Genetic Algorithm Applications to Optimization Modeling

Pi-Sheng Deng
California State University at Stanislaus, USA

INTRODUCTION

Genetic algorithms (GAs) are stochastic search techniques based on the concepts of natural population genetics for exploring a huge solution space in identifying optimal or near optimal solutions (Davis, 1991)(Holland, 1992)(Reeves & Rowe, 2003), and are more likely able to avoid the local optima problem than traditional gradient based hill-climbing optimization techniques when solving complex problems.

In essence, GAs are a type of reinforcement learning technique (Grefenstette, 1993), which are able to improve solutions gradually on the basis of the previous solutions. GAs are characterized by their abilities to combine candidate solutions to exploit efficiently a promising area in the solution space while stochastically exploring new search regions with expected improved performance. Many successful applications of this technique are frequently reported across various kinds of industries and businesses, including function optimization (Ballester & Carter, 2004)(Richter & Paxton, 2005), financial risk and portfolio management (Shin & Han, 1999), market trading (Kean, 1995), machine vision and pattern recognition (Vafaie & De Jong, 1998), document retrieval (Gordon, 1988), network topological design (Pierre & Legault, 1998)(Arabas & Kozdrowski, 2001), job shop scheduling (Özdamar, 1999), and optimization for operating system's dynamic memory configuration (Del Rosso, 2006), among others.

In this research we introduce the concept and components of GAs, and then apply the GA technique to the modeling of the batch selection problem of flexible manufacturing systems (FMSs). The model developed in this paper serves as the basis for the experiment in Deng (2007).

GENETIC ALGORITHMS

GAs were simulation techniques proposed by John Holland in the 1960s (Holland, 1992). Basically, GAs solve problems by maintaining and modifying a population of candidate solutions through the application of genetic operators. During this process, beneficial changes to parent solutions are combined into their offspring in developing optimal or near-optimal solutions for the given task.

Intrinsically, GAs explore multiple potentially promising regions in the solution space at the same time, and switch stochastically from one region to another for performance improvement. According to Holland (1992), regions in the solution space can be defined by syntactic patterns of solutions, and each pattern is called a schema. A schema represents the pattern of common attributes or features of the solutions in the same region. Let Σ be an alphabet of symbols. A string over an alphabet is a finite sequence of symbols from the alphabet. An n-ary schema is defined as a string in $(\Sigma \cup \{\#\})^n$, where $\# \notin \Sigma$ is used as a wildcard denotation for any symbol in Σ.

Conceptually, n-ary schemata can be regarded as defining hypersurfaces of an n-dimensional hypercube that represents the space of all n-attribute solutions. Individual solutions in the same region can be regarded as instances of the representing schema, and an individual solution can belong to multiple schemata at the same time. Actually, an n-attribute solution is a member of 2^n different schemata. Therefore, evaluating a solution has the similar effect of sampling 2^n regions (i.e., schemata) at the same time, and this is the famous implicit parallelism of genetic search. A population of M solutions will contain at least 2^n and at most $M \cdot 2^n$ schemata. Even for modest values of n and M, there will be a large number of schemata available for processing in the population. GAs perform an implicit parallel search through the space of possible schemata in the form of performing an explicit parallel search through the space of individual solutions.

The problem solving process of GAs follows a five-phase operational cycle: generation, evaluation, selection, recombination (or crossover), and mutation.

At first a population of candidate solutions is generated. A fitness function or objective function is then defined, and each candidate solution in the population is evaluated to determine its performance or fitness. Based on the relative fitness value, two candidate solutions are selected probabilistically as parents. Recombination is then applied probabilistically to the two parents to form two offspring, and each of the offspring solutions contains some characteristics from its parent solutions. After this, mutation is applied sparingly to components of each offspring solution. The newly generated offspring are then used to replace the low-fitness members in the population. This process is repeated until a new population is formed. Through the above iterative cycles of operations, GAs is able to develop better solutions through progressive generations.

In order to prepare for the investigation of the effects of genetic operations in the sequel of current research, we apply the GA technique to the optimization modeling of manufacturing systems in next section.

A GA-BASED BATCH SELECTION SYSTEM

Batch selection is one of the most critical tasks in the development of a master production plan for flexible manufacturing systems (FMSs). In the manufacturing process, each product requires processing by different sets of tools on different machines with different operations performed in a certain sequence. Each machine has its own limited space capacity in mounting tools and limited amount of available processing time. Under various kinds of resource constraints, choosing an optimal batch of products to be manufactured in a continuous operational process with the purpose to maximize machine utilization or profits has made the batch selection decision a very hard problem. While this problem is usually manageable for manufacturing small number of products, it quickly becomes intractable if the number of products grows even slightly large. The time required to solve the problem exhaustively would grow in a non-deterministic polynomial manner with the number of products to be manufactured.

Batch selection affects all the subsequent decisions in job shop scheduling for satisfying the master production plan, and holds the key to the efficient utilization of resources in generating production plans for fulfilling production orders. In our formulation, we use the following denotational symbols:

- M: the cardinality of the the set of machines available
- T: the cardinality of the the set of tools available
- P: the cardinality of the set of products to be manufactured
- *MachineUtilization*: the function of total machine utilization
- $processing\_time_{product,tool,machine}$: the time needed to manufacture product *product* using tool *tool* on machine *machine*
- $available\_time_{machine}$: the total available processing time on machine *machine*
- $capacity_{machine}$: the total number of slots available on machine *machine*
- *machine, tool, product*: indicators for machines, tools, and products to be manufactured correspondingly
- $slot_{tool}$: the number of slot required by machine tool *tool*
- $quantity_{product}$: the quantity of product *product* to be manufactured in a shift
- $Q_{product}$: the quantity of product *product* ordered by customers as specified in the production table

Fitness (or Objective) Function

The objective is to identify a batch of products to be manufactured so that the total machine utiliztion rate will be maximized. See Exhibit A.

The above objective function is to be maximized subject to the following resource constraints:

1. Machine capacity constraint (see Exhibit B)

The above function $f(\bullet)$ is used to determine if tool *tool* needs to be mounted on machine *machine* for the processing of the current batch of product.

2. Machine time constraint (see Exhibit C)
3. Non-negativity and integer contraints

Encoder/Decoder

The Encoder/Decoder is a representation scheme used to determine how the problem is structured in the GA