1

System Design Using Cultural Algorithms

Robert G. Reynolds

Computer Science, Wayne State University, Detroit, MI, USA
The Museum of Anthropological Archaeology, University of Michigan-Ann Arbor, Ann Arbor, MI, USA

Introduction

By and large, most approaches to machine learning focus on the solution of a specific problem in the context of an existing system. **Cultural Algorithms** are a knowledge-intensive framework that is based on how human cultural systems adjust their structures and contents to address changes in their environments [1]. These changes can produce a solution to the new problem within the existing social framework. Beyond that, the system can adapt its framework in order to produce the solution for a larger class of related problems. Cultural Algorithms are able to mimic this behavior by the self-adaptation of its' knowledge and population components.

In other words, we are participating in the Cultural learning process right now. However, as part of the process it is hard to assess what progress, if any, is being made by the system. The Cultural Algorithm provides a framework by which we can step outside of the system so that we can assess its trajectories more clearly. This issue is addressed somewhat by the notion of "human-centric" learning. However, such an approach suggests that we are ultimately in control of the learning activities. In reality, we are embedded in a performance environment that we have partially created on the one hand, and have been passed down as the result of millions of years of evolution on the other.

The framework for the Cultural Algorithm is given in Figure 1.1. A networked population of agents interact with each other in the population space. The network of agents is termed the **social fabric**. Agents are connected with each other in the network based on their level of interaction. If the level of interaction between a pair of agents falls below a certain level, that connection can be lost. In that sense, the network is like a piece of cloth where a stress on some portion of the fabric can lead

This work was supported by grant NSF #1744367.

Cultural Algorithms: Tools to Model Complex Dynamic Social Systems, First Edition.
Edited by Robert G. Reynolds.

Companion website: www.wiley.com/go/CAT

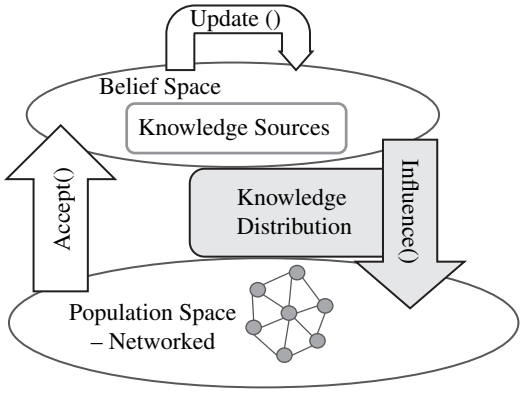

Figure 1.1 Cultural Algorithm framework.

to a disruption or tear in the fabric. Such tears can be mended over time if interactions resume. It is a key feature of Cultural Algorithms since they need to be able to simulate not only the growth but also the decline of social systems [2].

The results of agent interaction within the performance environment in which they are embedded can be accepted into the Belief Space. The Belief Space is a repository of the knowledge acquired by the system so far. It is viewed as a network of different knowledge sources. The accepted knowledge is then integrated into the network through the use of learning procedures that make focused adjustments to the cultural compendium of knowledge. The Information "cloud" can be viewed as the current manifestation of the Belief Space using current technology.

These knowledge sources in the Belief Space can be "active" and or "passive." Active knowledge sources directly select individuals based on their location and history in the social fabric (network). Passive knowledge sources are selected by individual agents in the network. A knowledge source can be both active and passive. The influence function in a Cultural Algorithm has two stages. In the first stage, each individual is assigned a **direct influence**, either actively or passively. Next, comes the knowledge distribution stage. Each individual's direct knowledge source is compared with a subset of its neighbors in the network in the **knowledge distribution stage**. If the knowledge sources are the same, then nothing more needs to be done for an individual. On the other hand, if there is a disagreement, then there is a conflict that needs to be resolved. This conflict is mitigated by a **knowledge distribution mechanism**. Currently, the mechanisms used are taken from traditional approaches to conflict resolution including drawing straws, majority win, weighted majority, win, various auction mechanisms, and various game frameworks including the Prisoners Dilemma and Stackleberg games.

The resultant distribution ranges from static, to moderate, to viral in nature. Individual agents then use their knowledge source(s) to direct their actions in the performance environment. The results of the actions are then sent to the Accept function to decide what will be used to update the Belief Space, and then the cycle continues.

