In the preceding chapters we described the main varieties of evolutionary algorithms and described various examples of how they might be suitably implemented for different applications. In this chapter we turn our attention to systems in which, rather than existing as stand-alone algorithms, EA-based approaches are either incorporated within larger systems, or alternatively have other methods or data structures incorporated within them. This category of algorithms is very successful in practice and forms a rapidly growing research area with great potential. This area and the algorithms that form its subject of study are named memetic algorithms (MA). In this chapter we explain the rationale behind MAs, outline a number of possibilities for combining EAs with other techniques, and give some guidelines for designing successful hybrid algorithms.

10.1 Motivation for Hybridising EAs

There are a number of factors that motivate the hybridization of evolutionary algorithms with other techniques. In the following we discuss some of the most salient of these. Many complex problems can be decomposed into a number of parts, for some of which exact methods, or very good heuristics, may already be available. In these cases it makes sense to use a combination of the most appropriate methods for different subproblems.

Overall, successful and efficient general problem solvers do not exist. The rapidly growing body of empirical evidence and some theoretical results, like the No Free Lunch theorem (NFL),\(^1\) strongly support this view. From an EC perspective this implies that EAs do not exhibit the performance as suggested in the 1980s, cf. Fig. 3.8 in Sect. 3.5. An alternative view on this issue is

---

\(^1\) The NFL is treated in detail in Chap. 16, including a discussion about what it really says. For the present we interpret it as stating that all stochastic algorithms have the same performance when averaged over all discrete problems.
given in Fig. 10.1. The figure considers the possibility that we could combine problem-specific heuristics and an EA into a hybrid algorithm. Furthermore, it is assumed that the amount of problem-specific knowledge is variable and can be adjusted. Depending on the amount of problem-specific knowledge in the hybrid algorithm, the global performance curve will gradually change from roughly flat (pure EA) to a narrow peak (problem-specific method).

Fig. 10.1. 1990s view of EA performance after Michalewicz [295]

In practice we frequently apply an evolutionary algorithm to a problem where there is a considerable amount of hard-won user experience and knowledge available. In such cases performance benefits can often arise from utilising this information in the form of specialist operators and/or good solutions, provided that care is taken not to bias the search too much away from the generation of novel solutions. In these cases it is commonly experienced that the combination of an evolutionary and a heuristic method – a hybrid EA – performs better than either of its ‘parent’ algorithms alone. Note, that in this sense, Figure 10.1 is misleading as it does not indicate this effect.

There is a body of opinion that while EAs are very good at rapidly identifying good areas of the search space (exploration), they are less good at the ‘endgame’ of fine-tuning solutions (exploitation), partly as a result of the stochastic nature of the variation operators. To illustrate this point, as anyone who has implemented a GA to solve the One-Max problem\(^2\) knows, the algorithm is quick to reach near-optimal solutions, but the process of mutation finding the last few bits to change can be slow, since the choice of which genes are mutated is random. A more efficient method might be to incorporate a more systematic search of the vicinity of good solutions by adding a local search improvement step to the evolutionary cycle (in this case, a bit-flipping hill-climber).

A final concept, which is often used as a motivation by researchers in this field, is Dawkins’ idea of memes [100]. These can be viewed as units of cultural

\(^2\) A binary coded maximisation problem, where the fitness is simply the count of the number of genes set to 1.