywords: Adaptive search; Agents; Design synthesis; Multi-objective optimization

1. Introduction

Traditionally, the engineering design process is a cooperative effort among engineers and management, where virtual and physical prototypes are constructed, tested and evaluated. Based on constantly changing design goals, available technologies, and design constraints, managers refine the design specifications and appropriate design teams, while engineers within those teams collaborate to affect changes to the design to meet target specifications.

The goals of a design problem are established to address market-driven issues such as product cost or performance, and engineering objectives such as target lifecycle or efficiency. One company's product can be distinguished from others in the manner in which it achieves the various product design goals. While some companies produce inexpensive designs in order to find a profitable market niche, others may produce high-end products; to position one's product amongst competing products a constant awareness of market demands is required. However, in the conceptual design phase, it is not always possible to set the relative importance of the various objectives and constraints prior to generating solutions, as is required in most automated design systems. Current computer tools tend to be static in, and unresponsive to, such changes in the problem description and the solution strategy. These limitations motivate the conceptual design methodology introduced in this paper, which is capable of adapting to changes in design problem specifications.

This paper introduces A-Design – a new search strategy for the conceptual stages of engineering design that incorporates **a**gent collaboration with an **a**daptive selection of designs. Having a multitude of agents that are responsible for the same task yet present different strategies and differing knowledge for solving the same design problem generates a greater variety of design alternatives. The system gains robustness due to the collaboration of these varying agents, thus affording the design process the ability to change focus when the user's preference changes. These agents are analogous to individual specialists within a design firm. As with a company, there is usually not a single optimal design, but rather designs evolve and, as they do so, the company selects designs at certain states and markets them as products. This dynamic nature of true engineering design is the foundation for creating A-Design in a manner that is adaptive in both the design process and design specification.

The basic subsystems of the A-Design theory are (1) an agent architecture that is responsible for creating and improving design alternatives, (2) a representation of the conceptual design problem that is comprehended by the agents in order to create design concepts, (3) a scheme for multi-objective

Correspondence and offprint requests to: Dr. J. Cagan, Computational Design Laboratory, Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA

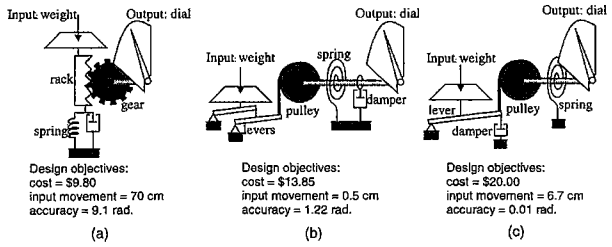

Fig. 1. Initial weighing machine results: (a) Low cost design, (b) minimum input displacement design, (c) accurate dial design.

decision making that retains solutions exhibiting different patterns of strengths and weaknesses in order to accommodate change in user preferences, and (4) an evaluation-based iterative algorithm for improving basic design concepts towards successful solutions. This paper introduces the A-Design theory for conceptual engineering design, with some examples to illustrate its general search technique and adaptability. The theory is then implemented on a problem where the goal is to produce configurations for a specified electro-mechanical function. These configurations are composed of actual components chosen from a catalog of various manufacturers, and the topologic connection of those components within the design. This electro-mechanical A-Design process differs from previous work in that (1) designs are not only generated but iteratively improved upon to best meet current objectives specified by a user, (2) synthesis of designs is performed within a rich descriptive representation of components and configurations that models real-world component interactions, and (3) key design alternatives are retained to allow the system the flexibility to adjust to future changes in the problem specifications.

The flexibility in the electro-mechanical A-Design method allows for the production of a variety of solutions for a given design problem. For example in Fig. 1 the user has defined a design problem to create weighing machines (fully described in Section 6) for which the process creates a variety of results with different strengths and weaknesses. In this problem, the user poses three objectives to be optimized within the A-Design process: minimize cost, minimize displacement at the input, and minimize the error in the dial. This figure shows three different designs that were generated by the system and exhibit different strengths in each of the objectives. Through choosing the relative importance of the design goals, the designer can guide the A-Design process as it generates practical and useful alternatives for a given set of needs. The designer's preference for the objectives can be changed throughout the process, in response to the types of designs that are being produced and to changes in the market forces that are driving the process.

In the remainder of this paper, we first describe related work and introduce the A-Design theory. Next we present two examples that illustrate the search strategy and adaptability of the approach. Implementation of A-Design as an electro-mechanical configuration design tool is then presented, and the resulting system is tested on a practical design problem, the design of a weighing machine. Finally, the implications of the technique are discussed.

