A New Genetic Algorithm for Set Covering Problems

Annual Operational Research Conference 42, Swansea, UK, 2000.

An indirect genetic algorithm for the non-unicost set covering problem is presented. The algorithm is a two-stage meta-heuristic, which in the past was successfully applied to similar multiple-choice optimisation problems. The two stages of the algorithm are an 'indirect' genetic algorithm and a decoder routine. First, the solutions to the problem are encoded as permutations of the rows to be covered, which are subsequently ordered by the genetic algorithm. Fitness assignment is handled by the decoder, which transforms the permutations into actual solutions to the set covering problem. This is done by exploiting both problem structure and problem specific information. However, flexibility is retained by a self-adjusting element within the decoder, which allows adjustments to both the data and to stages within the search process. Computational results are presented.

Dr Uwe Aickelin

School of Computer Science
University of Nottingham
NG8 1BB UK
uxa@cs.nott.ac.uk

Background

Aim:

- Develop a Genetic Algorithm to solve Set Covering Problems.

Background:

- Successfully Dealing with Constraints (Nurse Scheduling).

- Next Step: Set Partitioning Problems.

The Set Covering Problem

The problem of covering the rows of an m-row, n-column, zero-one matrix a_{ij} by a subset of the columns at minimal cost.

Defining $x_j = 1$ if column j with cost c_j is in the solution and $x_j = 0$ otherwise.

$$\text{Minimise} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad \forall i$$

$$x_j \in \{0,1\} \qquad \forall j$$

A New Genetic Algorithm for Set Covering Problems

Idea:

- If the order of rows is right, a (relatively) simple heuristic can solve the problem.

Method:

- Two stage approach.
- Genetic Algorithm finds the ‘best’ permutation of rows.
- Decoder Routine assigns ‘best’ columns to rows in given order.

Genetic Algorithms (GA)

- A Heuristic based on the principles of natural evolution and ‘survival of the fittest’.

- Population: The GA works with many solutions at the same time. New Solutions inherit good parts from old solutions.

- Coding: Transformation such that genetic operators can be applied to solutions. (Here: A permutation of the rows.)

- Fitness: The fitter a solution, the more likely it will contribute to new solutions. (Here: The cost of the columns to cover the rows.)

Genetic Algorithms (GA)

- Selection: Individuals (solutions) are ranked according to fitness. The higher the rank the more likely they are chosen as a parent.

- Crossover: Combining parts of parent individuals (cut and paste) to create new solutions: ‘Building Block Hypotheses’.

- Mutation: Minor random changes of an individual.

- Replacement: ‘Elitist’ Strategy, i.e. the best 20% of the old solutions are kept. The rest are replaced.

The Decoder

- Must be computationally efficient.

- Must be deterministic, i.e. same permutation always yields same solution.

- Should produce good solutions.

How to choose a column?

Cycle through all possible candidate columns.

Choose the one with the highest score based on:

- Cost of column.
- How many uncovered rows does it cover?
- How many rows does it cover in total?
- Will its inclusion make another column redundant?

How to balance these multiple criteria?

- Weights will depend on data and progress of algorithm.
- Parameter optimisation is too complex.

Solution:

- Genetic algorithm optimises weights parallel to permutations.
- For *w* weights attach *w* extra genes to a string and randomly initialise weights.
- Children inherit the rank-weighted average of their parents.
- More important score parts receive higher weights.

Further Enhancements

- How many rows does a column share with those already chosen?

- How does a column's cost rank for all uncovered rows it would cover?

- How does a column's cost rank for all rows it would cover?

- Remove redundant columns with a simple hill climber.

- Use a variety of crossover operators (from conservative to aggressive), controlled by the GA.

