Scheduling Planned Maintenance of the National Grid

W. B. Langdon*

Dept of Computer Science, University College London,
Gower Street, London WC1E 6BT

Abstract. The maintenance of the high voltage electricity transmission network in England and Wales (the National Grid) is planned so as to minimise costs taking into account:

- location and size of demand for electricity,
- generator capacities and availabilities,
- electricity carrying capacity of the remainder of the network, i.e. that part not undergoing maintenance.

This complex optimization and scheduling problem is currently performed manually (computerised viability checks can be performed after the schedule has been produced). This paper reports work aiming to automatically generate low cost schedules using genetic algorithms (GA). So far:

- A small demonstration problem has been identified,
- A fitness function has been devised,
- To date work has concentrated upon devising a representation based upon "greedy optimizers", which combine permutation GAs with scheduling heuristics,
- The best of these heuristics has been incorporated in the QGAME genetic algorithm programming environment and optimal solutions have been readily found.

1 Introduction

In England and Wales electrical power is transmitted by a high voltage electricity transmission network which is highly interconnected and carries large power flows. It is owned and operated by The National Grid Company plc. (NGC) who maintain it and wish to ensure its maintenance is performed at least cost, consistent with plant safety and security of supply.

There are many components in the cost of planned maintenance. The largest is the cost of replacement electricity generation, which occurs when maintenance of the network prevents a cheap generator from running so requiring a more expensive generator to be run in its place.

The task of planning maintenance is a complex constrained optimization scheduling problem. The schedule is constrained to ensure that all plant remains within its capacity and the cost of replacement generation, throughout

[1] *email:* W.Langdon@cs.ucl.ac.uk

the duration of the plan is minimised. At present maintenance schedules are produced manually by NGC's Planning Engineers (who use computerised viability checks on the schedule after it has been produced). This paper describes work by NGC's Technology and Science Laboratories and University College London which investigates the feasibility of generating practical and economic maintenance schedules using genetic algorithms [Hol92].

The work to date is based upon a demonstration four node test problem [Dun93]. Section 3 presents the fitness function devised for the four node problem and indicates changes that may be required when considering larger networks. Section 4 describes the four node problem in detail. Section 5 considers various possible genetic algorithms and explains why work has concentrated upon using "greedy optimizers". Section 6 details the four optimizers that have been tried and the results achieved. The best of these readily produces optimal solutions.

Section 8 considers scaling up from the demonstration problem and highlights concern about the computational complexity of the greedy optimizers. We have started using the best of our hybrid approaches on a real world problem of intermediate size but are also considering alternative approaches (such as expansive coding [BBM93] and genetic programming [Koz92]).

2 Approximating Replacement Generation Costs

NGC use computer tools for costing maintenance schedules, however because of their computational complexity, it was felt that these were unsuitable for providing the fitness function. Instead our fitness function is partially based upon estimating the replacement generation costs that would occur if a given maintenance plan were to be used. The estimate is made by calculating the electrical power flows assuming the maintenance will not force a change in generation. In practice alternative generators must be run to reduce the power flow through over loaded lines in the network. The cost of the alternative generators is modeled by setting the amount of replacement generation equal to the excess power flow and assuming alternative generators will be a fixed amount more expensive than the generators they replace.

As a first approximation issues concerning security of supply following a fault are not covered. We plan to increase the realism of the GA as we progress. Other costs (e.g. maintenance workers' overtime, travel) are not considered.

3 The Fitness Function

The GA's fitness function is composed of two parts; a benefit for performing maintenance plus penalties for exceeding line ratings, isolating nodes and splitting the network.

3.1 Maintenance Benefits

The maintenance requirements of the different components of the transmission network vary both in terms of the number of weeks required to perform them and their urgency. It may be advisable to hold over less urgent maintenance until the following year. We model the urgency by allocating each maintenance activity a benefit term in the fitness function.

During the plan each part of the network should be maintained a small number of times, typically once. Each maintenance activity takes a few weeks and has a particular fitness benefit. Should a trial maintenance plan schedule a line for maintenance for its required number of weeks, that trial plan's fitness is improved by the maintenance benefit associated with that line. Should a line require more than one maintenance period they must not overlap, however there may be an additional fitness benefit if they run consecutively (perhaps of about 20%). Once the target number of weeks of maintenance has been reached, there is no additional benefit or penalty from additional maintenance. The total benefit is obtained by summing across all lines for the whole year.

In the rest of this paper it is assumed that each line in the network can be maintained independently, however there are some cases where this is not so (notably "Tee" junctions). These require several items to be taken out of service simultaneously. As there are relatively few of these they can be treated as special cases.

3.2 Over Loading Costs

In order to calculate the line overloading costs, we must first determine which generators are to be used and when. This is done by using the available generators in strict price order (cheapest first) until the predicted demand for each week is met. This is known as the merit order dispatch. It is fixed and therefore the same for all trial schedules.

For each week of a trial schedule the predicted demand and the merit order dispatch are used in a "DC load flow" analysis which calculates the power flow through every line in the network. The over loading cost for each line is proportional to the amount the power flow though it exceeds its normal operating limit (it is zero if within the limit). The total over loading costs are the sum over each week of the maintenance plan and over all lines in the network.

3.3 Avoiding Isolating Nodes or Splitting the Network

From an operational point of view, no acceptable maintenance schedule would ever isolate a generation or demand node from the rest of the network or split the network. However the GA fitness function must be able to cope with every schedule that is generated. The "DC load flow" algorithm cannot cope with either as they require it to invert a singular matrix. Therefore the fitness function looks for these conditions and defines a fitness for them without calculating power flows.

As both represent highly unfit solutions, weeks of a schedule that cause either contribute a high penalty to the schedule's whole fitness. The penalty for each isolated node is proportional to the load or expected generation at that node.

The penalty for a network split is even more severe; it is proportional to the total load across the whole network in that week.