2. Related Work

The A-Design methodology combines innovations from artificial life, genetic algorithms, stochastic optimization, multi-objective optimization, qualitative physics and asynchronous teams. This section places A-Design within the perspective of these approaches. Table 1 summarizes the four basic subsystems of A-Design, and shows how they relate

Table 1. Derivation and Innovations of the A-Design theory and the electro-mechanical implementation

A-Design subsystems	Agent archecture	Iterative-based search strategy	Design selection	Representation of electro-mechanical systems
Related work	Artificial Life, MADS, A-Teams	Stochastic Optomization, Genetic Algorithms	Pareto Optimality	Qualitative Physics, Functional Grammars
A-Design innovations	Agents are goal-directed and work within a specific A-Design hierarchy where feedback to agents provides for an ever-changing environment	A-Design iteratively improves upon existing design alternatives and creates new ideas through management of maker- and modification-agents	Designs are divided into three unique populations that are used to add adaptability and diversity to design states and to provide a mechanism of feedback for agents.	Two tier representation allows system to both describe behavior of components and instantiate behavior with real-world components.

to each of the different approaches. The third row describes how A-Design has expanded upon or diverged from these research ideas to create the unique constituents of the theory. The following subsections discuss each of these areas as they relate to A-Design.

2.1. Artificial Life

Recent insights on how nature performs complicated problem solving have led to the field of Artificial Life (Langton 1988, Holland 1992). Artificial Life puts computation in the hands of naïve agents who often, through following simple reflexive operations, achieve a complicated emergent behavior. The complex adaptive systems created by the interaction of agents has the benefit of a highly distributed and parallel process that can lead to enhanced robustness and computational speed. Similarly, A-Design's combination of different agent behaviors occurring together within an engineering design setting ideally leads to more creative solutions than would be obtained from a system having one intelligent agent performing all design tasks.

2.2. Multi-Agent Design Systems (MADS) and Asynchronous Teams

Multi-agent systems have previously been applied in a number of engineering design applications (see the overview by Lander 1997). The approach in these multi-agent systems is to use agents to handle the pre- and post-processing of various computational analysis tools such as spreadsheets or CAD systems in order establish a common communication among them. The agents, which communicate through a common framework, act as 'experts' in that they represent the results produced by their encapsulated applications and present it to the design process (see the example in Goldstein 1994). Currently, these approaches offer an unobtrusive approach to communicating between the large computer tools used in concurrent engineering design. Other research projects in this area are incorporating reasoning and learning (Grecu and Brown 1996) into these agents in order to more closely simulate strategies used by human designers.

Asynchronous Teams, or A-Teams (Talukdar 1996), is a computational methodology which combines design utilities such as optimization techniques with autonomous agents. These autonomous agents perform computations independently from other agents, and contribute their results in a parallel and distributed fashion. It is believed that combining programs in a cooperative yet anarchistic manner will produce better results than if programs were executed individually. Talukdar (1993) describes this as synergy: "When the effectiveness of cooperation is so great that a super-object is, in some sense, greater than the sum of its parts, the cooperation is synergistic". The philosophy of the agent-based approach of A-Design is similar to that of A-Teams. Note that A-Teams have also explored the use of a multi-objective selection of designs, as seen in Murthy (1992) and Quadrel et al (1993). These projects were similar to A-Design both in the division of labor in agents, and in the agent preferences for design objectives. What differentiates our work (as will be discussed below) is our functional representation used to model and evaluate design alternatives, our hierarchical and specialized agents that operate upon various subtasks within the design problem, and our management and dynamic modification of agent populations.

2.3. Genetic Algorithms

Genetic algorithms are a popular form of problem solving (see the overviews in Mitchell 1996 and Goldberg 1989), and have been applied to various engineering design problems (e.g. Queipo et al. 1994, Koza et al. 1996, Gage and Kroo 1995). Often, they are used to find solutions in highly constrained situations where the user is more concerned about finding feasible solutions and less about optimizing.

Genetic algorithms synthesize solutions by combining the configurations of other alternatives. These algorithms store many design states simultaneously, as is done in A-Design, in order to compare, propagate and modify alternatives in a similar way to that used in natural evolution. A-Design differs most from these algorithms in that it maintains and evolves both populations of designs and populations of the design creators, the agents. Further, modification of designs in A-Design is determined by goal-directed agents rather than only random mutation and crossover, allowing for a more efficient search of feasible design states by preventing undirected design creation.

