ABSTRACT

Evolutionary computation (EC) has been successfully applied to a wide range of design problems. There has also been an abundant amount of work in applying interactive ECs in the design of displays, robot behavior, bitmaps, etc. In the EC literature, one can also see a number of successful design applications of distributed ECs. However, to date, there has been no research in the area of interactive distributed ECs. In this paper, we present an interactive distributed evolutionary algorithm (IDEA) for the design of simple emoticons. We will discuss a variety of ways that our IDEA is currently being used including the areas of EC education, Human Factors, and the modeling of human problem solving.

Keywords: Interactive Evolutionary Computation, Distributed Evolutionary Computation, Interactive Distributed Evolutionary Algorithm

1.0 Introduction


Figure 1 provides a pseudo-code example of an EC (also known as an evolutionary algorithm or EA). Initially a population of CSs (referred to as individuals) are randomly generated and evaluated using an evaluation function. The evalution function assigns each individual a fitness that is representative of the ‘goodness’ of that particular individual. After the initial population has been
created and evaluated the EC begins to iterative refine the population by: a) selecting parents based on their assigned fitness, b) allowing the selected parents to create offspring through crossover (sexual reproduction) and/or mutation (asexual reproduction), c) evaluating the offspring, and d) determining which individuals of the current population and the set of offspring get to survive to the next generation.

```plaintext
Procedure EC{
    t = 0;
    Initialize Pop(t);
    Evaluate Pop(t);
    While (Not Done)
        { Parents(t) = Select_Parents(Pop(t));
          Offspring(t) = Procreate(Parents(t));
          Evaluate(Offspring(t));
          Pop(t+1)= Replace(Pop(t),Offspring(t));
          t = t + 1;
        }
}
```

Figure 1: Pseudo Code Version of an EC

In this paper, we present an interactive distributed evolutionary algorithm (IDEA) that is used to evolve simple emoticons. In this paper, we will also discuss a variety of ways that our IDEA is currently being used in the areas of EC education, Human Factors, and the modeling of human problem solving.

2.0 Background

2.1 The Evolution of Evolutionary Computation

The founding of the field of EC is typically accredited to four individuals: Ingo Rechenberg & Hans-Paul Schwefel [Bäck 1991], John Holland [Holland 1975], and Lawrence Fogel [Fogel 1966]. Each of these founding fathers saw biologically-inspired search paradigms as a solution to the specific problems that they were independently attempting to solve. Rechenberg & Schwefel saw simulated evolution as means of solving complex parameter optimization problems [Bäck et al. 1998, Bäck 1991, De Jong & Spears 1991]. They allowed this problem to dictate the development of their solution and as a result invented Evolution Strategies (ESs) [back1, back2]. Similarly, Holland and Fogel also saw simulated evolution as a means of solving their particular problems of interest [De Jong & Spears 1991, Fogel 2000, Fogel 1966, Goldberg 1989, Holland 1975, Spears et al. 1993]. As a result Holland invented the concept of Genetic Algorithms (GAs) as a means of being robust adaptive systems and Fogel invented the concept of Evolutionary Programming (EP) as a means of solving prediction problems. Originally researchers of these first generation ECs remained isolated from one another until the mid-1980s [Dozier et al. 2001]. After this point, the field of EC has seen a great deal of growth as researchers have built upon the
foundation developed by the EC patriarchs. During this period of dramatic growth the field of EC itself has evolved.

The late 1980s and early 1990s saw the birth of a number of second generation ECs. These new paradigms departed from the traditional representations of simple representations of CSs (such as simple binary or real-vector CSs) and embraced more complex representations based on user-defined data structures [Michalewicz 1994], neural networks [de Garis 1990], and LISP programs [Koza 1992]. These concepts were known as Evolution Programs developed by Michalewicz and Genetic Programming developed by Hugo deGaris [deGaris 1990] and John Koza [Koza 1992]. Also a number of EC researchers began developing distributed and parallel forms of EC [Tanese 1989, Brown et al. 1989].