3.4 Combined Fitness Measure

The complete fitness measure is expressed in Megawatt weeks and is given by the following formula:

$$
\begin{aligned}
\text{cost} = & \sum_{\text{target}} K_t \times (1-\text{maintenance\_scheduled}) \\
& + \sum_{\text{weeks}} \left(\begin{array}{l}
\textbf{If } \text{network\_split } \textbf{then} \\
\qquad S_1 \times \text{ total\_demand} \\
\textbf{Else If } \text{isolated\_nodes} \geq 1 \textbf{ then} \\
\qquad S_2 \times \displaystyle\sum_{\text{isolated\_nodes}} \mid \text{demand} \vee \text{generation} \mid \\
\textbf{Else} \\
\qquad \displaystyle\sum_{\text{lines}} \textbf{If } |\text{flow}| > \text{rating } \textbf{then} \\
\qquad \qquad |\text{flow}| - \text{rating}
\end{array} \right) .
\end{aligned}
\tag{1}
$$

The first summation being over all target maintenance (n.b. the trial plan's cost is increased by K_t if the corresponding maintenance is not scheduled). The second outer summation being over each week of the maintenance plan; the first inner one, being over all isolated nodes, and the second, over all lines in the network.

For the four node system (next section) all the constants K_t are equal and S_1 and S_2 are equal to each other. This may not be the case in the full system.

4 The Four Node Demonstration Network

A small power system has been devised. The network is shown in Fig. 1. As the problem's network contains four substations (i.e. nodes in the network multi-graph), it is know as the "four node problem". Despite its small size, this problem includes many of the features, such as tight time limits, seasonal demand patterns and high power flows, which characterise the full network.

This power system contains nine generators of differing capacities, availabilities and prices. The substations are linked by pairs of transmission lines (i.e. links in the network multi-graph) to form a square. The transmission lines are of two sorts; 1100MW and 600MW nominal rating but all have the same impedance (nominal ratings are some 17% above normal operating limits). Table 1 gives the transmission line nominal ratings. Table 2 shows the predicted demand, which varies from week to week and node to node.

The problem is to devise a nine week maintenance schedule which, maintains each line for at least one week and minimises predicted line over loading costs whilst not splitting the network or isolating nodes in it. I.e. has minimum cost.

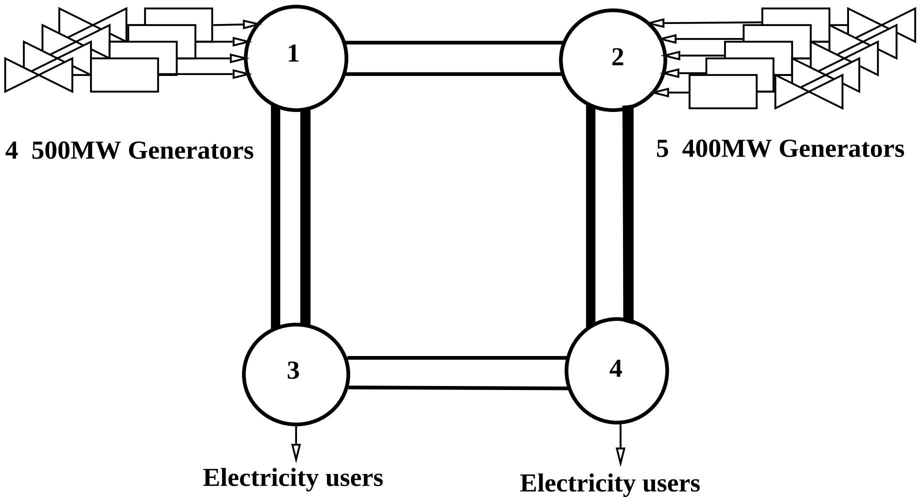

Fig. 1. Four node network

Table 1. Nominal line ratings (MW) and impedances

Line	Start node	End node	Rating	Impedance
1	1	2	600	1
2	1	2	600	1
3	2	4	1100	1
4	2	4	1100	1
5	4	3	600	1
6	4	3	600	1
7	3	1	1100	1
8	3	1	1100	1

4.1 Determining which Generators to Use

In each week of the maintenance schedule, the demand for power (Table 2) is met by the cheapest available generators. Table 3 shows which generators run, and how much power they generate in each week.

We now have enough information to calculate the power flows in every power transmission line in the network in every week for any given maintenance schedule. Once we have established the constants K and S, the fitness or cost of the schedule can be readily calculated from the known power flows using (1).

4.2 Determining Fitness Function Parameters K and S

The constants of proportional, K and S need to be chosen so that the maintenance benefit dominates. This was done by calculating the line costs associated

Table 2. Predicted demand and generator price, capacity and availability

Week	Demand				Node 1 Gens				Node 2 Generators				
	Node 3 MW	Node 4 MW	Total MW	Name	1.1	1.2	1.3	1.4	2.1	2.2	2.3	2.4	2.5
				Price	1.0	2.0	3.0	4.0	3.5	4.5	5.5	6.5	7.5
				Capacity	500	500	500	500	400	400	400	400	400
1	1200	1300	2500		x								
2	1200	1600	2800			x							
3	1000	2000	3000				x						
4	1100	2300	3400					x					
5	1100	2400	3500						x				
6	1000	2500	3500							x			
7	900	2600	3500								x		
8	900	2700	3600									x	
9	800	3000	3800										
				x $\Rightarrow$ cannot produce electricity this week									

Table 3. Merit order generator loading (MW)

Week	Node 1 Generators					Node 2 Generators						Total
	1.1	1.2	1.3	1.4	Total	2.1	2.2	2.3	2.4	2.5	Total	
1	x	500	500	500	1500	400	400	200	0	0	1000	2500
2	500	x	500	500	1500	400	400	400	100	0	1300	2800
3	500	500	x	500	1500	400	400	400	300	0	1500	3000
4	500	500	500	x	1500	400	400	400	400	300	1900	3400
5	500	500	500	500	2000	x	400	400	400	300	1500	3500
6	500	500	500	500	2000	400	x	400	400	300	1500	3500
7	500	500	500	500	2000	400	400	x	400	300	1500	3500
8	500	500	500	500	2000	400	400	400	x	400	1600	3600
9	500	500	500	500	2000	400	400	400	400	200	1800	3800
	x $\Rightarrow$ cannot produce electricity this week											

with many feasible schedules (using the heuristic given in Sect. 6.1). This gave the highest feasible line cost as being 3870MW weeks. K was set to 4,000 MW weeks, so that any schedule which failed to maintain even one line would have a higher cost that the worst "reasonable" schedule which maintains them all.

