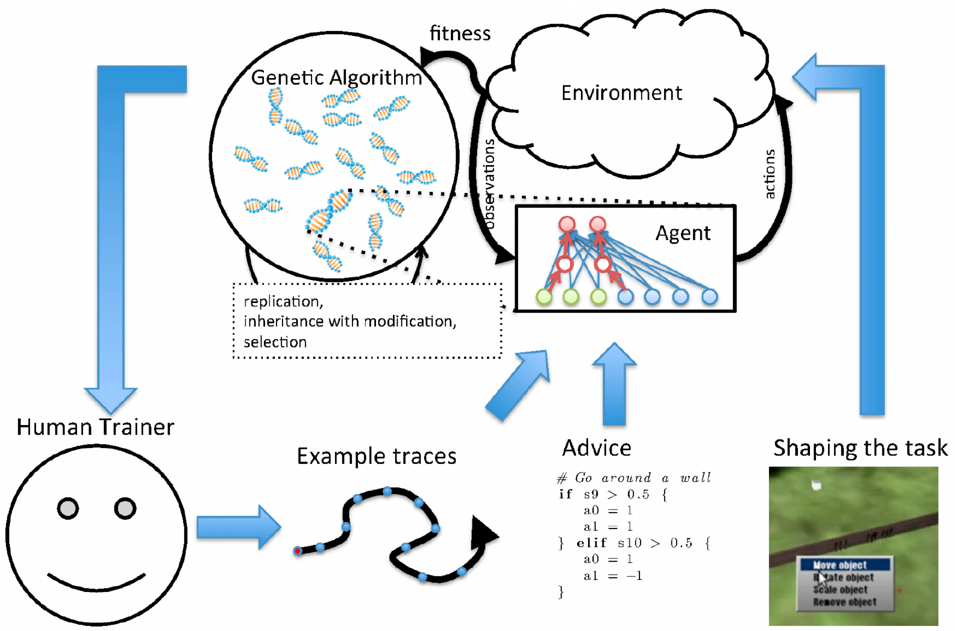


Figure 8.6: **A proposal for active human-guided neuroevolution.** The human expert provides advice, examples, and shaping for the neuroevolution process. The process monitors itself and determines what kind and when such input would be most useful. In this manner, humans and machines can work synergetically to construct intelligent agents. (Figure from Karpov et al. 2012)

for the agent and ask the user to provide an example. If evolution seems to have stagnated, it could propose the user to shape either the rewards or the environment to get evolution going again. It could even make specific suggestions, such as adjusting the sliders to make the task more demanding, or rolling back prior simplifications. Such an ability would eventually result in interactive neuroevolution where human knowledge and machine exploration work synergetically in both directions to solve problems (Figure 8.6).

8.3 Collaborative Neuroevolution

While NERO enabled players to shape the evolution of their team of agents, the game did not allow many humans to collaboratively train their teams by building on the interesting behaviors found by others. This chapter showcases some examples of interactive neuroevolution applications and games that were developed to incorporate such collaboration.

In particular, we'll take a closer look at Picbreeder (Secretan et al. 2011a), a highly influential generative AI system that came out of the lab of Kenneth Stanley. Picbreeder is a great example of a system that allows users to perform *collaborative* interactive neuroevolution, enabling them to explore a large design space together. Similarly to Dawkin's BioMorphs from his book The Blind Watchmaker, the basic idea in Picbreeder is to breed images. Users are presented with several images and asked to select the ones they like the most (Figure 8.7). The selected images are then used as parents to produce a new generation