A third generation of EC techniques has enjoyed an increased amount of interest by EC researchers. This generation has seen the development of EC paradigms that are composed of agents rather than complex representations. These agent-based ECs are different in a number of ways. First, agents making-up a population were ‘alive’ and active. In these techniques, a single agent possesses the ability to solve the problem at hand. Secondly, although the agents are sophisticated enough to solve a problem by themselves, they use adaptive external memory structures in a effort the communicate to other agents promising and/or futile regions of the search space which increases the speed and search efficiency of the agent society as a whole. This concentration on simulated sociological evolution as opposed to simulated biological evolution is a major departure from the ECs of the earlier generations. Third generation paradigms of this form collectively comprise a sub-area of EC commonly referred to as Swarm Intelligence (SI) [Dorigo & Gambardella 1997, Kennedy & Eberhart 2001, Sebag & Shoenauer 1997]. Other third generation EC techniques that are not members of SI include DNA-based computing [Adleman], memetic algorithms [Moscato 1989], artificial immune systems (AISs) [Forrest 1996], and interactive evolutionary computation [Takagi 2001].

2.2 Interactive Evolutionary Computation

An Interactive EC (IEC) is an EC that replaces one or more of the evolutionary processes of selection, procreation, and/or evaluation with a human [Takagi 2001]. IECs have been successfully applied to a wide range of problems such as the design of displays, robot behavior engineering, bitmaps, etc. However, to date, there has been no research in the area of interactive distributed ECs.

2.3 Distributed Evolutionary Computation

Distributed Evolutionary Computations (DECs) typically fall into one of three categories: function-based (Brown et al. 1989), domain-based (Belding 1995, De Jong & Sarma 1995, Tanese 1989), or variable-based (Potter et al. 1995, Dozier 2003). Function-based DEAs distribute tasks (or functions) of the evolutionary process (selection, procreation, evaluation) among k processors in order to speed-up the processing time of a single EC.

Domain-based DECs distribute a population of P candidate solutions (CSs) among k processors. There are two types of DBDECs: coarse-grained and fine-grained. In coarse-grained domain-
based DECs, $k$ sub-populations of $P/k$ CSs are maintained. Thus, $k$ ECs are executed in parallel. Periodically, a user-specified number CSs of each sub-population are allowed to migrate to other sub-populations. Fine-grained domain-based DECs usually assign one CS to each processor. The sub-populations or demes overlap as CSs are only allowed to mate and compete for survival within their geographical neighborhoods. In variable-based DECs, the structure representing CSs is distributed. Let $V$ represent the variables that form CSs of a problem. Variable-based DECs, distribute $|V|/k$ variables among $k$ processors. Each processor uses an EC to evolve a population of $P$ partial CSs.

### 3.0 An IDEA for Emoticon Design

IDEA is a web-based client/server application that allows a number of users to interactively (and asynchronously) design emoticons. Figure 2 provides a high level view of IDEA. In IDEA, candidate emoticons are exchanged to between users through the use of a meme space which is simply an array of user selected emoticons.

![Figure 2: A High Level View of IDEA](image)

In Figure 3, a pseudo-code version of an IDEA client application is provided. An IDEA client works as follows. Initially, nine candidate emoticons are randomly generated and presented to the user. The user is then allowed to select an emoticon, $e$, that they prefer best as well as select the type of mutation, $o$, that they think would lead to set of more suitable emoticons. The se-
lected emoticon, $e$, is then sent to the server to be added to meme space and a randomly selected emoticon from meme space, $m$, is then sent to the client.

![Figure 3: Pseudo-Code for an IDEA Client](image)

```
Procedure IDEA_Client{
    t = 0;
    Initialize Pop(t) // Randomly Generate 9 Emoticons;
    Present Pop(t) to User;
    While (Not Done)
    {
        Allow_User_to_Select_An_Emoticon(e);
        Allow_User_to_Select_A_Mutation_Op(o);
        Send_to_Meme_Space(e);
        Receive_From_Meme_Space(m);
        Parents(t) = {e, m};
        Offspring(t) = {
            Create_4_Mutants(e, o);
            Create_3_Recombinants(e, m, o);
        }
        Pop(t+1) = Parents(t) $\cup$ Offspring(t);
        t = t + 1;
    }
}
```

The emoticons, $e$ and $m$, serve as parents for 7 offspring. Four offspring are created via the user selected mutation operator, $o$, and 3 offspring are created as a result of recombining $e$ and $m$ using blend crossover (Eshelman & Shaffer 1993) along with the mutation operator $o$. Finally, the two parents and their 7 offspring comprise the next generation of emoticons and are presented to the user. This process of allowing the user to select an emoticon and a mutation operator, receiving an emoticon from meme space an allowing the two emoticons to create 7 offspring is repeated until the user has designed an acceptable emoticon.