The knowledge sources themselves can support exploitative, exploratory, or stem behaviors. **Exploratory** mechanisms produce new knowledge about the search space, while **exploitative** mechanisms focus the search within already discovered regions. A knowledge source that exhibits a "**stem**" behavior is one that can either produce exploitative or exploratory behavior dependent on the context. The term itself derives from the biologic notion of "stem cell." It is a useful transitional device since in the solution of a complex multiphase optimization problem knowledge sources that are useful in one phase may become less useful at the onset of another. The stem knowledge source can help expedite the transition from one set of knowledge sources, say exploitative, that are dominant at the end of one phase to a set that are more useful in the start of the next phase, such as exploratory ones.

This ability to transition from the use of one set of knowledge sources to another as problem dynamics change is one of the key features of cultures in general. The goal of a Cultural System like that of an operating system for a computer is to continue to provide resources for its active agents. The features inherent in the Cultural Algorithm that support this notion of process **sustainability** are as follows:

1) Cultural Algorithms inherently support **multiobjective** approaches to problem solving. A multiobjective problem is when there is some conflict in an agent's goals, such that the achievement of one goal takes resources away from achieving the other. Since conflicting objectives can reside simultaneously in the Belief Space, agents working on one goal may need to resolve conflicts with agents working on complementary ones. So Cultural Algorithms do not need to be restructured to explicitly deal with multiobjective problems, whereas other machine learning algorithms may need to do so.
2) Cultural Algorithms inherently support population **co-evolution**. Stress within the social fabric can naturally produce co-evolving populations. New links can be created subsequently to allow the separate populations to interact again.
3) Cultural Algorithms also support alternative ways to use resources through the emergence of subcultures. A **subculture** is defined as a culture contained within a broader mainstream culture, with its own set of goals, values, practices, and beliefs. Just as co-evolution concerns the disconnection of individuals in the agent network, subcultures represent a corresponding separation of knowledge sources in the Belief Space into subcomponents that are linked to groups of connected individuals within the Population Space.

4) Cultural Algorithms support the social context of an individual by providing mechanisms for that individual to resolve **conflicts** with other individuals in the population space through the use of knowledge distribution mechanisms. These mechanisms are designed to reduce conflicts between individuals through the sharing of knowledge sources that influence them. This practice can be used to modulate the flow of knowledge through the population. The use of certain distribution strategies can produce viral distributions of information on the one hand or slow down the flows of the other knowledge sources dependent on the context. This feature makes it a useful learning mechanism with regards to design of systems that involve teams of agents.
5) Cultural Algorithms support the idea of a **networked performance space**. That is, the performance environment can be viewed as a connected collection of performance functions or performance simulators. This allows agent performance to potentially modify performance assessment and expectations.
6) Cultural Algorithms can exhibit the flexibility needed to cope with the changing environments in which they are embedded. They were in fact developed to learn about how social systems evolved in complex environments [3].
7) Cultural Algorithms facilitate the development of distributed systems and their supporting algorithms. The knowledge-intensive nature of cultural systems requires the support of both distributed and parallel algorithms in the coordination of agents and their use of knowledge.

All of these features have been observed to emerge in one or more of the various Cultural Algorithm systems that have been developed over the years. In subsequent chapters of this book, we will provide examples of these features as they have emerged and their context.

The Cultural Engine

While there is wide variety of ways in which Cultural Algorithms can be implemented, there is a general metaphor that describes the learning process in all of them. The metaphor is termed the "Cultural Engine." The basic idea is that the new ideas generated in the Belief Space by the incorporation of new experiences into the existing knowledge sources produce the capacity for changes in behavior. This capacity can be viewed as entropy in a thermodynamic sense. The influence function in conjunction with the knowledge distribution function can then distribute this potential for variation through the network of agents in the Population Space. Their behaviors taken together provide a potential for new ideas that is then communicated to the Belief Space and the cycle continues.

We can express the Cultural Algorithm Engine in terms of the entropy-based laws of thermodynamics. The basic laws of classical thermodynamic concerning systems in equilibrium are given below [4]:

Zeroth Law of Thermodynamics, About Thermal Equilibrium

If two thermodynamic systems are separately in thermal equilibrium with a third, they are also in thermal equilibrium with each other. If we assume that all systems are (trivially) in thermal equilibrium with themselves, the Zeroth law implies that thermal equilibrium is an equivalence relation on the set of thermodynamic systems. This law is tacitly assumed in every measurement of temperature.