S was chosen so that isolating any node would have a high cost. The lowest demand/generation on a node is 800 MW. Setting S to five ensures that any schedule which isolates any node will have a cost higher the worst "reasonable" schedule (800 MW$\times$5 weeks = 4,000 MW weeks).

5 The Chromosome

Various ways of representing schedules within the GA, were considered. When choosing one, we looked at the complete transmission network which contains about 300 nodes and about 500 transmission lines which are maintained over a 52 week plan. Table 4 summarizes the GA representations investigated (Shaw [Sha94] gives a bibliography of GAs used on many scheduling problems).

Table 4. Possible chromosomes for the maintenance scheduling problem

Organization	Size	Gene	Expected Benefit	Disadvantages	Selected References
Linear	25,000	bit	Simple	Crossover disruptive, Too long, No structure, Sparse	[Gol89]
2D	500 by 52	bit	Simple, Obvious temporal structure	Crossover disruptive, Still very big, No grid structure, Sparse	[CH93] [VJ93] [And94]
Graph	500 by 52	bit	Realistic	Crossover disruptive, Complex, Little preceding work, Sparse	[MK93]
TSP	25,000	link	Widely studied	"By edge" costs, Sparse	[SMM+91] [VJ93]
Genetic Programming		S-expression		Complex, Little preceding work	[ABN94]
Expansive Coding	$25,000 \times n$	complex	Crossover friendly	Complex, Little preceding work	[BBM93]
"Greedy Optimization"	500	line id	Good crossover, Compact, Widely studied	Fitness evaluation expensive	[Sys91] [FRC93] [FRC94]

In the linear, 2D and TSP structures every attempt should be made to keep electrically close lines close to each other in the representation. The graph representation has the advantage that this can be naturally built into the chromosome.

With a simple linear chromosome, Holland or multi-point crossover [Gol89], would appear to be too disruptive to solve the problem. A simple example illustrates this. Suppose lines L_1 and L_2 are close together. It will often be the case that good solutions maintain one or the other but neither at the same time. But crossover of solutions maintaining L_1 but not L_2 with those maintaining L_2 but not L_1 will either make no change to L_1 and L_2 or not maintain either or maintain both. The first makes no change to the fitness whilst the second and third make it worse. That is, we would expect crossover to have difficulty assembling better solutions from good "building blocks" [Gol89]. The same problem appears in the temporal dimension and so affects the first three representations given in Table 4. This coupled with the sparseness of the chromosome (about 1 bit in 50 set) and other difficulties, has led to the decision to try a "greedy optimization" representation.

6 Greedy Optimizers

The approach taken so far to solving the power transmission network maintenance scheduling problem has been to split the problem in two; a GA and a "greedy optimizer". The greedy optimizer is presented with a list of work to be done (i.e. lines to be maintained) by the GA. It schedules those lines one at a time, in the order presented by the GA, using some problem dependent heuristic. Figure 2 shows this schematically, whilst the dotted line on Fig. 3 shows an *order* in which lines are considered.

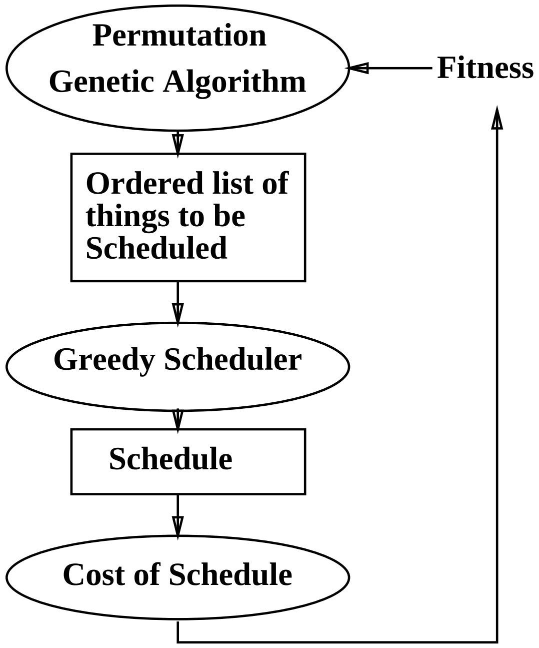

Fig. 2. Hybrid GA and "greedy optimizer"

This approach of hybridising a GA with a problem specific heuristic has been widely studied. Davis [Dav91] for example firmly advocates using hybrid GAs when attempting to solve difficult real world problems. Hybrid GAs, of various sorts, have been used on a number of scheduling problems (e.g. flight

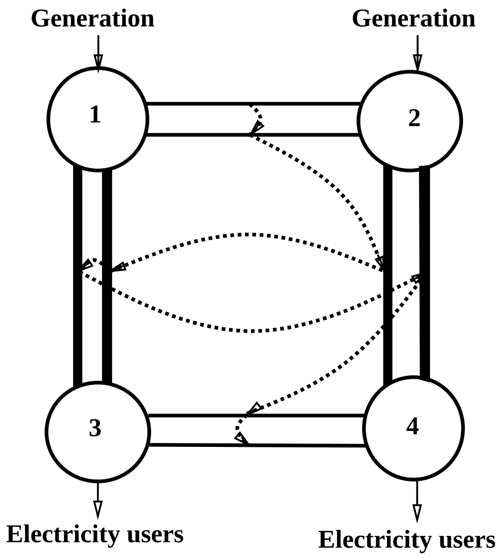

Fig. 3. Example order (dotted) in which lines are considered by a "greedy optimizer"

crew scheduling [Lev94], task scheduling [Sys91] and job-shop and open-shop scheduling [FRC93] [FRC94] [YN92]).

A variety of heuristics of increasing sophistication and computational complexity have been tried on the four-node problem which yielded progressively better results.

6.1 Heuristic 1 – One Week, One Line

If we ensure a schedule maintains all lines and does not isolate nodes its fitness will be independent of K and S. Such schedules are readily devised by only maintaining one line at a time. Further such schedules are reasonable solutions to the problem (indeed it was anticipated that the best solution would be of this type). By testing a number of these we used the results to guide choice of K and S. Figure 4 shows the distribution of line costs lie between 1470 and 3870 MW weeks.

In Figs. 4, 5, 6, 7, 8, 9 and 10 we plot cumulative frequency against over loading line costs (or fitness). Often GA fitness landscapes are very rough however the roughness or texture of a fitness landscape is a product of the representation used as well as the fitness itself. Using cumulative frequency makes the graph independent of representation and, in this case, makes it smooth.

