A Coevolutionary Model of Real-Time Airport Gate Assignment

Andrés Gómez de Silva Garza

Instituto Tecnológico Autónomo de México (ITAM)
Río Hondo #1, Colonia Tizapán-San Ángel
01000—México, D.F., México
agomez@itam.mx

Introduction and Process Model

In this paper I present a computational model of real-time problem solving that is based on the notion of coevolution. I then discuss the implementation domain of airport gate assignment as an example of the type of dynamic task environment for which the model might be useful.

The computational model I am proposing follows the algorithm shown in Figure 1, which consists of an external (coevolutionary) cycle and an internal (evolutionary) cycle.

```
Generate initial problem randomly
while (not EndOfCoevolution) do
  Generate initial solutions randomly
  while (not (exist OptimaSolution)) do
    Generate new sols. using crossover
    Evaluate new sols.
    Cull population based on evaluation
  end-while
  Generate new prob. using OptimalSol.
end-while
```

Figure 1. Coevolutionary algorithm.

The inner cycle uses an evolutionary algorithm to generate multiple potential solutions to the current problem. These solutions are produced in parallel using the genetic operator of crossover, and are then evaluated based on multiple constraints. The process repeats itself on an updated, improved, population (formed by culling the lowest-quality solutions based on the results of evaluation) indefinitely until an optimal solution is found. If an optimal solution is not found soon enough, the evolutionary algorithm can be stopped (and the best solution known at that moment used) after a fixed number of cycles in order to model the time constraints normally present in real-time problem solving. The outer cycle of the algorithm initiates each simulated time-step by modifying the current task that needs solving partially based on the current state of the solution (determined by the final solution found for the previous problem). This outer cycle represents the coevolution of the task environment together with the solutions for the current task. A graphical view of the algorithm is shown in Figure 2. In the figure solid arrows represent influences during coevolution and dashed arrows represent the flow of time.

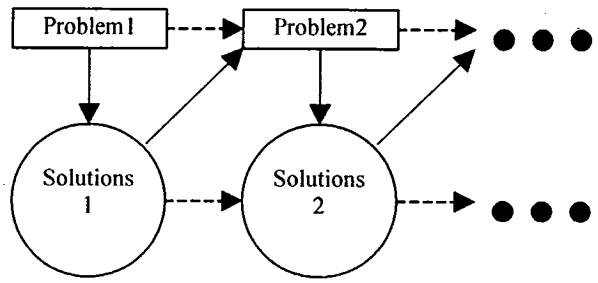

Figure 2. Graphical view of coevolutionary algorithm.

Airport Gate Assignment Domain

The problem of assigning gates to aircraft that are due to arrive at an airport is one that involves a dynamic task environment, and is therefore a suitable implementation domain for the coevolutionary computational model I have presented. Airport gates can only be assigned if they are currently available, and multiple additional constraints must also be satisfied. One example of such a constraint is that certain airlines "own" (or rent) certain gates (or entire terminals) in some airports and cannot be assigned gates located elsewhere in the airport under normal circumstances. In other situations there are gates in which only aircraft of a particular size can park, as the space around them is constrained, so the size of the approaching aircraft must be taken into account in order to assign it an arrival gate at an airport. Another type of constraint occurs when some flights require an arrival gate to be assigned in a specific terminal according to the facilities needed to welcome particular incoming flights (a terminal with immigration and customs areas for an international flight, a terminal with medical facilities for flights that are making an unscheduled stop in order to offload a passenger that is suffering a heart attack, a terminal with qualified personnel and equipment for handling flights that are transporting cargo). A fourth type of constraint involved in airport gate scheduling comes into play if one

wants to minimize the amount of time required by incoming passengers to either depart the airport or to connect to other flights. As soon as all these constraints have been satisfied and appropriate gates have been assigned to the flights that were known to be on the verge of landing just brief moments before, more arriving flights have approached the airspace surrounding the airport in question, and the entire process must be repeated in order to assign arrival gates to them. If flight arrivals at a particular airport always followed the same patterns in time, it would not be too difficult to find once (and in the future always reuse) an optimal solution, but delays, re-routings, changes in airline schedules, and other factors cause airport gate assignment to be a dynamic problem.