First Law of Thermodynamics, About the Conservation of Energy

The change in the internal energy of a closed thermodynamic system is equal to the sum of the amount of heat energy supplied to or removed from the system and the work done on or by the system.

Second Law of Thermodynamics, About Entropy

The total entropy of any isolated thermodynamic system always increases over time, approaching a maximum value. Therefore, the total entropy of any isolated thermodynamic system never decreases.

Third Law of Thermodynamics, About the Absolute Zero of Temperature

As a system asymptotically approaches absolute zero of temperature, all processes virtually cease, and the entropy of the system asymptotically approaches a minimum value.

We metaphorically view our Cultural Algorithm as composed of two systems, a Population of individual agents moving over a performance landscape as well as a collection of knowledge sources in the Belief Space. Each of the knowledge sources can be viewed statistically as a bounding box or generator of control. If we look closely at the second law, it states that over time an individual system will always tend to increase its entropy. Thus, over time the population should randomly spread out over the surface and the bounding box for each knowledge source expand to the edge of the surface, encompassing the entire surface. Yet this does not happen here. This can be seen in terms of a contradiction posed by Maxwell relative to the second law. This Contradiction is the basis for **Maxwell's Demon**.

In the 1860s, the Physician Maxwell devised a thought experiment to refute the second law [5]. The basis for this refutation was that the human mind was different than pure physical systems and that the universe need not run down as predicted by the second law. In his thought experiment, there were two glass boxes. Within each box was a collection of particles moving at different rates. There was a trap door connecting the boxes controlled by a demon, see Figure 1.2. This demon was able to selectively open the door to "fast" particles from A and allow them to go to B. Therefore, increasing the entropy of B and reducing that of A.

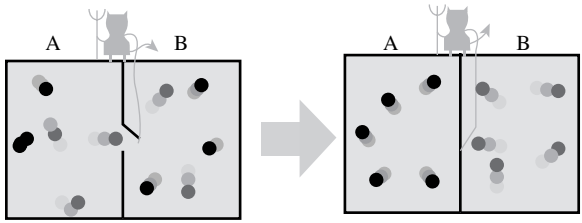

Figure 1.2 An example of Maxwell's demon in action. The demon selectively lets particles of high entropy from one system to another. Reducing the entropy in one and increasing it in the other.

This stimulated much debate among physicists. Leo Szilard in 1929 published a refutation of this by saying that the "Demon" had to process information to make this decision and that processing activity consumed additional energy. He also postulated that the energy requirements for processing the information always exceeded the energy stored through the Demon's sorting.

As a result, when Shannon developed his model of information theory he required all information to be transported along a physical channel. This channel represented the "cost" of transmission specified by Szilard. He was then able to equate the entropy of physical energy with a certain amount of information, called negentropy [6] since it reduced entropy as the Demon does in Figure 1.2.

We can use the metaphor of Maxwell's Demon as a way to interpret the basic problem-solving process carried out by Cultural Algorithms when successful. We will call this process the **Cultural Engine**. Recall that the communication protocol for Cultural Algorithms consists of three phases: vote, inherit, and promote (VIP). The voting process is carried out by the acceptance function. The inherit process is carried out by the update function. The promote function is carried out through the influence function. These functions provide the interface between the Population component and the Belief component. Together, similar to Maxwell's Demon, they extract high entropy individuals first from the population space, update the Belief Space, and then extract high entropy Knowledge Sources from the Belief Space back to modify the Population Space like a two-stroke thermodynamic engine.

Thus, the evolutionary learning process is viewed to be directed by an engine powered by the knowledge that is learned through the problem-solving process. While the engine is expressed here in terms of Cultural Algorithm framework, it is postulated that any evolutionary model can be viewed to be powered by a similar type of engine.

Outline of the Book: Cultural Learning in Dynamic Environments

Earlier it was mentioned that one thing that differentiates Cultural Algorithms from other frameworks is that it is naturally able to cope with changes in its environment. In Engineering, dynamic environments are typically modeled in three

basic ways. One approach is to take a general problem such as bin packing and make changes to that application problem over time. A second approach is to generate changes in a multidimensional fitness landscape over which the search problem is defined [7]. A third way is to use large-scale problems whose solution takes place in multiple phases such as the design of a cloud-based workflow system. In the design of a complex system such as this, the knowledge used in dealing with one phase may be different from the knowledge needed in subsequent phases.