The least cost solutions have the form of the schedule given in Table 5. These maintain the two right hand (referring to Fig. 1) high rating lines (numbered 3 and 4, see Table 1) in the first and fourth weeks; the left hand two (7, 8) in weeks two and three; the bottom cross lines (5, 6) in the fifth and six weeks,

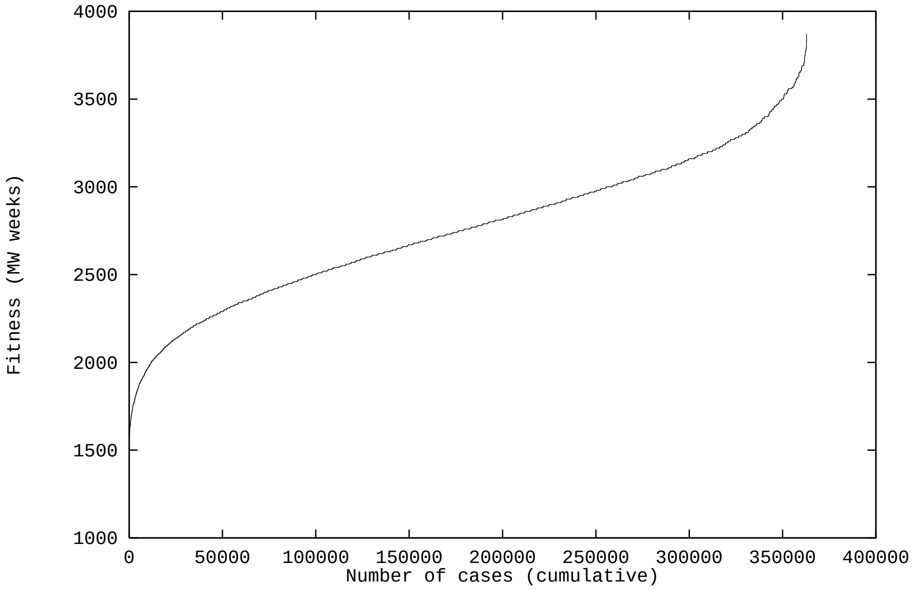

Fig. 4. Cumulative frequency v. line costs (one week, one line)

nothing in week seven and the top two cross lines (1, 2) in the last two weeks. We shall return to the two top cross lines later.

Table 5. Best schedule produced by one week, one line heuristic

Week	1	2	3	4	5	6	7	8	9
Lines	3	7	8	4	5	6		1	2
Fitness	1470 MW weeks								

6.2 Heuristic 2 – Minimum Power Flow

A greedy optimizer was devised which scheduled the maintenance of each line in the week in which the power flow through it is a minimum. In the event a tie, the earlier week is chosen. As each line is scheduled, the power flows through the rest of the network are recalculated. As Fig. 5 shows, the fitness of the schedules this algorithm produces varies from 1440 to 1710 MW weeks, with 40% being 1480 or less. No schedule causes a node to be isolated or the network to be split.

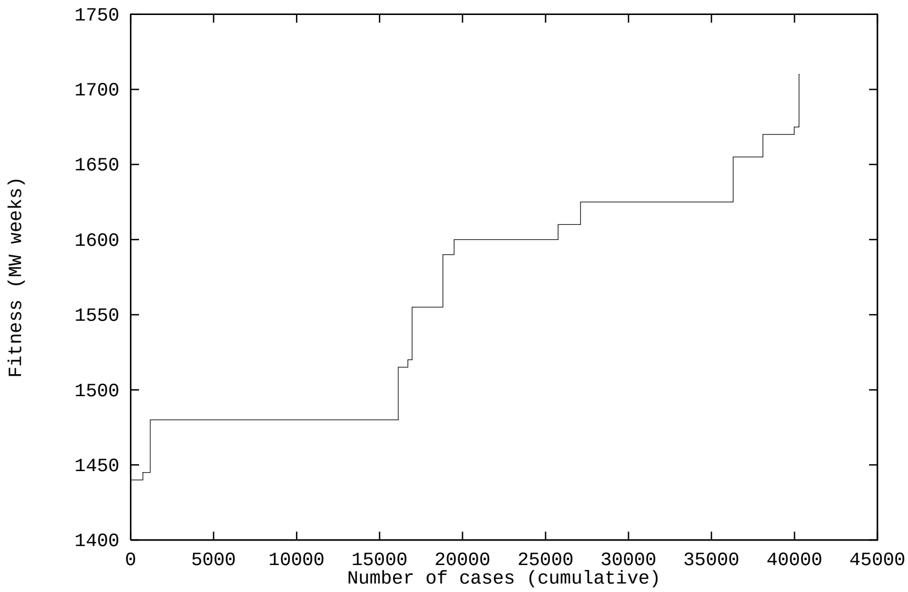

Fig. 5. Cumulative frequency v. fitness (minimum power flow)

Both the best and the worst solutions produced are better than the best and worst given by the one line, one week algorithm. In fact most solutions are better than all but 0.08% of the solutions given by the previous algorithm.

The lower costs have been achieved by maintaining lines when demand (and hence power flows) are least. Table 6 shows one of these. Even the highest cost schedule maintains the eight lines in the first four weeks, when demand is lowest. This reduces the number of possible schedules so there are many solutions which have identical fitness. This is further increased by the symmetries of the problem. For example there are 736 lowest cost schedules.

Table 6. Best schedule produced by minimum power flow heuristic

Week	1	2	3	4	5	6	7	8	9
Lines	4	3	5 6 7	1 2 8					
Fitness	1440 MW weeks								

These are good solutions which may prove to be acceptable but they are known not to be optimal. Whilst good solutions to the problem as posed, they would probably not be acceptable operationally because the maintenance is too bunched together. This reduces the resilience of the network, for example a single circuit failure could disconnect electricity consumers. It may also cause other problems, such as requiring too many maintenance gangs. However these issues are beyond the scope of the four node problem.

6.3 Heuristic 3 – Minimum Line Cost

The greedy lowest power flow optimizer was replaced by the more complex minimum line cost optimizer. This schedules maintenance in the week which currently contributes least to total costs. If there is a tie, the earliest week is used.

