# GENETIC ALGORITHM FOR REGIONAL SURVEILLANCE

**Maria John** **David Panton** **Kevin White**
Centre for Industrial and Applicable Mathematics
University of South Australia,
The Levels Campus, Mawson Lakes Blvd,
Mawson Lakes South Australia 5095.
matmj@linus.levels.unisa.edu.au, d.panton@unisa.edu.au, Kevin.White@unisa.edu.au
+61 8 8302 3343

## Abstract

An optimised flight route is sought for an aircraft conducting aerial surveillance of a given geographical region using synthetic aperture radar. Two models for solving this problem will be discussed; the first utilising Integer Programming and the second, Genetic Algorithms. Model efficiency and solution optimality of the two techniques will be compared to identify conditions under which it is appropriate to use each model.

## 1 INTRODUCTION

The Defence Sciences and Technology Organisation (DSTO) at Salisbury, South Australia, has developed *Ingara*, a technique whereby high resolution images of ground targets can be obtained using a Synthetic Aperture Radar (SAR) device. *Ingara* has obvious military and strategic usefulness as well as the potential for various civilian applications, for example, aerial mapping. Aerial surveillance is conducted over rectangular strips referred to as *swaths*, typically 24 kilometres (km) wide. However if higher resolution images are required, the width of the swath is reduced. The length of each swath can vary, provided it remains within operational limits. Surveillance is accomplished by the aircraft flying at 3048 metres (10,000 feet) above surface level, at a distance of 27 km from the side of the swath, and on a heading parallel to the orientation of the swath. As the aircraft maintains a steady altitude and heading, a radar beam is continuously projected across the width of the swath being tracked. Topological features lying within the swath affect the direction and intensity of the radar echo reflected back to the aircraft, so that an accurate picture of the surface topology of the swath can be constructed. The aircraft commences a mission from a given *starting base* and ends possibly at some other specified *ending base*. The purpose of this paper is to describe the problem of regional surveillance; formulating a mission to cover and scan an entire geographical region using swaths of fixed width and variable length. Typically, a surveillance region will be non-convex, and may contain certain **no-fly-zones** over which an aircraft cannot fly. This problem is dissimilar to the mission planning problem discussed in Panton (1999), since an entire region must be scanned, rather than a set of isolated swaths at designated locations. Regional surveillance incorporates another complication, namely that of ascertaining how to find a suitable cover for the region. Furthermore, the process by which a particular cover is selected is incorporated into the tour efficiency. The complication arises because of the dependence between the selection of an appropriate cover for a region and the optimality of the mission tour. Thus, developing a mission to completely scan a region is an order of magnitude harder than the problem discussed in Panton (1999). There are certain features of providing coverage and surveillance of a given region, namely:

- Complete Coverage: An appropriate selection of swaths must completely cover the region.
- No-Fly-Zones: The geographical region may contain areas which cannot be flown over (no-fly-zones). No-fly-zones are defined by a sequence of up to eight coordinate sets which together define a convex polygon with up to eight edges, as discussed in Panton (1999). The aircraft scanning the region may pass through any corner or fly along any edge of a no-fly-zone.

- Disconnected Regions: Regions that are disconnected in a mission are considered as isolated regions, and are therefore scanned individually.

This paper will discuss issues associated with providing cover for a region, describe an Integer Programming (IP) model and introduce a Genetic Algorithm (GA) used to obtain regional surveillance, and present computational results.

## 2 SWATH AND REGIONAL GRID STRUCTURE

Locating a suitable cover for a region is a matter of selecting a swath set according to certain optimisation criteria and subject to appropriate constraints. The geometry associated with a swath is displayed in Figure 1.

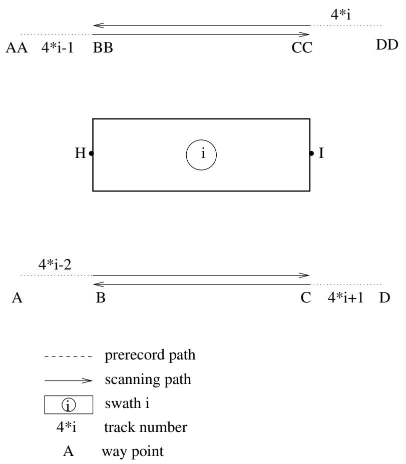


Figure 1: A swath and its associated tracks. Each of the four tracks is indexed by $i$. H and I represent the swath's centre line.

As discussed in Panton (1999), there are a number of definitions and tasks associated with the scanning of a swath, these being

1. The points H and I define the swath centre line. During surveillance the aircraft track is parallel to but displaced by up to 27 km either side of the swath.

2. The swath may be scanned from H to I or I to H, and from either side of the swath. However, only one track may be scanned for any given swath.

3. As a result of items 1 and 2 there are four aircraft tracks along which the swath may be scanned. Figure 1 indicates all four possible flight tracks.

4. Points such as $AA$ or $D$ are referred to as *way-points* on the flight tracks.

5. Prior to surveying a given swath, the aircraft heading must be aligned with the side of the swath from which scanning is to take place. The distance over which alignment is achieved is referred to as the pre-record distance. For example, $A \rightarrow B$ is the pre-record distance for alignment prior to scanning $B \rightarrow C$.

6. Each swath is defined by four sets of nodes ($A, C$; $D, B$; $AA, CC$; $DD, BB$), each pair being a possible candidate in an optimal solution. Note that once the first member in a pair is selected (e.g. $A$), then the second member ($C$) is automatically selected since the aircraft must fly along the track $A \rightarrow C$ in that case.