The focus of this book is on the design of Cultural Algorithm solutions for the development of complex social and engineering systems for use in dynamic environments. Chapter 2 introduces the Cultural Algorithm toolkit (CAT). That system contains a Cultural Algorithm that is connected to a dynamic problem landscape generator. The generator, the ConesWorld, is an extension of the work of Dejong and Morrison [7]. It was selected because its dynamics were described in terms of entropy, which makes it a good fit with the Cultural Engine model discussed above. It is written in Java and available on the website associated with the book. Examples of its application to problems in the simulated landscape along with some benchmark engineering design problems are presented.

A second feature of Cultural Algorithms mentioned earlier is their ability to provide a social context for an individual and facilitate the movement of knowledge through a network. Chapters 3 and 4 investigate the use of several knowledge distribution mechanisms using that platform. Chapter 3 by Kinniard-Heether et al. shows when an auction mechanism can be a useful tool in expediting the solution to optimization problems generated in the ConesWorld at different entropy levels. Al-Tirawi et al. in Chapter 4 investigates the extent to which allowing the knowledge sources' specific information about an individual's location in a social network can improve performance in dynamic problems with high entropy. The approach is called common valued auctions. The common value related to the shared knowledge that knowledge sources have about the location of individuals in a network. The Common value approach is then applied to ConesWorld landscape sequences that range from low entropy to highly chaotic systems.

Auctions can be viewed as competitive games, but the strategies available to bidders are by definition limited. In Chapter 5, Faisal Waris et al. investigate the use of competitive and cooperative games in CA problem solving. First, examples are presented within the ConesWorld environment and compared with other knowledge distribution mechanisms. The latter half of the paper investigates the use of a CA in the design of a real-world application for autonomous vehicles. The real-world system to be designed is an Artificial Intelligence pipeline for a pattern recognition component. Such pipelines consist of a series of components, and each of the components is tuned initially by their manufacturer. However, when placed within a pipeline, the parameters for all of the participating stages need to be tuned to optimize pipeline execution. Such pipelines can consist of 50 or more

stages. In the paper, it is shown that the CA that uses a competitive game framework provides a statistically more efficient solution than alternative approaches. In addition, maps of how knowledge sources that are distributed within a successful network are provided. Unsuccessful networks are more conservative with more homogeneous regions and possess overall less diversity in their distributions than those produced with games.

As stated earlier, another feature of Cultural Algorithms is their ability to produce effective designs of networks for team-based systems. The next two chapters focus specifically on team-based design. Chapter 6 by Kobti et al. describes the use of Cultural Algorithms in the design of a variety of real-world networks. They are interested in how CAs can be applied to the team formation problem, TFP. The TFP is in general NP-hard. Efficient team formation is key to the success of large-scale industry projects that employ a number of different individuals, each with their own expertise and skills. The first example used is a coauthorship network in which individuals collaborate to produce a specific product or outcome. A second example application is that of a palliative care network, where a team of healthcare providers are networked with each other and a number of patients. Cultural Algorithms are shown to be advantageous in each of these cases in comparison to traditional techniques in terms of producing efficient solutions to each of these different problems.

Chapter 7 by Ali et al. addresses the design of a competitive robot soccer team. It employs a population model based on Evolutionary Programming (EP) to evolve offensive and defensive strategies. First, Evolutionary Programming and Genetic Algorithms were each used to develop the offensive and defensive skills of a team. Next, EP was embedded into a CA as the population component. As it turned out, the CA enhanced the EP to beat both the unenhanced EP and GA teams as well as a hard-coded default team. In other words, it was able to produce an increase in team performance beyond that of a human expert and was able to beat the unenhanced versions as well.

The following two chapters focus on the use of Cultural Algorithms in the solution of multiobjective problems. Chapter 8 by Kattan et al. employs Cultural Algorithms to assess the impact that climate change has on artisanal offshore fishing in Peru. Artisanal fishermen of the Pacific coast of Peru use traditional equipment to catch fish, unlike the large-scale deep sea vessels. Marcus [8] collected the data for all fishing trips, over 6000, conducted from a specific coastal Peru port, Cerro Azul. During this period, the ecosystem was affected first by warming of the waters due to an El Nino, then by a subsequent cooling called La Nina, and finally a back to normal phase. A biobjective model of fishing behavior was produced that traded off quality catches versus investment in resources. On the one hand, a goal is to produce the highest payoff in terms of quality catches. On the other hand, since each fisherman is an independent producer, the goal is to minimize the resources needed to produce catches.