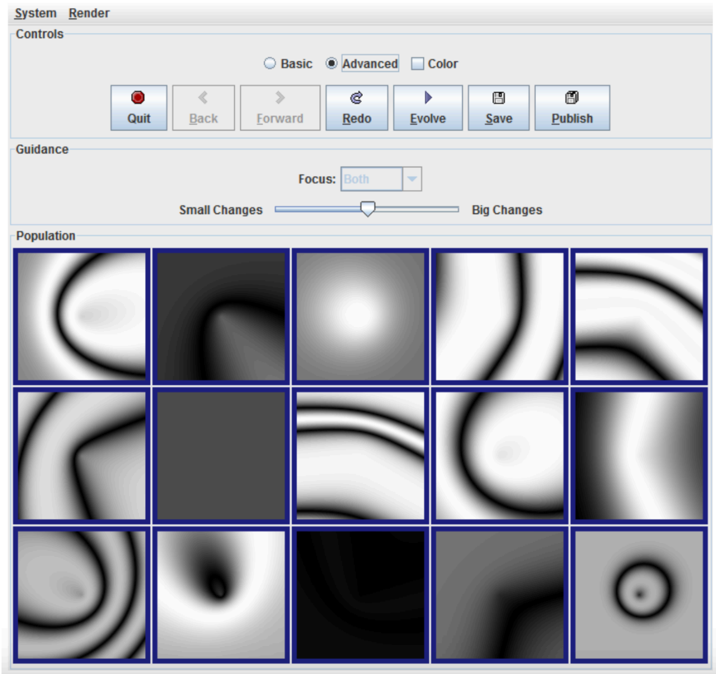


Figure 8.7: **Picbreeder interface.** Users in Picbreeeder select at least one CPPN-generated image, from which subsequent populations are generated through mutations and crossover of the underlying CPPNs. Users can also move back and forth through the generations and publish their creations, allowing others to branch off from their discoveries. (Figure from Secretan et al. 2011a)

of images through processes such as crossover (combining features from selected parents) and mutation (introducing slight variations to the underlying image representation). The new generation of images becomes the next population, and the process iterates. With each generation, users continue to select the images they prefer, and the algorithm evolves the images based on their choices.

Images in Picbreeder are represented by CPPNs (Chapter 4.3.1) and modified by the NEAT algorithm (Chapter 3.4). While the CPPN representation allows users to easily evolve images with interesting regularities, employing NEAT for the mutation and crossover of CPPNs has an added benefit: the evolved images gradually get more complex over generations because the underlying CPPNs are becoming more complex. To allow users to navigate the space of images in a meaningful way, NEAT mutation parameters for Picbreeder have to be chosen in a way where the next generation of images resembles their parents but also shows interesting variations.

With such an interactive evolution interface, one user by herself can already explore parts of the design space of images, but there is only so many generations a single person can evolve images for. Single-user interactive evolution applications often suffer from what is called user fatigue: The user might not see anything very interesting within 10 to 20 generations and thus lose interest to explore further (Takagi 2001). Picbreeder addresses these