2.4. Multi-Objective Optimization

Pareto optimality has been frequently used to address multi-objective optimization problems, for example in Balachandran and Gero (1984) and in Eschenauer et al. (1990). Many algorithms, especially genetic

algorithms, have incorporated this notion of comparing designs (see the overview in Fonseca and Fleming 1995). For example, Schaffer (1985) developed VEGA, a genetic algorithm that incorporates both Pareto optimality and computational search. The combination of Pareto optimality and agents has been explored by Petrie et al. (1995), where existing software tools are controlled by a single governing agent that makes decisions to keep designs based on Pareto optimality. Other than Pareto optimality, multi-objective conceptual design is also tackled through multi-attribute selection (Thurston 1991, D'Ambrosio and Birmingham 1995), and through spectral optimization (Dong and Agogino 1995). Our work uses Pareto optimality as the basis for dividing designs into the three populations of Pareto, good and poor, which is done to give the A-Design system both flexibility and focus on the changing problem specifications as shown in Section 7.

2.5. Stochastic Optimization

Few optimization techniques are able to address the highly nonlinear, discontinuous and multi-modal problems found in engineering design. However, stochastic optimization techniques such as simulated annealing (Kirkpatrick et al. 1983), Tabu search (Glover 1989) and hybrid methods (Fox 1992) have been able to solve these problems with some success. These approaches make numerous perturbations to solutions while evaluating the objective function with each change, thereby allowing the system to move through the design space in search of optimal states. Since these algorithms rely heavily on random moves and statistical behavior, they need to perform numerous iterations to arrive at good results. Optimization of this sort can be costly if the evaluation of each iteration is time-consuming, as is often the case in design problems. A-Design's use of agents attempts to reduce the inefficient moves in the design space by having agents make intelligent directed decisions about how to improve designs. A stochastic influence is still present in the random selection of appropriate agents, and components.

2.6. Conceptual Design Generation and Representation

To implement a conceptual design generation tool for electro-mechanical design, comprehension of functional behavior is important. Much of the work in qualitative physics (see the overview in Forbus 1988) is devoted to producing computer models that understand how the physical world operates on a symbolic or qualitative level. While a few research projects have addressed functionality as a consequence of the geometry of interacting components (Stahovich et al. 1998), most describe components as having non-dimensional influence on one another (Hoover and Rinderle 1989, Gero 1990, Navinchandra et al. 1991, Chandrasekaran et al. 1993). The representation aspect of our work builds primarily on the work of Welch and Dixon (1994) and Schmidt and Cagan (1995), with extensions to allow for real-world components and complex component configurations.

3. A-Design Theory

The A-Design theory presented here is an iterative process that incorporates both populations of designs as well as populations of agents, the creators of the designs. In each iteration, the agents modify the population of designs, and are themselves modified based on the quality of designs they produce. In this manner, A-design captures the evolutionary nature of design, and overcomes multi-modal, non-linear, discontinuous design spaces that are typical of conceptual electro-mechanical design problems. This section describes each of the four parts of the theory. First, we present an overview of the entire iterative process (Section 3.1), followed by a detailed description of the adaptive design selection method (Section 3.2), then the representation of designs (Section 3.3), and finally the creators of the designs, the agents (Section 3.4).

3.1. Iterative Process

Figure 2 shows a flowchart of the overall process. Initially, the system accepts a starting point or description of the problem to be solved. For example, the initial specification for electro-mechanical design is a formal description of input and output behaviors. One class of agents, known as maker-agents, works directly with these input specifications to produce a population of design alternatives. These design states are then evaluated on the various objectives specified by the user. The design sorting method is capable of retaining flexibility in the design process, and allowing for robust selection of designs regardless of the number of objectives. The strategy divides the designs into three populations – Pareto optimal, good and poor – via the design selection process described in the following section. As seen in the figure, the Pareto optimal designs are copied to the next iteration, as well as sent with the good designs to

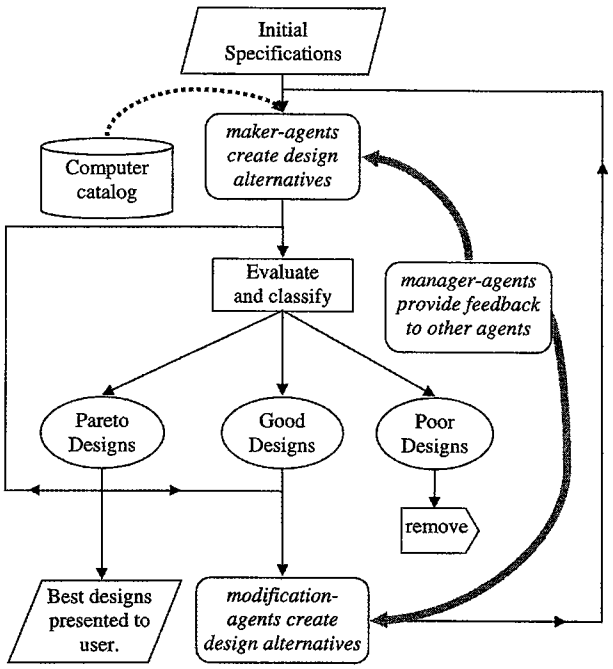

Fig. 2. General flowchart for A-Design theory.

modification-agents. This modification phase of the process involves agents that attempt to improve the solutions and effectively refine the current set of best designs. Poor designs are discarded since they have the least potential to improve future designs, and to make room for additional modified or new designs in future iterations. Throughout the process, manager-agents provide design sorting criteria, feedback to agents, and convergence criteria for the process. As the process unfolds, design states cycle through the exchange between maker-agents and modification-agents until the system converges or resource and time constraints require the acceptance of the current best design.

