Optimisation of Mixed Integer Linear Programs using Genetic Algorithm

Rajagopalan Srivinasan¹, I. A. Karimi¹, Tan P. J.²

Department of Chemical and Environmental Engineering
Department of Electrical and Computer Engineering
National University of Singapore

Abstract

Many engineering problems can be modeled as mixed integer linear programs (MILP). Conventional solution methods to such problem include heuristic methods such as branch-and-bound. In this paper, we propose a genetic algorithm (GA) for optimization of mixed integer linear programs. This paper will first discuss GA in brief giving accounts of various aspects of GA. This is followed by the MILP formulation and then the proposed GA method for MILP. In GA-MILP, basic variable selection and integer selection are placed under the same framework instead of two distinct components. We also introduce an adaptive GA here where certain potential search space is being concentrated when sufficient feasible solutions have been gathered. Some results will be given comparing the normal GA and our proposed GA.

Genetic algorithm

The idea of a genetic algorithm originates from Darwinian evolutionary theory which suggests the survival of the fittest. In designing a GA, solutions are represented by chromosomes and associated with each chromosome is its fitness value. These chromosomes form a pool and they evolve from one generation to another through operations such as crossover and mutation. Through these iterations, the fittest chromosomes will survive and the solutions they represent are the most optimal ones.

An efficient GA employs novel encoding of a solution in the form of a chromosome, good definition of fitness value as well as appropriate crossover and mutation schemes. Other techniques which can enhance optimization include elitism and repair mechanism. Elitism keeps the fittest chromosomes in each generation without letting them interact with the rest of the pool. Repair mechanism involves changes in the chromosome to satisfy certain properties, and this technique is particularly useful in constraint programs.

MILP formulation

The following parameters are associated with an MILP.

m number of continuous variables
n number of binary variables
c total number of constraints
e number of equality constraints
The mathematical formulation of MILP is as follow:

Minimize $A^T x + B^T y$
Subject to $C x' + D y = W$
Where $A$, $x$ are $m \times 1$ column vectors
$B$, $y$ are $n \times 1$ column vectors
$x'$ is $(m+c-e) \times 1$ column vector
$C$ is a $c \times (m+c-e)$ matrix
$D$ is a $c \times n$ matrix
$W$ is a $c \times 1$ column vector

The entries in $x$, $x'$ are continuous variables and entries in $y$ are binary variables. The original problem might have inequality constraints and hence $(c-e)$ slack variables must be added. Thus, $x'$ consists of the continuous variables and slack variables. $C$ is the coefficient matrix for the continuous variables and slack variables.

**GA-MILP**

We observe that there are two parts to the problem, one involving only binary variables and one involving only continuous variables. Once the binary variables are fixed, we will then have a purely linear program which can be solved using revised simplex method. Here we will use a modified version where the basic variables are pre-selected and encoded in the chromosome.

**Chromosome representation**

Each chromosome has two parts – one containing binary values corresponding to each binary variable and another containing the basic variables to be selected. In an MILP problem, this will correspond to $n$ binary values and $(m+c-e)$ basic variables. For example,

1 1 0 1 1 0 0 1, 1 2 4 5 7 9 10

in the above chromosome, we will have $y_1=y_2=y_4=y_5=y_8=1$ and $x_1, x_2, x_4, x_5, x_7, x_9, x_{10}$ are the basic variables. Here, $x_i$’s are the $i^{th}$ entry in the vector $x$, $y_i$’s are the $j^{th}$ entry in the vector $y$.

**Fitness value**

We compute the objective function as follow:

Substitute binary values $y=y'$
Select the columns of $C$ corresponding to the basic variables, forming matrix $C'$
Solve $C'x''=W-Dy'$ for $x''$ where $x''$ are the basic variables
Substitute the values for $x$ and $y$ into $A^T x + B^T y$ to get the value of objective function

However one of the following three possibilities could occur and the fitness value of each chromosome is defined as follow:

a) $x''\geq 0$: fitness value = value of objective function

b) $x''<0$: fitness value is represented by the constraint that is being violated

c) $C'$ non-invertible: fitness value is represented by the constraint that is being violated
**Tournament selection**
Two randomly selected from the pool will compete to be passed on to the next generation based on their fitness value. The winner of the tournament survives but the loser will still stand a chance to compete if there are more tournament selections.