When the user discovers an acceptable emoticon, they select it and click on the ‘save’ option of the user interface. This option will send a copy of the emoticon to the server. To sever the connection between the client and server, one can click on the ‘done’ option. After the connection has been severed the user may continue to evolve the population of emoticons; however, emoti-
cons will no longer migrate of to and from meme space. Figure 4 provides an example of the IDEA user interface.

![IDEA User Interface](image)

Figure 4: The IDEA User Interface

### 3.1 Representation of a Candidate Emoticon

A candidate emoticon is represented as vector of 11 integer variables. This vector contains the y-coordinates of eleven points on the emoticon. In Figure 5, a template of a candidate emoticon is shown. Notice that the first three variables correspond to the left eyebrow, the next three represent the right eyebrow, and the last five values are used to represent the mouth. Of course, the x-coordinates are fixed. The first six variables can be assigned values within the interval, [35..50] while the last five variables can be assign values within the interval, [95..115]. Therefore, the size of emoticon space is $3.645 \times 10^{13}$.

In order to display a candidate emoticon, eight lines are drawn. The first two lines make up the left eyebrow, \(\text{line}(<x_0,e_0>,<x_1,e_1>)\) and \(\text{line}(<x_1,e_1>,<x_2,e_2>)\) where \(\text{line}(p1,p2)\) represents a line from point p1 to point p2. Also, two lines are used to display the left eyebrow, \(\text{line}(<x_3,e_3>,<x_4,e_4>)\) and \(\text{line}(<x_4,e_4>,<x_5,e_5>)\). In order to display the mouth, four lines are used, \(\text{line}(<x_6,e_6>,<x_7,e_7>)\), \(\text{line}(<x_7,e_7>,<x_8,e_8>)\), \(\text{line}(<x_8,e_8>,<x_9,e_9>)\), and \(\text{line}(<x_{10},e_{10}>,<x_{10},e_{10}>)\).
3.2 Mutation Operators

An IDEA client allows a user to make mutations of a selected emoticon as follows:

$$e_i' = e_i + \sigma_k N(0,1),$$

where $e$ represent a selected emoticon and let $e'$ represent an offspring and where $\sigma_k$ represents the type of mutation operator selected. In the current version of the IDEA user interface, the ‘Small Changes’ button corresponds to $\sigma_1 = 0.1$, the ‘Medium Changes’ button corresponds to $\sigma_2 = 0.25$, and the ‘Large Changes’ button corresponds to $\sigma_3 = 0.5$. After each offspring is created it is checked to see if all assigned values are within their pre-specified domains ([35..50] for $e_0$ to $e_5$ and [95..115] for $e_6$ to $e_{10}$). If the value assigned to a variable of an offspring is smaller than the lower bound for that variable, then that variable is assigned the value of the lower bound. Similarly, if the value assigned to a variable of an offspring is larger than upper bound for that variable, then that variable is assigned the value of the upper bound.

3.3 Meme Space & Migration Scheme

As shown previously in Figure 2, an important part of IDEA is a structure referred to as the meme space. The meme space is simply an array of the $s$ most recent emoticons that have been selected by the IDEA users, where $s$ is referred to as the meme space size. The value of $s$ can determine the quality of IDEA search. If $s$ is too small (i.e. $s = 1$) then the users do not exchange
any emoticons. However, if $s$ is too large, then the users have a greater probability of receiving emoticons that are too primitive relative the overall best designs.

The IDEA migration scheme is straight-forward. Each time the IDEA server receives an emoticon from a particular user it returns a randomly selected emoticon from meme space.

### 4.0 Emoticon Design Experiments

For the results presented in this paper, 3 emoticon design experiments were conducted. Fourteen undergraduate students in the department of Industrial Systems Engineering at Auburn University were selected to participate in these experiments. These students were asked to design 3 emoticons: (a) a smiley emoticon, (b) an angry emoticon, and (c) an emoticon for ‘hand-in-gear’ (an emoticon for intense pain). In this study, 14 students used IDEA with a meme space size of 35 (which is 2.5 times the number of users), while 8 students used IDEA with a meme space size of 1. Remember that when $s = 1$ no emoticons get exchanged between users; each user just receives the emoticon that was sent to the server. For the “smile” and “hand-in-gear” conditions, students were provided with visual referents as a basis for designing a particular face. The visual referent for the “smile” condition is shown in Figure 6.