This methodology is intended for open-ended design problems where there is not a well-defined solution; hence, an in-depth search of the design space and adaptability to user preferences is required. Solutions to the design problem are constantly being improved upon via the selection of designs in a manner similar to that done in genetic algorithms. From a population of design alternatives, the best ones are selected to propagate into the next iteration, while the remaining ones are discarded to make room for new solutions. In genetic algorithms, this selection pressure or 'survival of the fittest' is the primary motivating factor in finding optimally directed designs. A-Design's selection pressure conservatively eliminates only the designs that would never be desired under any user preference. The user's preference, however, is used to guide the process towards solutions that best exemplify the desired trade-offs by propagating a higher concentration of designs that meet the current user preference. This selection of designs is the process' main method of searching the space, and is described in more detail in the next section.

Along with selection pressure, A-Design is also able to search for improved designs through intelligent modification of past alternatives and through feedback provided to agents. The modification-agents do not perform simple mutation and crossover operations as in genetic algorithms, but instead perform intelligent fragmenting or chunking of portions of a design based on the evaluations of the design. For example, a heavy design might have its more massive components removed in an attempt to improve the design when it is reconstructed by the maker-agents.

3.2. Design Selection

In a conceptual design problem where one is considering various component choices and configurations, there are often many goals and constraints that need to be addressed. These can include objectives of various levels of specificity, from the more general such as minimizing weight, cost and size to more specific goals such as minimizing the amount of movement at a given point, or minimizing overshoot in a transient response. Constraints present in a design description can include, for example, preventing excessive pressures in fluid lines or not exceeding maximum temperature requirements. Further, the problem description in conceptual design often changes during the design process, thereby necessitating that a conceptual design algorithm be able to handle multiple objectives in a flexible manner. To do so in A-Design, the process focuses on current user preferences to determine which objectives and constraints are most important, while also maintaining designs that exemplify other relative strengths. These alternatives act as recessive design traits in that they preserve combinations not optimal under the current user preference, but potentially so under other preferences, thus allowing A-Design to quickly respond to changes in the design environment.

The algorithm thus maintains a population of design candidates that is divided into three separate populations labeled poor designs, good designs and Pareto optimal designs, as described below. By using the idea of Pareto optimality, we can mathematically

determine which design alternatives are clearly better than others without simplifying the objectives to only a single scalar, either by a weighted sum or other methods used in utility-theory. The best design for any user preference can be found in a set of designs called the Pareto optimal set, defined as (Eschenauer et al. 1990):

$$x^* \in X \text{ is a Pareto optimal point if and only if there is no other vector } x \in X \text{ such that}$$
$$f_j(x) \leqslant f_j(x^*) \quad \forall j \in \{1, \ldots, m\} \quad \text{and} \quad f_j(x) < f_j(x^*) \; \exists j \in \{1, \ldots, m\} \tag{1}$$

In this equation, f is an objective, m is the total number of objectives, and X is the set of designs. By requiring all objectives to be optimized through minimization, it is necessary to note that no other design should exist that has more minimal values for the total of the objectives presented. This can be visually determined through plotting alternatives on a graph, with the axes representing different objectives, as in Fig. 3(a). If only two objectives are to be optimized, one can view design alternatives on a two-dimensional grid, and the Pareto optimal designs can easily be determined. These designs form a 'front' of solutions that are close to the coordinate axes. Despite the difficulty in viewing the Pareto optimal set for higher dimensions, the designs can still be found by applying the mathematical criteria stated above.

The essence of the A-Design process is to propagate this Pareto front to constantly find better solutions, as seen in Fig. 3(b). However, at any given time in the process, the user might change the relative importance of objectives. As mentioned above, the multi-objective selection is useful in handling large numbers of objectives or constraints in a more robust manner, but it is also useful in tailoring designs for specific designer needs or consumer markets. Imagine a situation in which performance and cost are two competing objectives in a conceptual design problem. A-Design might be initiated to conceptualize designs, with more importance given to minimizing cost than maximizing performance. The process concentrates its effort on this user preference, but by maintaining the complete Pareto optimal set, the system can accommodate changes to this preference. The user preference can be implemented in a variety of ways. In the current implementation, the user defines a linear weighting; however, multi-attribute decision-making (Thurston 1991), for example, could also be used.