**Crossover**
Crossover takes place separately at 2 locations one in each part of the chromosome as shown below.

Before crossover:

```
  1 1 | 0 1 1 0 0 1, 1 2 4 5 7 | 9 10
  0 1 | 1 0 1 1 0 0, 1 3 5 6 8 | 10 11
```

After crossover:

```
  1 1 0 1 1 0 0, 1 2 4 5 7 10 11
  0 1 0 1 1 0 0 1, 1 3 5 6 8 9 10
```

**Mutation**
Mutation occurs randomly across the whole chromosome. For example,

```
  1 1 0 1 1 0 0 1, 1 2 4 5 7 9 10 becomes
  1 0 0 1 1 1 0 1, 1 2 5 6 8 9 12
```

**Repair mechanism**
After above mentioned operations, there is a possibility of repeated basic variables. Repair method works on such chromosomes by changing the basic variables so that the whole chromosome is valid.

**Algorithm**
We now present the algorithm GA-MILP incorporating the various genetic operations mentioned earlier. GA-MILP first searches for solution with a low mutation but slowly increasing the mutation. Once about 5% of the population is feasible, the algorithm will concentrate searching in this potential region by having tournament selecting among mostly the best chromosomes, meaning that crossover will take place amongst the best 5% of the chromosomes and some of them will get mutated. While carrying out crossovers, GA-MILP makes sure that if two chromosomes are not feasible, they will crossover only if the chromosomes violate different feasibility constraints as represented by their fitness values. GA-MILP terminates if maximum generation has reached or about top 15% of chromosomes have converged to the same fitness. Elites are kept so that we do not lose any good solutions.

Algorithm GA-MILP
Generate 400 chromosomes
Select the best 200 chromosomes as the initial population
While not converged or maximum generation
  If not concentrating
    Tournament selection ensures chromosomes are in complimentary pairs
    Crossover between complimentary chromosomes
Mutation of other chromosomes
Increase mutation

Else
Tournament selection to get the best 5% of the chromosomes
Crossover amongst them
Mutation of other chromosomes

Results

Two GA programs are used to optimize the following four problems with different sizes in terms m, n and c with e=0. Normal GA simply carries out crossover and mutation through each generation without any concentrating any potential region whereas GA-MILP does that as we have discussed earlier. GA-MILP does better in all sections, including producing better average and best results in a shorter time. Moreover, graphs of best fitness value in each generation show that GA-MILP consistently produce better results in each generation as compared to the normal GA.

<table>
<thead>
<tr>
<th>problem</th>
<th>m</th>
<th>c</th>
<th>n</th>
<th>GA-MILP</th>
<th>Normal GA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Best Average</td>
<td>Best Average</td>
</tr>
<tr>
<td>1</td>
<td>40</td>
<td>8</td>
<td>40</td>
<td>18.0 (0.8) 62.1 (0.5)</td>
<td>110 (2.5) 162.8 (2.4)</td>
</tr>
<tr>
<td>2</td>
<td>1000</td>
<td>8</td>
<td>40</td>
<td>31.0 (1.1) 45.9 (2.3)</td>
<td>271.9 (3.5) 376.5 (3.5)</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td>12</td>
<td>40</td>
<td>15.0 (3.9) 73.4 (3.9)</td>
<td>356 (3.8) 427.8 (3.9)</td>
</tr>
<tr>
<td>4</td>
<td>40</td>
<td>8</td>
<td>400</td>
<td>131.0 (5.3) 156.8 (5.43)</td>
<td>344.8 (5.9) 424.4 (5.5)</td>
</tr>
</tbody>
</table>

Table 1. Results of the 4 problems

![Figure 1. Problem 1](image1)
![Figure 2. Problem 2](image2)
In conclusion, a GA based solution to optimization of MILP has been discussed. MILP consists of two major components which are integer selection problem and basic variable selection. GA-MILP places both parts into the same framework. Moreover, our proposed GA searches for a diverse solution space before concentrating into a potential region as compared to the normal GA which does not concentrate into certain region. Results show that GA-MILP is a potential solution method in terms of the quality of solution and the time taken.

Figure 3. Problem 3

Figure 4. Problem 4

Conclusion