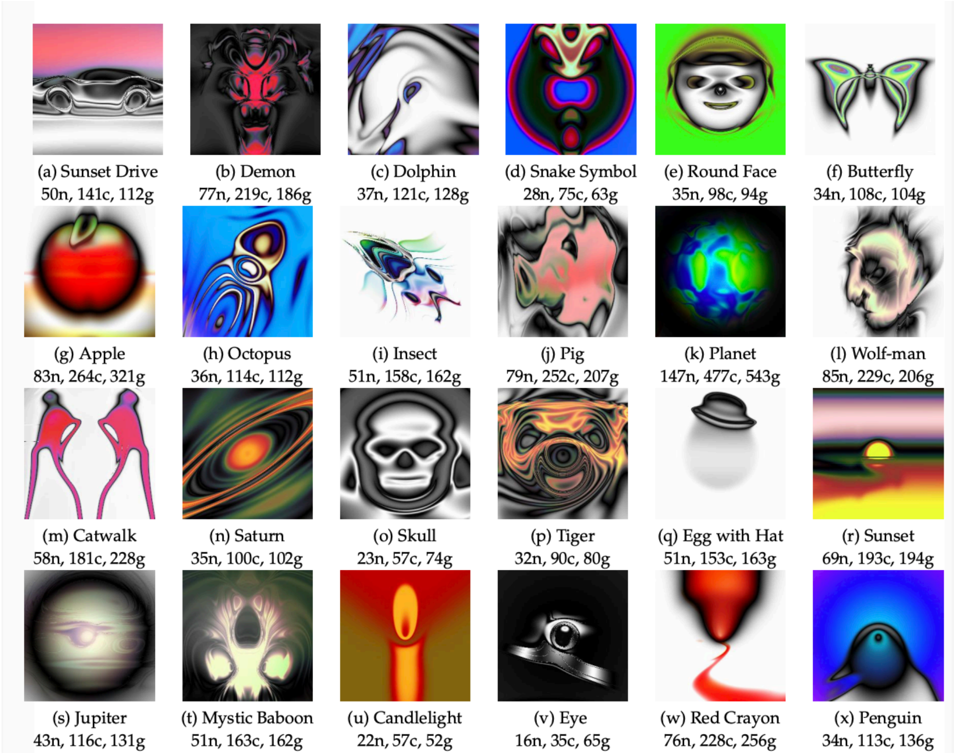


Figure 8.8: **Example of Picbreeder Images.** Shown are a variety of designs that were evolved by many collaborating users. For each design, the number of nodes n, connections c of the underlying CPPN are also shown together with the total number of cumulative generations g. (Figure from Secretan et al. 2011a)

issues in a clever way, by allowing users to evolve collaboratively thereby taking advantage of the fact that different users naturally want to evolve different artifacts. For example, some users might start out with the idea to evolve a particular image such as an insect, while others keep selecting the images that appear most compelling to them without a preset target in mind. In Picbreeder, a user can see what others have evolved and decide to continue evolution from any of their published images, a mechanism called branching. Through this process, users have been able to explore large parts of the design space. Figure 8.8 shows some selected images that many users were able to evolve together. Initially all starting out from abstract shapes similar to the ones shown in Figure 8.7, users were able to collaboratively evolve a great variety of different images, resembling subject matters such as faces, animals, landscapes, and many others.

Picbreeder has spawned a large number of projects that extend on its original idea, such as EndlessForms (Clune and Lipson 2011), which allows users to breed 3D artifacts instead of 2D images using a three-dimensional CPPN representation. Other examples include Artbreeder, which combines a Picbreeder-inspired interface with generative AI models such as GANs to allow users to directly start the evolutionary search in an interesting part of the design space. We take a closer look at these hybrid systems in Chapter 13 on generative

Figure 8.9: **The Petalz video game.** Players in Petalz can decorate their balconies with various different pots and balcony designs. They can breed new flowers by clicking on the existing flowers and trade flower seeds with other users. By allowing players to branch off the flowers discovered by others, Petalz allows a new type of digital social interaction that links players through collaborative interactive NE. (Figure from Risi et al. 2016)

AI. Interactive neuroevolution does also not need to be limited to generated visual artifacts, as demonstrated by systems such as NEAT drummer or MaestroGenesis (Hoover, Szerlip, and Stanley 2014), which allows users to interactively breed musical accompaniment to existing songs.

However, a common challenge with many of these systems is that, even though the process of interactive evolution by itself can be entertaining for a while, users often do not spend that much time on the site. Wrapping the whole collaborative evolution loop inside a game can address this issue, as we will see next.

8.3.1 Evolving Game Content

Just as interactive NE paved the way for innovative games like NERO, the concept of collaborative NE also facilitated the emergence of other types of video games, such as Petalz (Risi et al. 2016) and Galactic Arms Race (Hastings, Guha, and Stanley 2009). In both of these games, collaborative interactive Neuroevolution serves as a method for what is called Procedural Content Generation (PCG). In PCG, the goal is to generate game content, such as levels, characters, items, and more, algorithmically rather than manually designing them. In Petalz, which was a casual Facebook game, the main idea was to allow players to collaboratively breed different types of procedurally generated flowers. More specifically, players in Petalz possess a balcony they can decorate with various available flower pots (Figure 8.9). Additionally, players can visit the balconies of friends and water or like their flowers. Players can evolve their flowers by clicking on existing flowers which opens a menu that allows