As Fig. 6 shows this algorithm produced only a marginal reduction in the best solution found (1411.67 MW weeks) and the cost of the worst schedule is actually worse (2265 MW weeks). Instead of 40% having a fitness of 1480 or better, now only 10% do.

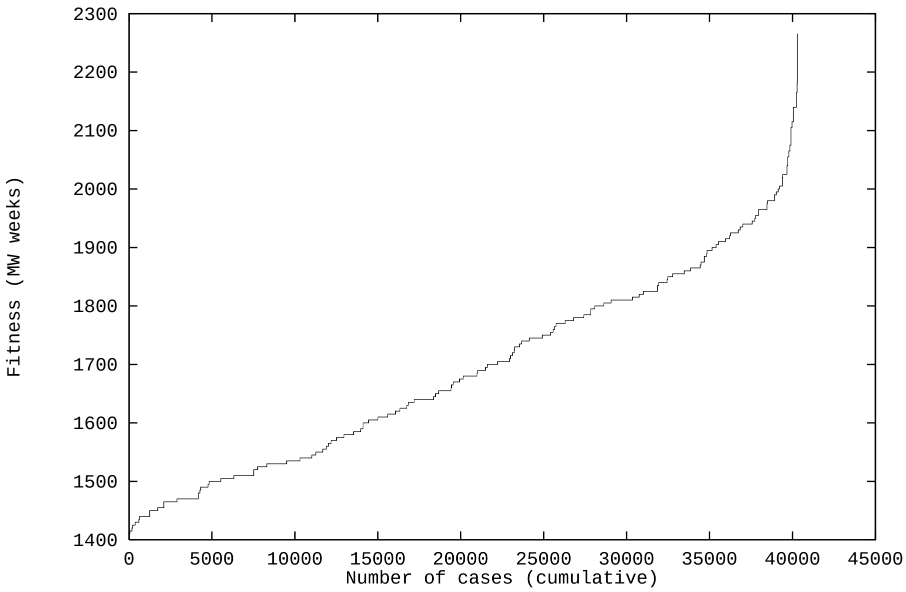

Fig. 6. Cumulative frequency v. fitness (minimum line cost)

Most schedules again concentrate maintenance in the first four weeks but some, including the best (see Table 7), use the fifth and sixth weeks as well. This more distributed maintenance plan is only a marginal improvement on the previous result.

Table 7. Best schedule produced by minimum line cost heuristic

Week	1	2	3	4	5	6	7	8	9
Lines	4	6 8	5 7	3	2	1			
Fitness	1411.67 MW weeks								

Table 8 shows one of the worst schedules. It suggests, with this greedy optimizer, there is merit in delaying consideration of lines 1 and 2. We return to this point in the next section.

Table 8. Worst schedule produced by minimum power flow heuristic

Week	1	2	3	4	5	6	7	8	9
Lines	7	1 2 8	3	4	5	6			
Fitness	2265 MW weeks								

6.4 Heuristic 4 – Minimum Increase in Line Cost

The previous greedy optimizers had been based on the assumption that placing a line in maintenance was bound to increase the power flows on the remaining lines and so must increase line costs (or leave them unchanged). Whilst it is theoretically possible for the change in power flows to decrease line costs it was assumed that this would not occur in practice. As this section shows, this assumption was wrong. It is possible to schedule lines 1 or 2 so that they reduce line costs.

Briefly this can happen when one of the pairs of high rating lines (e.g. 3 and 4) is over loaded but the rest of the network (particularly lines 1 and 2) is not and lines 1 and 2 are transferring power to them. Maintaining line 1 increases the impedance between nodes 1 and 2 and so while the power through line 2 increases, the total power flow between nodes 1 and 2 decreases. As the generation remains unchanged, the power through the over loaded lines decreases by the same amount, so reducing the line costs (the change is met by increased power flows in the rest of the network).

The greedy minimum line cost optimizer was replaced by the more complex least increase in line cost optimizer. This schedules maintenance in the week in which maintaining it would lead to the least increase in line costs (or in which there is most decrease). If there is a tie, the earliest week is used. N.b. this optimizer looks one week ahead whereas the others make their decisions using only the lines that have already been scheduled.

As Fig. 7 shows this algorithm manages to find the optimal solution (1255 MW weeks) and the cost of the worst schedule (1705 MW weeks) is marginally better than the worst found by the least power flow algorithm. Now more than 70% have a fitness of better than 1480 MW weeks. Figure 8 shows the fitness distributions for all four heuristics plotted to the same scale.

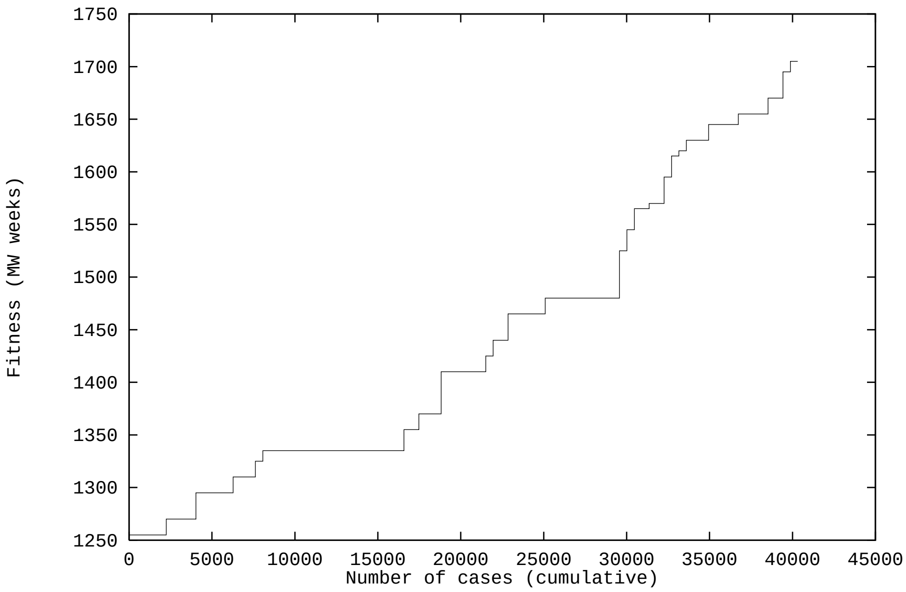

Fig. 7. Cumulative frequency v. fitness (minimum increase in line cost)

Trial solutions now concentrate maintenance in the first three or four weeks except for lines 1 and 2 which are placed in the last two weeks (when demand is heaviest). Table 9 gives an example schedule produced by this method, which is the best known solution to the four node problem.