Pareto fronts were produced for each of the three phases and the results analyzed and compared statistically by a parallel Cultural Algorithm, CAPSO. CAPSO is short for **C**ultural **A**lgorithm **P**article **S**warm **O**ptimizer. The results indicate that the changes in the ecosystem produced by the warming and cooling of the regional waters resulted in statistically significant changes in the fishing behavior of the individual fishing agents. In the El Nino phase, individuals favored catch quality over resource investment. In the La Nina phase, they emphasized resource investment to insure producing any catch at all. The back to normal phase produced a more balanced set of fronts that represented the knowledge obtained over hundreds of year of fishing in the area, now that the requisite food chains were back to normal.

Chapter 9 by Stanley et al. describes the design of the parallel Cultural Algorithm, CAPSO. While CAPSO was designed initially to deal with the intensive parallelism inherent in the Peru Fishing computations, they were interested in how parallelism was actually needed to support the efficient solution of benchmark problems in multiobjective optimization. Often algorithms are tweaked to produce better results relative to existing benchmarks. That way, individuals can compare their approach with the solutions provided by other systems. This often results in algorithms that may be more tailored to the needs of the benchmark problems than those of the real world.

When CAPSO was applied to a representative set of benchmark problems, two basic patterns emerged. First, very little parallelism was needed to find an efficient solution of each of the problems. At most, around 30 parallel threads were needed as compared to the hundreds required for the fishing example. Second, the knowledge sources most frequently used to guide the search were exploitative in nature, rather than exploratory. The mathematical formulations of the examples were such that once explorations found a piece, the exploiter knowledge sources were able to fill in the rest. So the parallelism that was observed was primarily due to the exploration portion, which contributed to the overall computational time in a limited way.

Like Chapter 8, Chapter 10 also deals with cultural change. However, Chapter 8 dealt with relatively local and short-term change, but the scale of change is markedly different in Chapter 10. The paper by Palazzolo et al. examines the use of Cultural Algorithms on a much larger scale project, one that involves hundreds of square miles and thousands of years. The Land Bridge project is an NSF supported cooperative project between the University of Michigan-Ann Arbor and Wayne State University [9]. The project used AI technology to create a virtual world model of an ancient environment, now submerged under over a hundred feet of water in Lake Huron. The goal of the project is to predict the location of archaic hunting sites that existed over 10 000 years ago. This was done by producing a model of caribou behavior and using Cultural Algorithms to guide the production of optimal migration paths for large caribou herds across the ancient landscape;

the assumption was that hunting sites and activities were positioned relative to these migration pathways. Cultural Algorithms were then employed to produce a system that used knowledge from the anthropological literature to predict hunting site location. The results were used to guide researchers to previously undiscovered sites.

References

1 Reynolds, R.G. (1999). An overview of Cultural Algorithms. In: *Advances in Evolutionary Computation* (eds. D. Corne, M. Dorigo and F. Glover), 367–378. New York: McGraw-Hill.

2 Jayyousi, T.W. and Reynolds, R.G. (2014). Extracting urban occupational plans using cultural algorithms [application notes]. *IEEE Computational Intelligence Magazine* 9 (3): 66–87.

3 Reynolds, R.G. (1978). On modeling the evolution of hunter-gatherer decision-making systems. *Geographical Analysis* 10 (1): 31–46.

4 en.wikipedia.org Laws of Thermodynamics, 2020.

5 Reynolds, R.G. (2018). *Culture on the Edge of Chaos*. Springer.

6 Woodward, P. (ed.) (1957). Entropy and negentropy. *IRE Transactions on Information Theory.* 3 (1): 3–3.

7 Morrison R., De Jong K. (1999). A test problem generator for non-stationary environments. Proceedings of the Congress on Evolutionary Computing, pp. 25–31.

8 Marcus, J. (ed.) (2016). *Coastal Ecosystems and Economic Strategies at Cerro Azul, Peru. The study of a Late Intermediate Kingdom*. Ann Arbor, MI: Memoirs of the Museum of Anthropology, University of Michigan.

9 O'Shea, J.M. (2002). The archaeology of scattered wreck-sites: formation process and shallow water archaeology in western Lake Huron. *The International Journal of the Nautical Archaeology* 31 (2): 211–247.