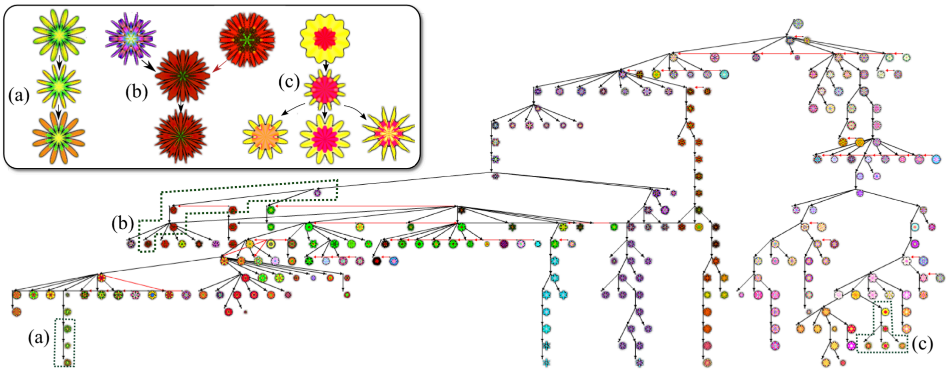
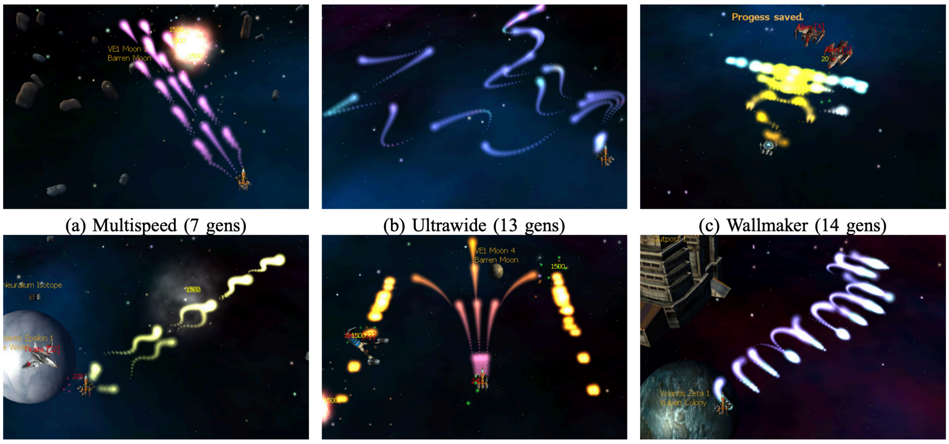


Figure 8.10: **A Petalz flower phylogeny.** Shown is a family tree that tracks the collaborative efforts of 13 distinct users. Each pair of parent and offspring is divided by one generation. For cases where a flower emerges from cross-pollination, the connecting line to the second parent is highlighted in red. The inset offers a closer look at the evolutionary dynamics, featuring minor phenotypic changes (a), an instance of cross-pollination (b), and substantial yet shared phenotypic transformations (c). (Figure from Risi et al. 2016)

generating flower offspring through mutations or to cross-pollinate a flower with another one, thereby performing a crossover. Flowers are generated by a CPPN representation that is modified to generate flower images and shapes (instead of arbitrary images), which are themselves also allowed to become more complex via the NEAT algorithm. Players can also sell their flower seeds in a digital market place or gift them to other players, thereby allowing others to continue breeding new flowers and creating a whole new lineage. An analysis of the flower market also showed that were cheaper or more aesthetically pleasing sold better, which showcased the economic value of such evolved artifacts. This marketplace also facilitated users to discover a wide variety of aesthetically pleasing flowers, such as the ones shown in the flower phylogeny in Figure 8.10. All in all, Petalz demonstrated that collaborative NE allows a new type of digital social interaction that links players in the interaction of exchanging flower seeds and collaborative flower breeding.