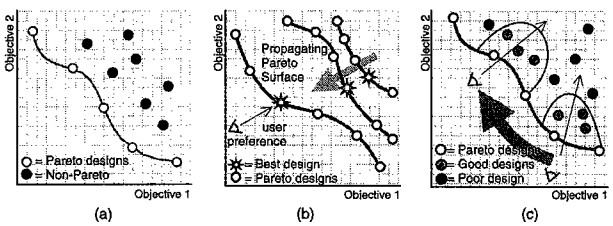

Fig. 3. (a) Two-dimensional plot of designs depicting pareto optimal set, (b) propagating Pareto surface while considering user preference, (c) bulb of good populations move with user preference.

Outside of Pareto optimal designs, the system further divides solutions into good and poor designs. At a given time, A-Design concentrates its effort on improving designs that best meet the user preference. Some designs, while not Pareto optimal, might better meet user preference than some of the outlying Pareto optimal alternatives that are preserved for extreme changes in user preference. This set of preferred but non-Pareto designs makes up the good population which, for example, can be visualized as a set of solutions located within a given radius of the intersection of current user preference and the Pareto set (Fig. 3(c)). In general, the good designs are a user-defined percentage (25% in the current implementation) of the non-Pareto designs that best meet the user preference. By concentrating on improving the set of good designs, A-Design focuses on the desired user preference. If the preference changes, then the location of the good population with respect to the Pareto optimal set will also change. The purpose of the good designs is to propagate alternatives which might not currently be optimal, but which through some modification might prove to be a successful design alternative. The good population serves to add diversity in design configurations as opposed to diversity in objective values, as is done by the Pareto optimal designs. The remaining poor designs are discarded by the system to make room for new design states in subsequent iterations.

Another important feature of the division of designs is to track the agents that created them. We increase the prevalence of agents that create Pareto and good designs so that, in future iterations design activity will produce more designs like the current Pareto and good set. Conversely, agents that produce poor designs are penalized in order to eliminate design activity that leads to inferior design states. Manager-agents control this feedback mechanism, which modifies the maker-agent and modification-agent populations; in future, more sophisticated feedback will be explored.

This division of designs into Pareto, good and poor is thus the process' method of keeping track of useful alternatives, as well as providing feedback to agents. It is this division and the propagation of the Pareto

front that provides the foundation for our approach to maintaining recessive characteristics, providing a pool of designs for possible evolution, and influencing agent selection.

3.3. Design Representation

It is necessary in any computational process to have a formal description of the artifact being manipulated, created or optimized. Traditionally, in engineering optimization, the representation is a simple vector of a fixed length, set prior to execution, containing variables representing different physical attributes of a design. The perturbation of this vector, $\mathbf{x}$, is often done either randomly or through knowledge of previous evaluations of $\mathbf{x}$. In the case of genetic algorithms, for example, the $\mathbf{x}$ is often expanded to a bit string to represent a genotype encoding. The manner in which the string is manipulated can have unpredictable or far-reaching effects in the phenotype or physical construction of the design. The unsystematic changes that can result by manipulating the design encoding make logical modification of the design state difficult.

It is for this reason that we develop a description of the design state that is most natural to modeling the actual device, and not a description which is merely beneficial for the manipulation and creation of designs through computational methods. As will be seen in Section 5.2, an electro-mechanical representation is developed to model a range of electro-mechanical devices, and does not depend upon the manner of constructing or altering designs. Allowing more freedom in constructing computational representations, be it through data structures or variable length vectors, allows design synthesis to be more general without the restrictions imposed by the construction process. The drawback of having a more general design description, however, is the need for more involved techniques for configuring designs, hence agents are used to act as a buffer between the iterative process and the representation.

3.4. Agents

The creation of any one design in A-design is due to collaboration among several different agents. These agents contain knowledge of how to design based on their individual strategies and preferences. They are constructed to understand the representation of a design state, be it complete or incomplete, and contribute in a manner that leads to successful solutions. The strategies used by these agents are based on deterministic algorithms, such as tree-searching or pattern matching. In the current implementation, agents are not autonomous, but are triggered by the system or by other agents. A simple hierarchy exists where one set of agents, known as manager-agents, invokes the operations of the other agent classes: the maker- and modification-agents. Within each agent class subclasses can exist, as well as different agent types within each subclass, and populations of agents of the same agent type. For example, the electro-mechanical A-Design approach shown in Section 5.3 has two subclasses of maker-agents, Configuration-agents (C-agents) and Instantiation-agents (I-agents); there are 32 different C-agent types and six different I-agent types, and within a given iteration of A-Design there is an average population of 3000 maker-agents in all.