Table 9. Best schedule (produced by minimum increase in line cost heuristic)

Week	1	2	3	4	5	6	7	8	9
Lines	3	4	5 6 7	8				1	2
Fitness	1255 MW weeks								

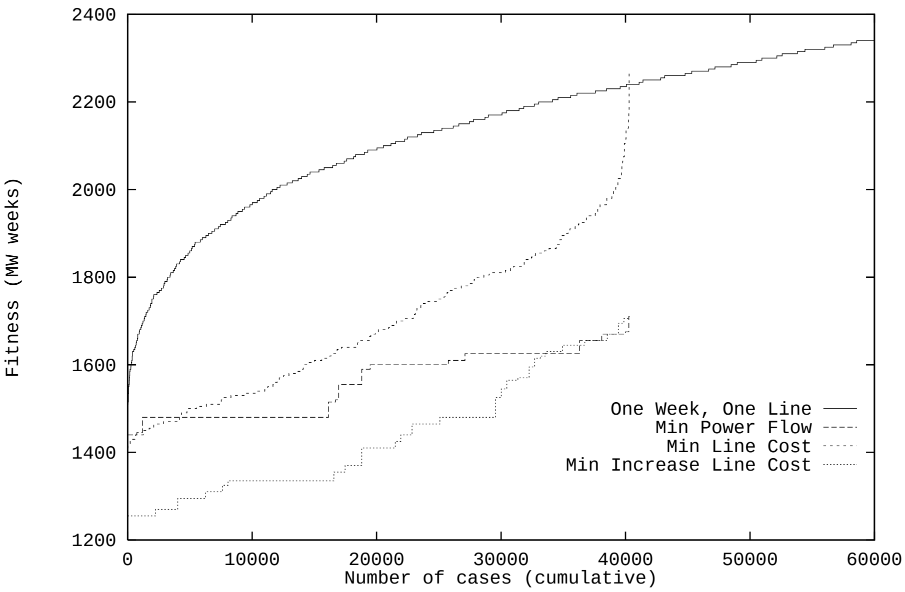

Fig. 8. Cumulative frequency v. fitness (greedy optimizers)

The worst schedules (Table 10) suggest there is merit (with this type of greedy optimizer) of considering lines 5 and 6 last. We are now free to consider line 1 and 2 at any point, since the optimizer will almost certainly place them either in week eight or nine regardless.

Table 10. Worst schedule produced by minimum increase in line cost heuristic

Week	1	2	3	4	5	6	7	8	9
Lines	5 6 7	3	4	8				1	2
Fitness	1705 MW weeks								

7 Using QGAME

Using a chromosome which consisted of a permutation of the lines to be scheduled, single point PMX crossover [Gol89] and a fitness function which used the minimum increase in line costs greedy optimizer (Sect. 6.4) the genetic algorithm QGAME [FT94] had no difficulty in finding the optimal solution. With a population of 20 and default parameters (see Table 11) optimal solutions were found by generation four in a couple of seconds.

By generation 12 approximately half the population consisted of the same optimal solution, however there were a number of slightly different chromosomes which also gave rise to optimal schedules. By generation 20 almost all the population consists of optimal solutions but there are now at least two different but equivalent optimal schedules and there are at least four different but optimal chromosomes. The population now appears to be dynamically stable, with no appreciable change in either chromosomes (genotype) or schedules (phenotype) by generation 100.

Table 11. QGAME genetic algorithm parameters

Name	Value	Meaning
Problem_type	Combinatorial	Combinatorial crossover and mutation operators are used on the chromosome, which holds a permutation of the lines to be scheduled
GeneType	G_INT	Line identification number
POPULATION	20	Size of population
CHROM_LEN	8	Eight genes (lines) per chromosome
OptimiseType	Minimise	Search for lowest fitness value
PCROSS	0.6	On average each generation 0.6×POPULATION/2 pairs of individuals are replaced by two new ones created by crossing them over
PMUT	0.03	On average each generation 0.03×POPULATION individuals are mutated by swapping two genes
Selection	Truncated	The best TRUNCATION×POPULATION of the population are passed directly to the next generation. The remainder are selected using normalised fitness proportionate (roulette wheel) selection. Fitness is linearly rescaled so the best in the population has a normalised fitness of 1.0 and the worst zero
TRUNCATION	0.5	The best 50% passed directly to the next generation
POOLS	1	Single population
CHROMS	1	Individuals use single chromosome
MAX_GEN	100	Run for 100 generations

8 Discussion

Having solved the four node problem, we now consider how this technique will scale up to larger problems. There are two aspects to scaling up; how difficult it is to search the larger search space and how the time taken by fitness evaluation scales.

Table 12 shows the computational complexity of the four heuristics. A large component of them all, is the DC load flow calculation, which assuming the number of nodes $\approx$ no. of lines, is $O(n^3)$. In practice we expect to do rather better than simple interpolation using Table 12 would indicate because:

- Table 12 reflects scaling for large problems, i.e. the time taken by low order terms will fall as a proportion of the total as the problem grows,
- We expect to use sparse matrix and other techniques to speed up the DC load flow calculation,
- The code has not been optimized as yet.

But even so, considerable growth in the fitness evaluation time must be anticipated, as the size of the problem (i.e. the number of nodes) is increased.

Table 12. Computational complexity of fitness function

Heuristic	Order of Computational Complexity	
One week, One Line	weeks$\times$DC	weeks$\times$nodes^3
Min Power Flow	(lines+weeks)$\times$DC	(lines+weeks)$\times$nodes^3
Min Line Cost	(lines+weeks)$\times$DC	(lines+weeks)$\times$nodes^3
Min Increase Line Cost	weeks$\times$lines$\times$DC	weeks$\times$lines$\times$nodes^3

We can also calculate how the size of the search space grows with problem size but assessing how difficult it will be to search is more problematic. The size of the complete search space grows exponentially, being $2^{\text{lines}\times\text{weeks}}$. However this includes many solutions that we are not interested in, e.g. we are not interested in solutions which take a line out of service when all its maintenance has already been done.

