Problem Decomposition and Multi-Objective Optimization

Richard A. Watson

Dynamical and Evolutionary Machine Organization,
Volen Center for Complex Systems, Brandeis University, Waltham, MA, USA
richardw@cs.brandeis.edu

1 Sub-Problems and Objectives

Divide and conquer techniques in problem solving are familiar and intuitive; first find the solution to sub-problems and then re-use these to find solutions to the whole problem. For example, we may decompose the problem of designing a vehicle into designing the engine and designing the body. It is acknowledged that most real-world problems (vehicles included) do not decompose neatly into separable sub-problems. For example, the optimal properties of a drive system have dependencies with the passenger capacity. Nonetheless, it is very often possible to simplify a problem greatly by identifying sub-problems that exhibit some degree of independence.

Multi-Objective Optimization, MOO, is similarly familiar and intuitive; there are several features of a system that we wish to optimize simultaneously and we wish to examine the alternatives that optimize each of the features independently, and/or offer a compromise of multiple objectives simultaneously. For example, we wish to minimize both the materials cost and construction time for our vehicle. It is acknowledged that sometimes multiple objectives can be satisfied simultaneously. For example, perhaps there is a simple design that is both cheap and fast to manufacture. This is the basis of Pareto dominance; a solution that is preferred with respect to all objectives. Nonetheless, it is often useful to acknowledge that objectives are constrained and to accept a set of solutions that optimize different objectives, rather than a single compromise.

Both these forms of optimization recognize some form of component structure in the problems they address. In problem decomposition we think of a problem with multiple sub-problems, in MOO we think of a problem with multiple objectives. We propose that the difference between the two approaches is primarily one of emphasis. The main difference between a sub-problem and an objective is that the former expects some degree of independence from other sub-problems, whereas the latter expects some degree of constraint with other objectives. Yet, problem decomposition accepts that compromise may be necessary, and MOO accepts the possibility of a solution that may be good in respect to all objectives. In reality, both approaches acknowledge that components of a problem exhibit both independence and constraint.

Ideally, in both these approaches we would like to take a solution that is good with respect to one objective or sub-problem, and put it together with a solution that is good with respect to another objective or sub-problem, and somehow combine them to find a solution that is good at both objectives or the whole problem. The similarity of these

approaches suggests that a unification of the principles and methods may be available, or at least, we may find some cross-fertilization from one field to the other.

2 A Compositional Problem with Constrained Building-Blocks

We have been studying a class of problems that exhibit a hierarchical building-block structure where the solutions to blocks are strongly constrained (1, 2). The problem is posed as a single objective function, but since this function is a summation of sub-functions corresponding to the component sub-blocks, the problem is amenable to some degree of problem decomposition. However, since the optimal solution to one sub-problem is dependent on how other sub-problems are solved, the problem is not amenable to naïve decomposition. That is, we cannot optimize each block serially in a way that is completely independent of other blocks. We can make some progress on a sub-problem by narrowing down the set of configurations that are valuable, but we must maintain a set of competing solutions to each block at least until a combination is found that resolves their interdependencies.

More specifically, the canonical version of the function evaluates binary problem parameters. To evaluate a solution it is divided into two non-overlapping subsets of the problem parameters recursively. Each subset, at each recursive level, corresponds to a building-block, and confers a fitness contribution equal to its size if its bits are either all-ones or all-zeros. This highly structured function leads search to find small building-blocks quite easily. If correct combinations of small blocks can be found, then they may be assembled to find larger blocks, and so on, to solve the whole problem. But not all blocks can be put together successfully – to gain additional fitness contribution from higher levels of the structure, only blocks of the same type (ones or zeros) can be brought together.

The competing solution types within a given partition of the variables represent constrained objectives; a block cannot be good at both ones and zeros simultaneously. But, each non-overlapping partition of the variables corresponds to an independent sub-problem. However, we see that as search progresses, sub-problems that seemed independent at the lower level, are in fact, contained within higher-level blocks, and are therefore mutually constrained. In this way, independent sub-problems and constrained objectives are just perspectives on the same structure.

Solving this class of problem requires an algorithm that is capable of discovering and maintaining competing solutions to many sub-problems simultaneously. In our experiments, we use six hierarchical levels and therefore 64 components at the bottom level of the hierarchy. Maintaining a diversity of competing solutions requires a novel method of segregating competition and cautious block assembly.

3 A Multi-Objective Approach to Problem Decomposition

In recent work we have used techniques from MOO to devise an automatic problem decomposition algorithm that solves our test problem very effectively (3). The algo-

rithm incorporates principles from coevolution, Pareto optimization, and problem decomposition. It has three main features.

First, we introduce a technique to transform the single objective problem into a co-evolutionary game. An individual in this game defines an arbitrary subset of the function parameters. In essence, the game is to specify some subset of parameters that can form good solutions when combined with other subsets of parameters – these other subsets are other players in the coevolutionary game. This permits individuals to collectively represent a decomposition of the problem. Second, the selection of good individuals involves playing each individual in many different games, but rather than selecting individuals based on average performance, we determine the superiority of individuals using Pareto dominance. Specifically, if some individual *A* is at least as good as some individual *B* against all opponents tested (and superior with respect to at least one opponent) then *A* may replace *B*. In this manner, we are using the coevolving opponents as objectives for MOO. This Pareto optimization technique permits a tolerance for individuals that represent alternate solutions to a block and thereby permits alternate ways to satisfy the constraints between the building-blocks. Third, a join operator combines individuals together creating larger sub-sets of the problem parameters. The result of a join will be selected IFF the result Pareto dominates both component individuals. This enables sub-sets of parameters representing solutions to sub-problems to be combined together if their mutual constraints are resolved. With these three features the algorithm is able to automatically discover the building-blocks in the problem, find alternate solutions to each block, search for combinations of solutions that resolve the interdependencies between blocks, and construct a solution to the entire problem.

In effect, this algorithm creates co-adapted players that each represent a potential solution for a subset of the problem parameters, and then uses these players as dimensions for Pareto dominance. This makes the algorithm quite different from other approaches to problem decomposition, cooperative coevolution, and MOO. However, our work is preliminary, and its advantages and limitations with respect to existing techniques have not yet been mapped. In the meantime, our problem domain and algorithmic approach illustrate opportunity for unification and cross-fertilization between the approaches of problem decomposition and multi-objective optimization.

References

1. Watson, RA, Hornby, GS & Pollack, JB, 1998, “Modeling Building-Block Interdependency”, Parallel Problem Solving from Nature, proceedings of Fifth International Conference /PPSN V, Springer, pp.97-106 .
2. Watson, RA, & Pollack, JB, 1999, “Hierarchically-Consistent Test Problems for Genetic Algorithms”, Proceedings of 1999 Congress on Evolutionary Computation (CEC 99). Angeline, Michalewicz, Schoenauer, Yao, Zalzala, eds. IEEE Press, pp.1406-1413.
3. Watson, RA, & Pollack, JB, 2000, “Symbiotic Combination as an Alternative to Sexual Recombination in Genetic Algorithms”, Parallel Problem Solving from Nature, proceedings of Sixth International Conference /PPSN VI, Springer.