In terms of evolutionary algorithms, both the problem that requires solving and the potential solutions proposed for the problem can be visualized as different species. The genotype of the problem species in the airport gate scheduling domain consists of a string or list of flights that are departing or arriving at any given time. The genotype of the solution species consists of associations between flights on the verge of arrival and free airport gates. An evolutionary algorithm generates several possible genotypes for the solution species in parallel, using the genetic operator of crossover, while taking into account the makeup of the genotype of the problem species (i.e., the current problem to be solved), plus all other constraints imposed on the task in the given situation. When an acceptable solution has been found, the process repeats itself, thus modeling the continuous nature of aircraft arrivals and departures at a real-world airport.

When this repetition occurs, the new problem genotype which initiates another evolutionary cycle is partially constructed based on the final solution genotype for the previous evolutionary cycle. For instance, the part of the problem genotype that lists departing flights consists of flights selected from among those that are currently stationed at a departure gate, not just any randomly made-up flights. Therefore, both species in the domain influence each other's genetic makeup as they evolve over time, something essential to coevolution..

Knowledge Representation

The current problem that needs solving at a particular moment in time during the simulation consists of a list or string of flights that are on the verge of arriving or departing at that time. Each of these flights is a gene, and the set of flights is the genotype of the problem species. Arriving flights form part of the problem genotype because they directly indicate the number and characteristics of the flights for which arrival gates have to be scheduled at the present time. Departing flights form part of the problem genotype because they indicate changes to the current state of the airport (by freeing up previously-occupied gates), something which indirectly influences the range of possible solutions that may be proposed for the current problem.

In order to implement the model in the domain of airport gate assignment the part of the problem genotype that lists arriving flights can be generated at random at each coevolutionary cycle (perhaps taken from a list of flights which normally arrive at the airport being simulated on a typical day). The part of the problem genotype that lists departing flights can be generated at random by choosing some flights currently stationed at the airport for departure.

Each coevolutionary cycle corresponds to the existence of one problem genotype for which a solution must be found (i.e., to one time-step in the simulation of the dynamic task environment). For each problem to be solved an initial population of potential solution genotypes can be seeded at random, and then evolved over time until an optimal solution genotype is found. A solution genotype consists of a list or string of associations between available airport gates and the arriving flights from the problem genotype corresponding to the current coevolutionary cycle. The associations are achieved according to position, each location within a solution genotype corresponding to a particular available gate. Solution genes are either empty (when the solution involves not scheduling the particular gate in the corresponding genotype position to receive an incoming flight) or describe flights (which in the solution are being assigned the gate corresponding to their genotype position), and as such are represented in the same way as flights are represented when they belong to problem genotypes. Since different gates can have different characteristics, these must also be explicitly represented, but the gate description does not vary over time or with each different problem, and therefore does not form part of a solution genotype (though it is associated to the contents of a solution genotype by the fact that each gene position in the genotype corresponds to an airport gate).

The genotypes of potential solutions are generated (after the initial, randomly-produced population) using the evolutionary operator of crossover to produce multiple possible gate assignments. Resulting genotypes in which some of the arriving flights are repeated (i.e., assigned to more than one airport gate) as a result of the crossover mechanism or do not appear will of course be discarded quickly, as they will be assigned low fitness values during evaluation.

Fitness values are assigned to each proposed solution based on a weighted calculation of how much each solution satisfies each of the multiple constraints on the problem. Some of the constraints used are static, as they remain the same throughout the simulation (e.g., certain gates admit only certain types of aircraft), whereas others are dynamic (e.g., minimizing the distance that arriving passengers will have to walk to depart the airport and/or transfer to other flights, which depends on the particular requirements or characteristics of the passengers in the arriving flight).