If we are prepared to consider only solutions which maintain each line in a single operation for exactly the required time then the number of such solutions is:

$$(1 + \text{weeks} - \text{weeks to do maintenance})^{\text{lines}} \quad . \tag{2}$$

Although a big number this is considerably less than $2^{\text{lines}\times\text{weeks}}$. For the four node problem the ratio between them is $9.1\ 10^{-15}$ and this difference grows exponentially as the problem increases in size.

The greedy optimizers use their problem specific knowledge to further reduce the search space. They consider at most (lines)! points in the smaller search space referred to in (2). Once again this is a considerable reduction. For the four node problem it is a thousandfold reduction and this difference grows rapidly as the problem increases in size. However, as we have seen from the first three greedy optimizers, there is a danger in all such heuristics that by considering some solutions rather than all, the optimum solution may be missed.

For the four node problem it was feasible to investigate a significant part of the search space. Figure 9 shows part of the search space in which schedules do not do more maintenance than required. This includes all $(1+\text{weeks}-\text{weeks to do maintenance})^{\text{lines}}$ schedules which do the required amount of maintenance plus those that fail to maintain one or more lines (above 10,000 MW weeks the graph is shown dotted as detailed statistics were not recorded, values below 10,000 are plotted on an expanded scale in Fig. 10). Figure 10 indicates that it is hard to find the global minimum but it should be straightforward to find solutions costing less than 2500MW weeks. Figure 8 shows all of the greedy optimizers should readily find solutions better than this and the three that use DC load flow calculations would readily find solutions better than 1480MW weeks (i.e. within 225MW weeks of the global optimum).

In addition to reducing the size of the search space, the successful greedy algorithm, considerably enhances the chance of finding the optimal solution because there are many (2240) solutions which produce it. Whilst some (32) of the multiple optimal solutions arise from the symmetry of the original problem others come from the power of the greedy algorithm itself.

There is no guarantee, despite the care in designing the four node problem, that larger problem's fitness landscapes will be similar to that of the four node problem. Assuming they are, then we can expect our greedy scheduler to readily find near optimal solutions.

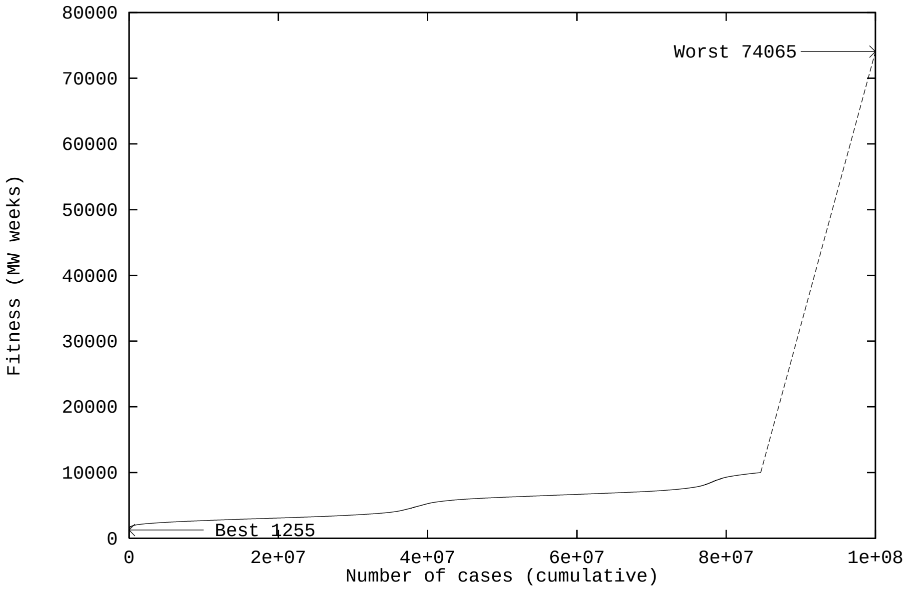

Fig. 9. Cumulative frequency v. fitness (no line maintained more than once)

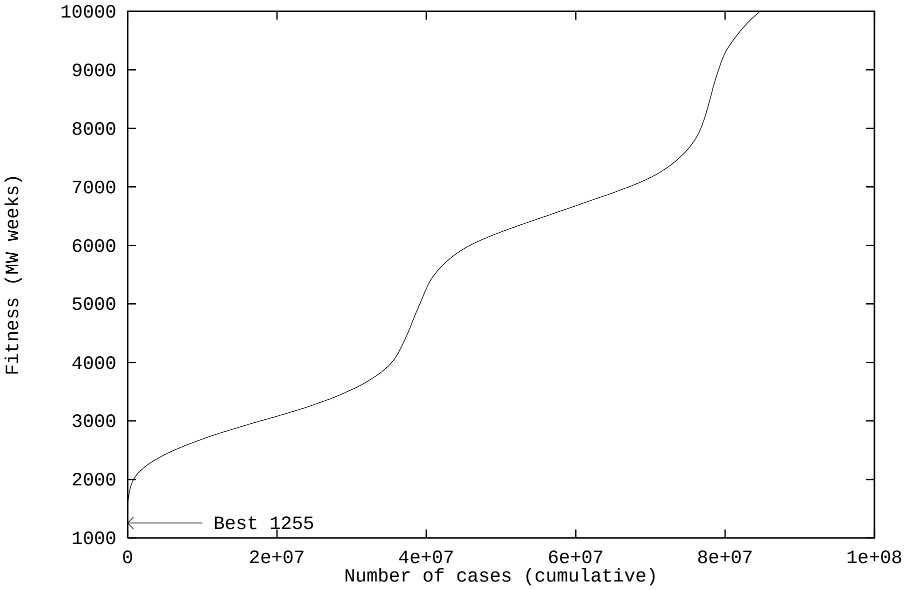

Fig. 10. Cumulative frequency v. fitness (no line maintained more than once)

9 Conclusions

This paper has described the complex real world problem of scheduling preventive maintenance of a very large electricity transmission network. A demonstration problem which, although small, includes many of the features of the real problem, has been created. The approach of hybridising a genetic algorithm (GA) with a problem specific heuristic has been successfully applied to the demonstration problem.

The time taken to perform GA fitness evaluations and with it program run time, grows rapidly with problem size, however there are a number of techniques which could be used to contain this.