In the multiplayer game Galactic Arms Race (GAR), another game build on CPPNs and NEAT, players pilot a spaceship and fight enemies to acquire unique and procedurally generated particle weapons. GAR is another machine learning game, in which the integration of user preferences is slightly less direct than in a game such as Petalz, in which the users directly choose which flowers to reproduce. To smoothly integrate user preferences into a realtime game such as GAR, here the neuroevolutionary algorithm takes into account implicit information within the game's usage statistics. In particular, in GAR the game keeps track of how often players fired the different weapons that have in their three available weapon slots. New weapons being spawned into the game world are chosen to be mutations of the weapons that players preferred in the past. This way, players can collaboratively discover a wide variety of different particle weapons. Instead of describing a static 2D or 3D image, CPPNs in GAR are an interesting example of a CPPN generating a dynamical system. For each frame and for every particle of a particular weapon, the CPPN receives the particle's current position as input, in addition to the position it was initially



(a) Multispeed (7 gens) (b) Ultrawide (13 gens) (c) Wallmaker (14 gens)

(d) Corkscrew (3 gens) (e) Tunnelmaker (2 gens) (f) Blue Ladder (42 gens)

Figure 8.11: **Evolved particle weapons in Galactic Arms Race.** The interactive evolution component of GAR allowed players to evolve a large diversity of different and aesthetically pleasing weapons. More importantly, different evolved weapons have different tactical implications such as the Wallmaker (c), which favors defense-play by creating a particle wall in front of the player, or the (e) Tunnelmaker, which protects the player from attacks from the left or right side. (Figure from Hastings, Guha, and Stanley 2009)

fired from. The CPPN then outputs the particle's velocity in addition to its RGB-encoded color. While all particular weapons have the same number of particles, the ability of player projectiles to intersect enemy projectiles can lead to several tactical trade-offs explored by evolution. Slower projectiles offer the benefit of easier blocking against incoming fire, providing a defensive advantage. On the other hand, faster projectiles are better suited for precise aiming at distant enemies, offering offensive prowess. Two particularly fascinating types of evolved weapons are shown in Figure 8.11. Wallmakers are capable of forming a literal wall of particles in front of the player, and tunnelmakers generate a protective line of particles on both sides of the player.

Together, the examples in this and the previous chapter show that interactive neuroevolution can enable the creation of novel types of machine learning games with engaging player dynamics. Petalz had over 1,900 registered online users and 38,646 unique evolved flowers, which showcases the potential for PCG to enable these kinds of casual game mechanics. In the first two months of going online in 2009, GAR had over 1,000 registered online players that evolved 379,081 weapons. In addition to demonstrating the increasing entertainment value with a constant stream of evolved content, these examples also demonstrate the versatility of CPPNs to ecnode a variety of different types of content, from flower images to particle weapons, which all benefit from NEAT's ability to complexify the underlying representations and thus the resulting phenotypic patterns.

Beyond their application to games, interactive evolution systems can also serve other important functions. They enable researchers to visually explore the representation power of different types of encodings or the way that users individually or collaboratively explore such a space, leading to surprising insights. For example, as mentioned already in Chapter 5.3, while Picbreeder was initially invented to explore the CPPN encoding [is that correct?], playing with the system and realising that users in Picbreeder explore a vast search space very differently to current optimisation algorithms, led Kenneth Stanley and Joel Lehman to invent the novelty search algorithm. Interestingly, the different ways a search space is explored can also led to very different types of representations. In CPPN-representations evolved by users in Picbreeder, developmental canalization often emerges, where certain dimensions of variation are more likely while others are prevented (Huizinga, Stanley, and Clune 2018). For example, in Picbreeder, some of these canalized dimensions of variation are a "gene" for the size of objects, a "gene" determining how much the mouth of a skull (show in Figure 8.8,o) is open/closed, or a "gene" that controls the shadow of objects in an image. This type of developmental canalization is often linked to the evolution of evolvability in natural systems, which many believe to be essential for the tremendous diversity of functional organisms we see in nature. Representations evolved with traditional objective-based evolution, do not show this type of canalization and mutations to single genes here often effect none or many parts of the image. Artificial evolutionary systems can thus help us to determine under what circumstances different properties evolve, and we will return to this important topic in Chapter 14.