![Figure 6: Referent for the Smile Emoticon Design Experiment](image)

Under the “smile” condition, subjects were asked to use the IDEA system to develop a face that they felt best matched the face depicted in Figure 6. For the “hand-in-gear” visual referent, subjects were presented with a machine-guarding safety symbol from the American National Standard Institute (ANSI 2002). This symbol is shown in Figure 7.
Under the “hand-in-gear” referent condition, subjects were asked to use the IDEA to develop a face that they felt best matched the safety symbol depicted in Figure 7. For the anger condition, subjects used a verbal referent as the basis of the face that they would develop. The verbal referent was a printed paragraph in which Izard (1977) described the emotion of anger. For each condition, once subjects found a face that they felt was most appropriate for the referent, they saved each face’s design specifications to a file.

5.0 Results

5.1 Comparison Between the Two Groups

Figure 7 shows the results of the three experiments. A decision tree learning algorithm, C4.5, (Quinlan, 1993) was used to learn the difference between emoticons developed by both groups. In Figure 7, the variables used to develop a decision tree for classifying resulting emoticons as either from the networked group or the group of individuals, are shown along with their associated values. Below each set of variables is the classifications made by C4.5. The difference in the emoticons designed by both groups is statistically significant.

For each of the three emoticons designed, the emoticons designed by the separated group were correctly classified. For the networked group, the C4.5 decision tree made 5 classification errors on the smile emoticons, 3 classification errors on the anger emoticon, and 7 classification errors
on the ‘hand-in-gear’ emoticon. We believe the reason for the miss classifications is mostly likely due to a subset of the networked users working at a different pace. Thus, it may be the case that there are really two networked subgroups that emerged: one that worked faster and another that worked slower.

![Facial Parameter Patterns Discovered by C4.5 Algorithm](image)

**Figure 6: Results of the Three Experiments**

### 5.2 Factor Analysis of the Networked IDEA Users

In addition to C4.5, factor analysis (Gorsuch 1983) coupled with paired t-Tests were used to identify which design parameters of the face changed significantly as the 14 IDEA users evolved faces for each of the three referents.

Factor analysis was performed on a data set comprised of 42 (14 subjects × 3 referents) face vectors. Each vector contained values for the 11 design parameters that comprised the corresponding facial expression. Results of the factor analysis, in terms of the design parameter communalities and factor loadings, are summarized in Table 1.

<table>
<thead>
<tr>
<th>Design Parameter</th>
<th>Communality</th>
<th>Factor 1</th>
<th>Factor 2</th>
<th>Factor 3</th>
<th>Factor 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>e0 (Left Eyebrow)</td>
<td>0.67</td>
<td>0.07</td>
<td><strong>0.72</strong></td>
<td>0.25</td>
<td>0.29</td>
</tr>
<tr>
<td>e1 (Left Eyebrow)</td>
<td>0.63</td>
<td><strong>0.79</strong></td>
<td>-0.05</td>
<td>-0.01</td>
<td>-0.04</td>
</tr>
<tr>
<td>e2 (Left Eyebrow)</td>
<td>0.49</td>
<td><strong>0.47</strong></td>
<td>-0.40</td>
<td>-0.32</td>
<td>0.10</td>
</tr>
<tr>
<td>e3 (Right Eyebrow)</td>
<td>0.59</td>
<td>0.13</td>
<td><strong>-0.73</strong></td>
<td>0.03</td>
<td>-0.18</td>
</tr>
<tr>
<td>e4 (Right Eyebrow)</td>
<td>0.76</td>
<td><strong>0.84</strong></td>
<td>-0.21</td>
<td>0.02</td>
<td>0.03</td>
</tr>
<tr>
<td>e5 (Right Eyebrow)</td>
<td>0.79</td>
<td>-0.24</td>
<td><strong>0.79</strong></td>
<td>-0.02</td>
<td>-0.33</td>
</tr>
<tr>
<td>e6 (Mouth)</td>
<td>0.70</td>
<td><strong>0.62</strong></td>
<td>0.07</td>
<td>-0.48</td>
<td>-0.29</td>
</tr>
<tr>
<td>e7 (Mouth)</td>
<td>0.66</td>
<td>-0.21</td>
<td>0.29</td>
<td><strong>0.73</strong></td>
<td>0.04</td>
</tr>
<tr>
<td>e8 (Mouth)</td>
<td>0.85</td>
<td>-0.07</td>
<td>0.11</td>
<td>0.01</td>
<td><strong>0.91</strong></td>
</tr>
<tr>
<td>e9 (Mouth)</td>
<td>0.74</td>
<td>0.16</td>
<td>-0.11</td>
<td><strong>0.84</strong></td>
<td>-0.07</td>
</tr>
<tr>
<td>e10 (Mouth)</td>
<td>0.57</td>
<td>0.46</td>
<td>-0.13</td>
<td><strong>-0.57</strong></td>
<td>-0.14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fits Pattern Does Not Fit</th>
<th>p = 0.0015</th>
<th>9</th>
<th>0</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Puts Pattern Does Not Fit</td>
<td>p = 0.0015</td>
<td>11</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Fits Pattern Does Not Fit</td>
<td>p = 0.02</td>
<td>7</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
As shown in Table 1, factor analysis of the facial design parameter vectors revealed a 4-factor structure that accounted for 49-85 percent of the variability in the individual face design parameters. Factor loadings from the analysis indicated the following:

- Increases in the Factor 1 subscale score would be due primarily to increases in the $e_1$, $e_2$, $e_4$, and $e_6$ facial design parameter values.
- Increases in the Factor 2 subscale score would be due primarily to increases in the $e_0$ and $e_3$ parameter values and/or a decrease in $e_5$ parameter value.
- Increases in the Factor 3 subscale score would be due primarily to increases in the $e_7$ and $e_9$ parameter values and/or a decrease in the $e_{10}$ parameter values.
- Increases in the Factor 4 subscale score would be due primarily to an increase in the $e_8$ parameter value.

A series of pair-wise t-tests were used to evaluate the within-subject changes in these factor subscale scores across the three referents. The results of this testing are summarized in Table 2.

<table>
<thead>
<tr>
<th>Pair-wise Comparison</th>
<th>Mean Difference (Standard Deviation)</th>
<th>Pair-wise T Statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1: Smile-Anger</td>
<td>-1.51 (0.74)</td>
<td>-7.65</td>
<td>&lt; 0.001*</td>
</tr>
<tr>
<td>Smile-Gear</td>
<td>-1.00 (1.03)</td>
<td>-3.65</td>
<td>0.003*</td>
</tr>
<tr>
<td>Anger-Gear</td>
<td>0.51 (1.20)</td>
<td>1.58</td>
<td>0.14</td>
</tr>
<tr>
<td>Factor 2: Smile-Anger</td>
<td>1.01 (0.89)</td>
<td>4.26</td>
<td>0.001*</td>
</tr>
<tr>
<td>Smile-Gear</td>
<td>-0.53 (1.38)</td>
<td>-1.44</td>
<td>0.17</td>
</tr>
<tr>
<td>Anger-Gear</td>
<td>-1.54 (1.17)</td>
<td>-4.92</td>
<td>&lt; 0.001*</td>
</tr>
<tr>
<td>Factor 3: Smile-Anger</td>
<td>0.97 (0.92)</td>
<td>3.96</td>
<td>0.002*</td>
</tr>
<tr>
<td>Smile-Gear</td>
<td>1.16 (1.34)</td>
<td>3.25</td>
<td>0.006*</td>
</tr>
<tr>
<td>Anger-Gear</td>
<td>0.19 (1.03)</td>
<td>0.71</td>
<td>0.49</td>
</tr>
<tr>
<td>Factor 4: Smile-Anger</td>
<td>0.69 (1.06)</td>
<td>2.41</td>
<td>0.03**</td>
</tr>
<tr>
<td>Smile-Gear</td>
<td>1.00 (1.12)</td>
<td>3.34</td>
<td>0.005*</td>
</tr>
<tr>
<td>Anger-Gear</td>
<td>0.31 (1.50)</td>
<td>0.78</td>
<td>0.45</td>
</tr>
</tbody>
</table>

* significant at the $\alpha = 0.01$ level
** significant at the $\alpha = 0.05$ level

As shown in Table 2, there were statistically significant increases in the Factor 1 subscale values when IDEA users moved from the “Smile” to the “Anger” and “Hand-Caught-in-Gear” referents. The Factor 2 subscale scores were significantly greater for the “Smile” and “Hand-Caught-in-Gear” referents than they were for the “Anger” referent. The Factor 3 and Factor 4 subscale scores for the “Smile” referent were significantly greater than those associated with “Anger” or “Hand-Caught-in-Gear”.

5.3 Changes in Facial Expression During IDEA Use

Results of the factor analysis and pair-wise T-test comparisons suggest that the IDEA users search for facial expression was driven by (1) their subjective assessment of the images presented to them, and (2) the nature of the referent (Smile, Anger, or Hand-Caught-in-Gear).
Results indicates as the referent changed from “Smile” to “Anger”, the Factor 1 subscale scores increased while the subscale scores for Factors 2, 3, and 4 decreased. These changes in factor subscale scores values would reflect the following changes in facial expression:

- Both the left and right eyebrows are angled towards the face’s centerline.
- Both the left and right corners of the mouth move downwards on the face while the middle portions of the mouth move upwards.