The maker-agents have two responsibilities: create design alternatives based on the problem description; and re-build designs returned by the modification-agents. Construction of a complete design state is accomplished by several maker-agents, which each add their distinctive parts to make a complete state as prescribed by the design representation. The modification-agents are active at the end of the evaluation phase of the process (see Fig. 2) to take design states from the Pareto and good populations and improve them based on how they were evaluated. These agents allow the process to move from current design states to possibly better ones. As seen in Fig. 3(b), the modifications performed by these agents can significantly advance the Pareto optimal front.

While the maker- and modification-agents perform directed strategies to construct or improve designs, the manager-agents add both a stochastic and learning aspect to the A-Design system. Initially, all maker- and modification-agents have the same population, and the manager-agents randomly invoke these agents until all design tasks for the given iteration are accomplished. At the end of the iteration, the manager-agents analyze the designs created by these agents and provide feedback about how to improve the design process according to the user's preferences. Currently, a single manager agent is implemented which provides feedback in the form of increasing or decreasing the population of particular agent types, thereby forcing the agents to compete for survival by contributing to better designs.

There is some ambiguity in computational research over the proper use of the word 'agent'. Our use of agent is consistent with the definition of Russell and Norvig (1995), where agents are viewed as perceiving their environment (here, the design state) through sensors (i.e. functional inputs) and acting upon their

environment through effectors (i.e. modifications to the design state). While some might argue that without displaying specific behaviors of autonomy, mobility or sociability (Sycara 1998, Franklin and Graesser 1996), the agents in A-Design do not fully conform to the definition of 'agency'. However, within the Artificial Life community, agents often are defined as simple strategies that when combined lead to a more complex emergent behavior (Langton 1988). Similarly, our agents are defined as knowledge-based strategies for solving open-ended problems that, when cooperatively combined with other similar strategies, leads to a more complex and often emergent behavior for achieving the design goal. For example, in designing travel paths between two set destinations, one can imagine different agents to accomplish the goal. One agent might try to determine the best path for driving while another might look at possible flight paths for connecting the start and end points. The various agents would accomplish the same goal but in different manners. Because, in design, there is no single or clear-cut answer, different agents working on the same design problem can generate completely different solutions. By having agents with different abilities contributing to designs, the process gains robustness and variety in solving various conceptual design problems.

4. A-Design as Search Strategy

The following two test examples explore the effectiveness of the A-Design theory as a computational search tool. They were created to test individual sections of the theory, and are presented here to illustrate the capabilities of the methodology. The first demonstrates the operation of the multi-objective design selection and interaction of agent subsystems, while the second challenges A-Design with a numerical optimization problem. In each of these examples, the design representation and agents are quite simple. After this exploration of the operation of the general components of A-Design, a richer representation and set of agents is introduced in Section 5 for the conceptual design problem of electro-mechanical configuration design.

4.1. Manhattan Transfer

This example presents a problem for which it is relatively easy to find solutions, but difficult to find solutions that maximize the satisfaction of the traveler's preference. The object of the Manhattan Transfer problem is to get from one location to another in a grid-based city in the minimum amount of time, cost and effort. A user specifies the start and end locations of a trip as the initial specification to the algorithm. It is the algorithm's duty to find solutions that connect the start and end locations via various transportation media. A simulated two-dimensional grid of squares represents city blocks, while the transportation devices consist of bike, walk, run, bus, taxi and subway; each having different values for the cost, time and effort required. The problem specification requires A-Design to create alternatives using combinations of the six transportation devices in order to best satisfy the user's weighted criteria of minimizing cost, time and effort.

The process starts as in Fig. 2 with the maker-agents contributing partial travel paths along the way to the creation of complete trips. Rather than each agent solving the complete problem from start to end, the maker-agents subgoal on shorter trips that combine to make a complete solution. Therefore, all design states are the result of the combined effort of several maker-agents. These maker-agents differ in which transportation device they add to a trip and how they handle constraints in the system such as bus and subway stops, and maximum distances one can walk or run. After maker-agents complete designs, the designs are evaluated on their cost, time and effort, and solutions are sorted into Pareto, good and poor populations. Next, modification-agents remove elements from solutions that are believed to be preventing designs from reaching optimal states and return the fragmented designs back to the maker-agents for reconstruction. After modification, another iteration ensues until the process is finished and returns to the user a solution represented as a list of transportation devices with their start and stop locations making up the complete travel plan.