8.4 Making Human Contributions Practical

Interactive evolution experiments require significant human effort, which makes it difficult to take advantage of it more broadly. Some domains, like Picbreeder, are inherently interesting and rewarding, and a large number of people can contribute to it through publicly available websites. But other domains may be more abstract and progress in them less obvious, resulting in users fatiguing and losing interest. [Note Sebastian: Another way to make them more practical is to combine them with pre-trained networks. Maybe we should here link to the section on Hybrid Systems as well]

One solution is to use human computation markets (HCM), such as Amazon Mechanical Turk, to recruit humans to this role. In a sense, monetary reward can thus be used as a substitute for the intrinsic enjoyment of creativity and curiosity. Of course, using HCM requires funds, but so does other types of computation as well. In a sense, some of the computational budget is used for human computation instead of cloud computation.

HCMs can be used effectively in three roles (Lehman and Miikkulainen 2013): To bootstrap experiments to become interesting, to evaluate different designs, and to extend interactive evolution to long experiments.

First, even if a task such as a Picbreeder is eventually engaging and rewarding, it is not so at the very beginning. The forms are simple and stay simple for several generations. It is difficult to get people to evaluate such images, and evaluation itself is not very meaningful. It turns out that if this phase is automated, or HCM used to get through it, the final images turn out more interesting. For instance in the Picbreeder domain, it is possible to generate an initial set of images algorithmically, and thus make them more complex and interesting than simple geometric forms (Lehman and Kenneth O. Stanley 2012). A simple fitness such as one based on rarity (or novelty) and complexity (or effort) can be used to guide this initial evolution. At the next phase, it is then possible to use HCM to improve upon those