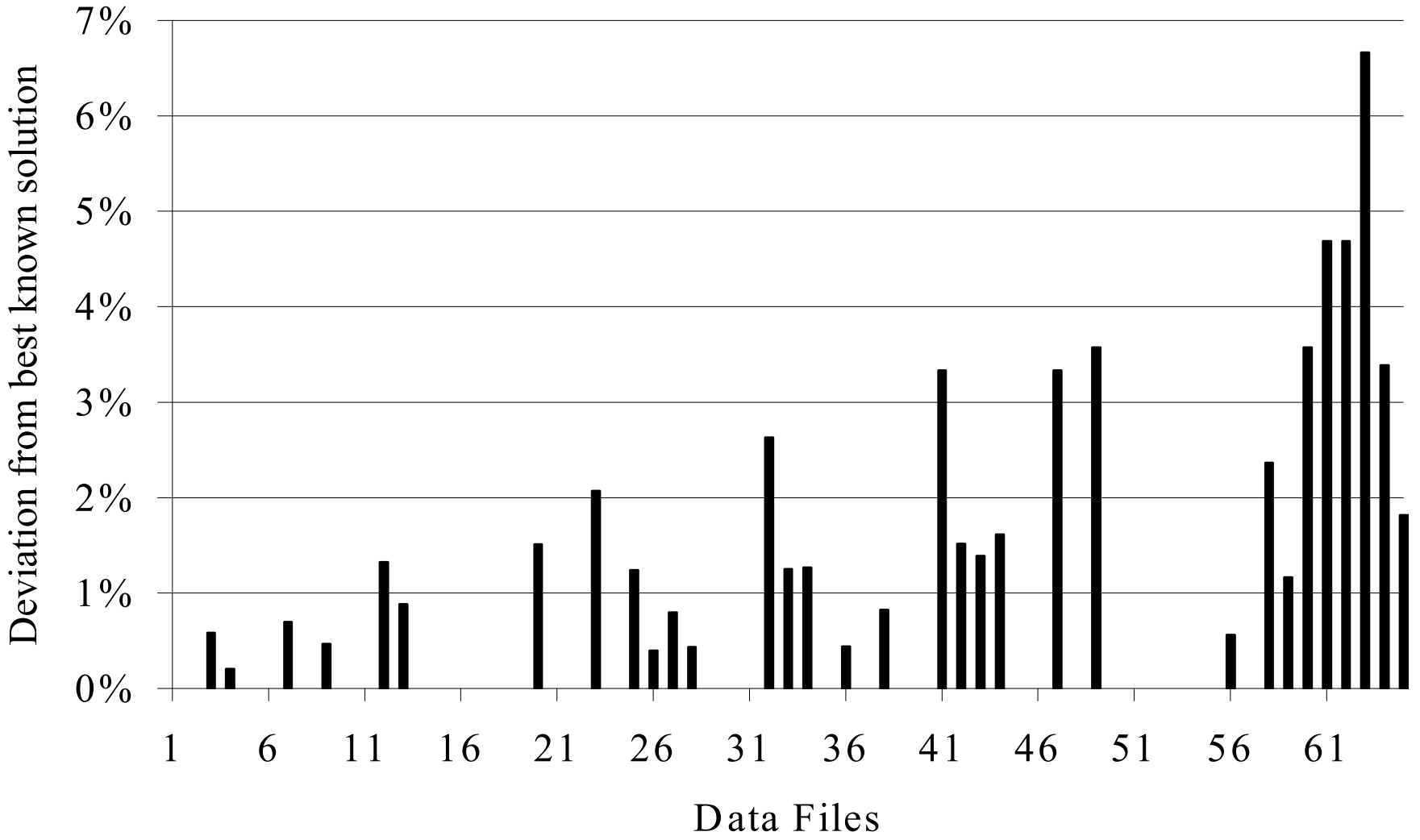
Results

- 65 data sets from 200 rows / 1000 columns to 1000 rows / 10000 columns.

- 10 run average over all files within 2% of optimal / best known solutions.

- Best of each run on average within 1% of optimal / best known solution.

- Solution time on average 38 seconds on a 450 MHZ PC.


Deviation from best known solution
7%
6%
5%
4%
3%
2%
1%
0%
1
6
11
16
21
26
31
36
41
46
51
56
61
Data Files

Remaining Questions

- How often / early should the hill climber be used?

- Other criteria to select columns?

- Allow mutation to alter solutions directly?