The Manhattan Transfer results are generated by specifying the beginning (0, 0) and end (20, 20) locations, as well as the relative importance of each objective. At the end of the process, A-Design returns several solutions that best meet the user preference. For example, if the user feels that minimizing cost is more important than minimizing time or effort, a solution such as the one shown in Fig. 4 is suggested. This solution was generated for a user whose preference for cost is five times more important than time and two times more important than effort. The design was found in 62 iterations with a maximum design population of 160. In addition, the user is free to change preference throughout the process, allowing the system to adapt appropriately. By examining the results produced, the user can adjust the preference weighting to achieve the desired trip. In this example,

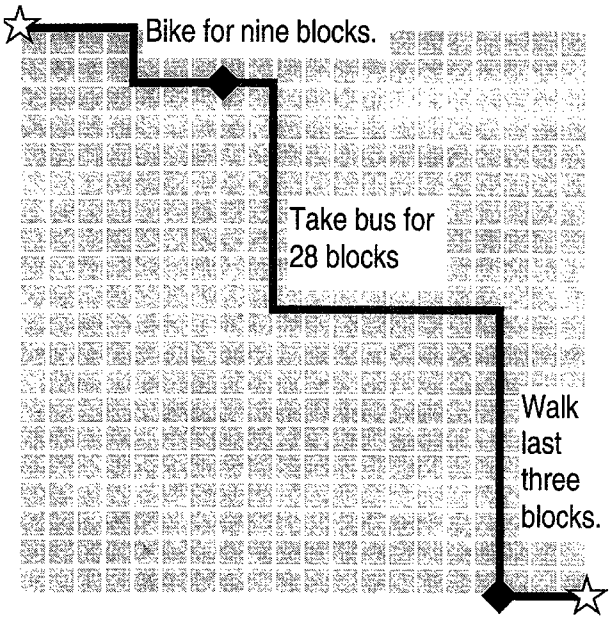

Fig. 4. Possible solution for Manhattan Transfer problem.

user preference was changed midway through the process to prefer minimizing time twice as much as effort, and five times as much as cost, thus producing the result of 'take taxi from start to finish'. By using the population of designs created under the old preference, which includes a variety of designs, the process is easily adapted to this new preference. As a result, the algorithm took only 11 iterations to converge, thus illustrating the power of the recessive traits stored in the Pareto set. It was determined through a separate experiment that the complete space of uniquely evaluatable design states numbers approximately 1.2 million, and that of these, only 99 are Pareto optimal. In comparison, A-Design found 40 Pareto points of the 99 while searching only 0.6% (approximately 7000 design states) of the design space.

Although the Manhattan Transfer example deals with a highly abstracted situation, it illustrates that agents with different characteristics are able to work together to find optimal design configurations, and their interaction produces a flexibility in adapting to changes. Alone, the agents consider minimizing only single objectives. For example, low-cost designs are usually created by agents that prefer low-cost devices. Through the collaboration of different agent types, solutions are constructed that exemplify the preferences suggested by the user. The combination of agents that prefer low-cost devices with agents that prefer low-effort devices will ideally lead to designs that are both low in cost and low in effort. In the example, the initial user preference was for low-cost designs, but when the preference changed to emphasize quick trips, A-Design was able to accommodate this alteration and switch focus to agents with a preference for quick designs. The example also demonstrates the algorithm's effectiveness in advancing the Pareto optimal front through the various iterations. Early iterations produce a set of alternatives that through modification lead to better states, which in turn add new solutions to the Pareto surface and remove previous ones that are shown to be suboptimal.

4.2. Numerical Optimization

In addition to A-Design's ability to adapt to changes in user preference is the need for A-Design to optimize the various objectives specified by the user. This example, therefore, tests the optimizing power of A-Design apart from its use as a conceptual design generation tool. In its final form, A-Design creates designs to best meet the objectives and constraints specified by a user, and is thus driven toward optimally directed configurations.

This example sets up two numerical functions to be optimized in order to maintain a multi-objective problem. The two functions shown in Fig. 5, are expressed by

$$f_1(x,y) = 300(1-x)^2 e^{-x^2-(y+1)^2} - 1000\left(\frac{x}{5} - x^3 - y^5\right)e^{-x^2-y^2} - \frac{100}{3}e^{-(x+1)^2-y^2} - |x| - |y| \quad (2)$$

where the minimum is -59.2 and is found at ($x = 0.023$, $y = -2.105$), and

$$f_2(x,y) = x^2 + y^2 - 100\sin(x) - 100\sin(y) \quad (3)$$

where the minimum is -234.6 and is found at ($x = -1.57$, $y = -1.57$).