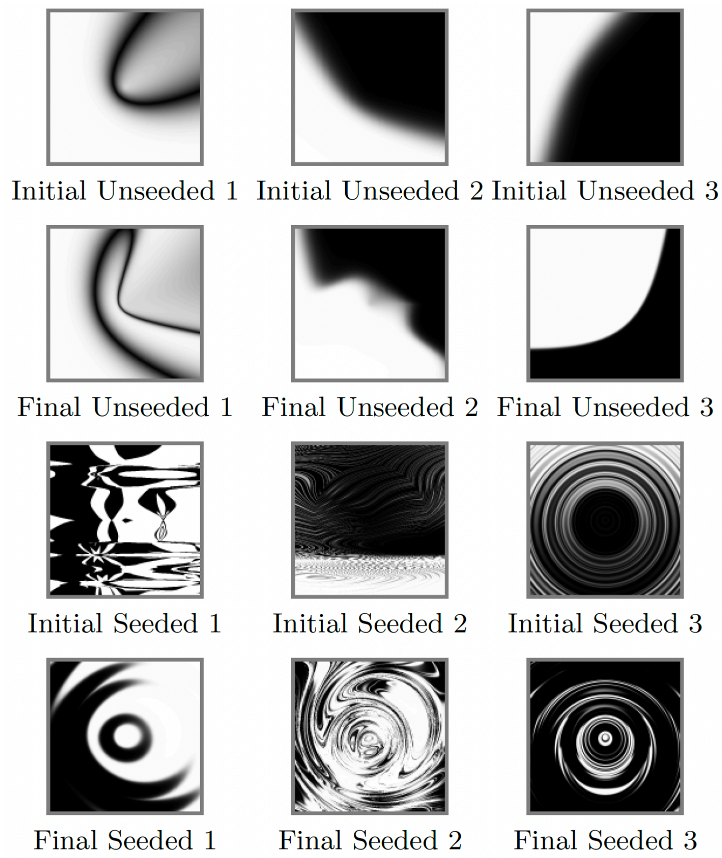


Figure 8.12: **Example initial and final images with and without seeding interactive neuroevolution.** The early phase of Picbreeder is not very engaging, but can be bypassed by seeding. In this comparison, the initial unseeded images were generated with random CPPNs; the initial seeded ones were generated by running CPPN evolution for a while and selecting the most impressive images. Both sets of images were then evolved further with Picbreeder using HCM. Interactive evolution from seeded images results in more complex and appealing final images, suggesting that proper initialization is crucial in taking full advantage of interactive evolution. (Figure from Lehman and Miikkulainen 2013)

images further, up to a level where the images are actually appealing to humans, and the creativity/curiosity rewards can take over.

Figure 8.12 compares three interactive evolution runs of Picbreeder in these two conditions: starting from random images, and starting from algorithmically seeded images, in both cases followed by a period of further evolution with HCM. The seeded runs resulted in more complex images, and human judges also found them more aesthetically appealing. Thus, initial machine exploration and HCM can be used to make interactive evolution experiments more effective.

Second, there are also tasks where the creativity/curiosity reward never becomes large enough to justify the human effort, and therefore HCM is necessary to perform the experiments in the first place. A particularly important general case is experimental design of such experiments. For example, the images can be encoded in various ways: for example through using CPPNs or simple ANNs with different activation functions. It may not be possible to make these design choices correctly without running preliminary experiments, and such experiments are often not very interesting to human users. HCM can be used to good effect in discover best designs before running the actual experiments.

Third, in some cases evolution needs to be run very long in order to get good results. Even if the task is interesting, the users will eventually fatigue. HCM can provide a continual, indefinite stream of new users in such experiments. On the other hand, each user makes only a transient contribution to the evolutionary process, and these contributions may be inconsistent. It turns out, however, that long-running evolution can still utilize them as a guide towards good solutions. Evaluations in most domains are always noise, and such inconsistency is simply another form of such noise. As usual, evolution is robust against noisy evaluations, and they may even boost creativity by encouraging exploration. Thus, HCM can be harnessed to enable long-running interactive evolution experiments.

Thus, while interactive evolution experiments require significant human effort, there are ways to make them practical, and thus realize full potential of human guidance.

8.5 Chapter Review Questions

1. **Conceptual Understanding:** How does interactive neuroevolution differ from standard neuroevolution, and what types of problems is it particularly well-suited to solve?
2. **Human-Guided Evolution:** In the context of the NERO game, what tools are provided to the human player to guide the neuroevolution process? How can these tools shape the evolution of agent behaviors?
3. **Real-Time Evolution:** What is the role of rtNEAT (real-time NEAT) in NERO, and how does it enhance the interactive experience compared to traditional generational neuroevolution?
4. **Behavioral Shaping:** Describe how curricular evolution is implemented in NERO to train agents progressively. Why is this approach often more effective than using a single, static objective function?
5. **Surprising Behaviors:** Give examples of unexpected strategies discovered by evolution in NERO. How do such discoveries highlight the balance between human guidance and evolutionary creativity?
6. **Interactive Machine Learning Games:** Based on the NERO example, what characteristics make machine learning games engaging for human players, and how does the circularity of strategies contribute to the gameplay?
7. **Collaborative Exploration:** How does Picbreeder address the challenge of user fatigue in interactive neuroevolution, and what role does branching play in enabling collaborative exploration?

8. **Generative Applications:** Describe how Petalz and Galactic Arms Race utilize collaborative neuroevolution to procedurally generate game content. How do their approaches differ in incorporating user preferences?
9. **Representation and Evolvability:** What is developmental canalization, and how does it emerge in CPPN representations evolved in Picbreeder? Why is this property significant for understanding evolvability?
10. **Practical Implementation:** What strategies can make interactive neuroevolution more practical in domains with limited user engagement or long-running experiments? Provide examples of how Human Computation Markets (HCM) can be effectively utilized.