Analysis of the search space of the demonstration problem indicates, provided it is representative of larger problems, our hybrid GA approach should be able to find near optimal solutions to larger maintenance scheduling problems. Since the demonstration system was devised to contain features of the larger problem we may expect, provided the growth in fitness evaluation time is not excessive, this approach to be suitable for scheduling the maintenance of electrical power transmission networks.

10 Further Work

We have started applying these techniques to real data for the transmission network in South Wales. This is a significant network consisting of 28 nodes, connected by 40 lines which have to be maintained by a 53 week maintenance plan. Whilst at an early stage, promising results have been obtained using the same parameters as were used on the four node problem.

The run time of these experiments is consistent with Table 12 and the fact that results (albeit preliminary) have been obtained indicates some similarity between the search space and that of the four node problem, despite it being $2.1\ 10^{60}$ fold larger.

We also hope to consider alternative chromosomes, such as expansive coding, genetic programming or those used on some time tabling problems [Ros94]. First establishing their performance on the four node problem.

So far we have not considered the problem of ensuring maintenance does not unduly effect the resilience of the network. This will be left to a later stage.

We intend to reduce the computation cost of the fitness evaluation and greedy optimizer by using "sparse matrix" techniques and possibly incremental evaluation. Further speed ups might be achieved by using approximate rather than exact calculations.

11 Acknowledgments

W. B. Langdon is funded by the EPSRC and The National Grid Company plc.

I would like to thank Mauro Manela and my supervisors (M. Levene and P. C. Treleaven) for their criticisms and ideas, Mike Calviou and Maurice Dunnett for the fitness function and the four node problem, Laura Dekker for assistance with setting up QGAME and the anonymous reviewers for their helpful suggestions.

The four_node problem definition and QGAME are available via anonymous ftp, site `cs.ucl.ac.uk` directory `genetic/four_node`.

References

[ABN94] Laurent Atlan, Jerome Bonnet, and Martine Naillon. Learning distributed reactive strategies by genetic programming for the general job shop problem. In *Proceedings of the 7th annual Florida Artificial Intelligence Research Symposium, Pensacola, Florida, USA.* IEEE Press, 1994.

[And94] David Andre. Automatically defined features: The simultaneous evolution of 2-dimensional feature detectors and an algorithm for using them. In Kenneth E. Kinnear, Jr., editor, *Advances in Genetic Programming*, chapter 23. MIT Press, 1994.

[BBM93] David Beasley, David R. Bull, and Ralph R. Martin. Reducing epistasis in combinatorial problems by expansive coding. In Stephanie Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*, pages 400–407. Morgan Kaufmann, July 1993.

[CH93] Hugh M. Cartwright and Stephen P. Harris. Analysis of the distribution of airborne pollution using genetic algorithms. *Atmospheric Environment*, 27A(12):1783–1791, 1993.

[Dav91] Lawrence Davis, editor. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

[Dun93] R. M. Dunnett. A proposal to use a genetic algorithm for maintenance planning. PSBM note, National Grid, Technology and Science Laboratories, July 1993. Private communication.

[FRC93] Hsiao-Lan Fang, Peter Ross, and Dave Corne. A promising genetic algorithm approach to job-shop scheduling, rescheduling and open-shop scheduling problems. In Stephanie Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*. Morgan Kaufmann, 1993.

[FRC94] Hsiao-Lan Fang, Peter Ross, and Dave Corne. A promising hybrid GA/heuristic approach for open-shop scheduling problems. In A. Cohn, editor, *ECAI 94 Proceedings of the 11th European Conference on Artificial Intelligence*, pages 590–594. John Wiley & Sons, Ltd., 1994.

[FT94] J. L. Ribeiro Filho and P. Treleaven. GAME: A framework for programming genetic algorithms applications. In *Proceedings of the First IEEE Conference on Evolutionary Computing – Proceedings of the 1994 IEEE World Congress on Computational Intelligence*, volume 2, pages 840–845. IEEE Press, 1994. 26th June-2nd July, Orlando, USA.

[Gol89] David E. Goldberg. *Genetic Algorithms in Search Optimization and Machine Learning*. Addison Wesley, 1989.

[Hol92] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, 1992. First Published by University of Michigan Press 1975.

[Koz92] John R. Koza. *Genetic Programming: On the Programming of Computers by Natural Selection*. MIT press, Cambridge, MA, 1992.

[Lev94] David Levine. *A Parallel Genetic Algorithm for the Set Partitioning Problem*. PhD thesis, Illinois Institute of Technology, Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439, USA, May 1994.

[MK93] M. L. Maher and S. Kundu. Adaptive design using a genetic algorithm. In John S. Gero and Fay Sudweeks, editors, *Formal design methods for computer-aided design*, pages 211–228, University of Sydney, NSW, Australia, Jun 1993. Key Center of Design Computing, University of Sydney.

[Ros94] Peter Ross. *About PGA 2.8*, 1994. Available via ftp `ftp.dai.ed.ac.uk` directory `pub/pga-2.8`.

[Sha94] Jane Shaw. References on the application of genetic algorithms to production scheduling, June 1994. Available via anonymous ftp site `cs.ucl.ac.uk` file `genetic/biblio/ga-js-shed-bibliography.txt`.

[SMM+91] T. Starkweather, S. McDaniel, K. Mathias, D. Whitley, and C. Whitley. A comparison of genetic sequencing operators. In Richard K. Belew and Lashon B. Booker, editors, *Proceedings of the fourth international conference on Genetic Algorithms*, pages 69–76, Briarcliff Manor, NY, USA, June 1991. Morgan Kaufmann, San Mateo, California.

[Sys91] Gilbert Syswerda. Schedule optimization using genetic algorithms. In Lawrence Davis, editor, *Handbook of Genetic Algorithms*, pages 332–349.

Van Nostrand Reinhold, New York, 1991.
[VJ93] Christine L. Valenzuela and Antonia J. Jones. Evolutionary divide and conquer (I): novel genetic approach to the TSP. *Evolutionary Computation*, 1(4):313–333, 1993.
[YN92] Takeshi Yamada and Ryohei Nakano. A genetic algorithm applicable to large-scale job-shop problems. In R Manner and B Manderick, editors, *Parallel Problem Solving from Nature 2*, pages 281–290, Brussels, Belgium, 1992. Elsevier Science.