Such changes would be consistent with a frowning / glaring facial expressions.

The Factor 1 and Factor 2 subscale scores for the “Hand-Caught-in-Gear” referent were significantly greater than the corresponding scores associated with “Smile” and “Anger” respectively. In addition, the Factor 3 and Factor 4 subscale scores for the “Hand-Caught-in-Gear” referent were significantly lower that the Factor 4 scores generated under the “Smile” referent. Such differences suggest faces generated under the “Hand-Caught-in-Gear” referent result in:

- Lower resting eyebrows and a greater frowning expression in the mouth parameters compared to the faces generated under the “Smile” referent
- Both eyebrows more tilted away from the face’s centerline compared to those faces generated under the “Anger” referent

Such conclusions are consistent with the faces depicted in Figures 7-12.
Figure 7: The Smiley Faces Evolved by the 14 Networked Users

Figure 8: The Smiley Faces Evolved by the 8 Non-Networked Users
Figure 9: The Anger Faces Evolved by the 14 Networked Users

Figure 10: The Anger Faces Evolved by the 8 Non-Networked Users
Figure 11: The Hand-In-Gear Faces Evolved by the 14 Networked Users

Figure 12: The Hand-In-Gear Faces Evolved by the 8 Non-Networked Users
6.0 Current Applications of IDEA

Currently IDEA is being used for a number applications such as EC education, human factors, and as an aid for the development of high-performance ECs through the modeling of human problem solving.

6.1 IDEA Potential for Education and Human Factors

At Auburn University, IECs have been used since 1998 in an effort to teach introductory AI students to the basic concepts of ECs [Ritterbush 2000]. The IDEA presented in this paper is being used to teach students how selection, recombination, and mutation work together to form a powerful problem-solving paradigm. Students are also introduced to the concept of distributed EC. They discover first hand the possible speed-up that distributed/parallel ECs can provide. They also begin to understand how migration schemes and topologies can impact distributed/parallel EC search.

In a broader context, instructors and students from other disciplines and education levels could also benefit from further development of the IDEA approach. At the collegiate level, student teams from disciplines as diverse as architecture, human factors engineering, and industrial design could use such systems to develop creative and innovative solutions to various multi-disciplinary design problems. Such teams could use IDEA systems to participate in interactive, real-time, group-learning environments that are not constrained by the geographic locations of its human participants. Development of a language independent (i.e. no text required) interface for future IDEA systems could also prove beneficial to instructors working with elementary, middle, and high school students who have limited literacy skills or who suffer from disorders that impair their ability to communicate verbally (e.g. autism, mental retardation). By relying predominantly on visual (not verbal) communication, such IDEA systems could enable these individuals to take a more active role in student group participation activities and allow them to more effectively express their own creative design ideas to other people.

6.2 The Development of High-Performance ECs

By studying how the users solve problems using IDEA we believe that we can discover heuristics for adapting the usage rates of the evolutionary operators that are used in traditional ECs. This will result in the development of ‘intelligent’ ECs. We envision that these new forms of ECs will be able to backtrack in an effort to escape local optima as well as climb and follow ridges within a search space.

7.0 Conclusions and Future Research

The IDEA presented in this paper has been shown to be effective in allowing a number of users to interactively and collaboratively design emoticons. We believe that IDEA, as a concept, represents a new and exciting form of distributed problem solving. We believe this for a number of reasons: (1) not all designers are the same, (2) not all designers work at the same pace or time,
(3) not all designers have the same skill sets and gifts, (4) IDEA can exploit design team diversity, and (5) the IDEA meme space allows the design team to backtrack and consider older designs as well as allowing more universal ideas and concepts to survive longer (an permeate through all of the subpopulations).

Concerning our IDEA, there are a number of questions that need to be answered. They are as follows.

1. Given a design team of N users what is the most effective meme space size?
2. How do we make sure that faster users don’t overrun the meme space?
3. What is the best composition for a design team?
4. How many designs should one receive from meme space on a given generation?
5. How does this number change over the evolutionary process? How can we detect and reduce user fatigue and frustration?

The questions will be used to drive our future research.

REFERENCES


No. 1, pp. 23-33 April.