These functions are both highly multi-modal, and therefore difficult to solve by traditional methods. A-Design was compared with a robust SQP algorithm (Lawrence et al. 1993) in finding an optimum for a weighted sum of the two objectives. Both systems were first tested by weighting f_1 ten-to-one over f_2, and then tested again with a weighting of f_2 ten-to-one over f_1 using the results of the first run as the initial specifications for the second run. The 10:1 and 1:10 preferences are used to establish quite different optimal points for the lumped objective functions. Agents within this problem are simple functions that increase or decrease parameters within the equations to reduce objective values or avoid local minima.

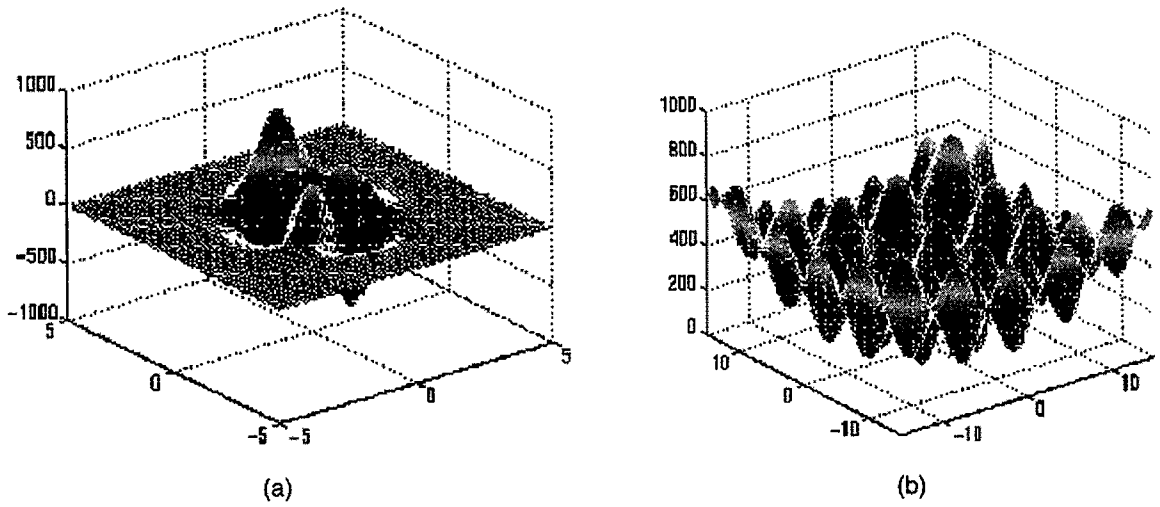

Fig. 5. Numerical optimization test functions: (a) f_1: peaks function, (b) f_2: parabola plus sine function.

A single run of A-Design proved to be a time-consuming process, partly due to the number of design state evaluations and partly due to agent and design management. Although SQP found the optimal solution quicker for the 10:1 weighting, it successfully found the global minimum only one out of seven times. The A-Design algorithm, due to its agent-based search and storing of design alternatives, found the solution in every run with a population size of 100 and an average of 88 iterations. When the weightings changed, A-Design only needed to perform a single iteration to arrive at the new optimum, while SQP had to be rerun 13 times before finding the new optimum. By retaining the Pareto optimal set from the first run, the A-Design algorithm quickly adapts to changes in weighting of the two objectives. Certainly, SQP is a more efficient optimization strategy if it starts in the neighborhood of the optimum. However, A-Design is, in general, more robust in its ability to find the optimum and the time required in arriving at Pareto optimal solutions for the first user preference results in a large savings when the process is reinitiated with different weights.

The two examples discussed above demonstrate the versatility of the A-Design methodology. Both examples illustrate the methodology's potential outside the intended use as a conceptual design tool. These examples show that A-Design, through its unique design selection scheme of preserving the Pareto front and iterative-based agent operations, can successfully produce results for a wide variety of problems.

5. Electro-Mechanical Configuration Design

5.1. Overview

Since the philosophy behind A-Design is to aid or even automate portions of the conceptual design process, it is important that we construct an example which captures the open-endedness of true design. The previous two examples simplify the design process to optimizing search, but the real design process is much more complicated than searching for a single global optimal state. Design is an ever-changing process where not only does the space of relevant designs change, but also the means in which those designs are evaluated.

This section describes a completed implementation of A-Design that creates electro-mechanical configuration designs for a general class of user-defined problems. In Fig. 6, a detailed flowchart for the electro-mechanical A-Design process is presented. Note that it is an augmentation of the general A-Design flowchart presented in Fig. 2. Initially, the user defines a design problem by the functional description of the inputs and outputs of an electro-mechanical device as the starting point for the algorithm; it is assumed that the device to be designed is fully describable by the inputs and outputs. Maker-agents are divided into two groups: configuration-agents (C-agents) and instantiation-agents (I-agents). The configuration-agents reason about how these inputs and outputs can be successfully connected to make conceptual design alternatives. These config-