Dependent on the mission being flown, swaths can be considered to be of fixed width, but variable length and orientation. This variability in length and orientation produces an inordinate number of swaths to select from. In order to reduce the number of swaths to something more manageable it is useful to discretise the length and orientation of a swath, thus producing a finite selection of available swaths. This could be achieved in a multitude of ways. The technique used here is to overlay the region to be scanned with a finite rectilinear grid, the squares of which are denoted as pixels. This is illustrated in Figure 2.

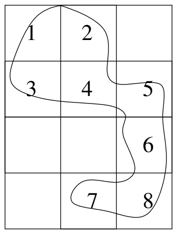


Figure 2: A region and its associated rectilinear grid. Each active pixel is allocated a number.

The size of the pixel is chosen so that the fixed swath width is an integer number of pixels. In the consequent models a swath is one pixel in width. Each swath must now also be an integer number of pixels in length. As the grid is rectilinear there are now only two possible orientations for each swath, either perpendicular or parallel to a given side of the grid. For each swath there still exist four possible scanning tracks.

Using this technique, pixel size determines the total number of swaths (now finite), and the complexity of finding a good cover from the swath set. Grid orientation is initially chosen arbitrarily.

The extent of the grid may well exceed that of the survey region which results in regions of the grid lying outside the bounds of the survey region. In order to differentiate between those pixels that constitute the survey region and those that do not, they will be referred to as active and inactive respectively. A swath may include inactive pixels. Although this represents an area unnecessarily surveyed it may be convenient to do so in order to survey active pixels. At present, the genetic algorithm approach described in section 5 does not incorporate swaths that contain inactive pixels.

## 3 NOTATION

Following is a list of notation to be used in the ensuing model development.

- $s$ - the total number of potential swaths associated with a regional cover,
- $SW_q$ - the set of swaths associated with active pixel $q$ in the defining grid,
- $Y_k$ - a 0-1 variable which is 1 if swath $k$ is used in the cover, and 0 otherwise,
- $n$ - the number of nodes associated with the swath set, including the starting and ending base. The total number of nodes in the resultant network is given by $n = 4s + 2$,
- $S_k, \quad k = 1 \dots s$ - subsets of the node set $N = \{1, 2, \dots, n\}$. For example $S_1 = \{2, 3, 4, 5\}$ and so forth,
- $x_{ij}$ - a 0-1 variable indicating whether or not the edge from $i \to j$ is used in the optimal tour,
- $w_{ij}$ - the length of edge $i \to j$,
- $t$ - the number of active pixels associated with a geographical region,
- $v$ - the number of nodes associated with the active pixel set. The total number of active pixel nodes in the resultant network is given by $v = 8t$,
- $T_l, \quad l = 1 \dots p$ - subsets of the active pixel node set $V = \{1, 2, \dots, v\}$. For example $T_1 = \{1, 2, 3, 4, 5, 6, 7, 8\}$ and so forth.

The Integer Programming model deals with the swath set generated from the grid and active pixels associated with the region. However, the genetic algorithm initially employs the structure and orientation of the active pixels as can be seen in section 5. Thus, in the genetic algorithm approach a region is represented by two node sets, $N$ and $V$.

## 4 INTEGER PROGRAMMING MODEL

We now consider an Integer Programming model (IP-model) as presented in John *et al* (1998) to address the factors discussed in section 1, and simultaneously determine an optimal cover and an associated optimal tour.

Our problem is to minimise the length of the tour

$$\sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} x_{ij}$$

subject to the constraints

$$\begin{aligned}
\sum_{j \neq 1,n} x_{1j} &= 1 && (1)\\
\sum_{i \neq 1,n} x_{in} &= 1 && (2)\\
\sum_{j \in S_k} \sum_{i \notin S_k, i \neq n} x_{ij} &= Y_k \quad \forall k && (3)\\
\sum_{k \in SW_q} Y_k &= 1 \quad \forall q && (4)\\
\sum_{i \notin S_k, i \neq n} x_{ij} - \sum_{l \notin S_k, l \neq 1} x_{jl} &= 0 \quad \forall j \in S_k, \quad \forall k && (5)
\end{aligned}$$

$$\begin{aligned}
&0 \leq x_{ij} \leq 1 \quad \forall i, j, && (6)\\
&x_{ij} \quad integer \quad \forall i, j, && (7)\\
&\text{no subtours allowed} && (8)
\end{aligned}$$

Constraint (1) requires that exactly one edge leaves the starting base (node 1), while (2) requires that exactly one edge enters the ending base (node $n$). Constraint (3) ensures that if swath $k$ is used then exactly one node in its swath set is selected (e.g. node 3 in $S_1 = \{2, 3, 4, 5\}$). Constraint (4) ensures that every active pixel $q$ in the defining grid is covered by exactly one swath. Constraint (5) ensures that once a node in a swath set is selected, that node must connect to a node in another swath set (or node $n$). Constraints (6) and (7) ensure that $x_{ij}$ is a $0-1$ variable. Constraint set (8) ensures that no subtours occur in the optimal solution.

There are several potential versions of the subtour elimination constraints. Most of these possibilities however are not useful, because the number of constraints grows exponentially with the number of swaths. As discussed in John *et al* (1998) we have employed an alternative set of constraints due to Miller-Tucker-Zemlin (M-T-Z) given by:

$$u_i - u_j + s \times x_{ij} \leq s - 1 \quad 2 \leq i \neq j \leq s$$

where $u_i \geq 0$. Since we need only prevent subtours between swaths, a reduction in the number of constraints is possible with an aggregated version of the above constraint set:

$$\sum_{i \in S_l} u_i - \sum_{j \in S_m} u_j + s \times \sum_{i \in S_l, j \in S_m} x_{ij} \leq s-1 \quad 2 \leq l \neq m \leq s$$

The model was implemented using the GAMS (Brooks, 1992) software package with CPLEX (CPLEX, 1998) as the solver.

# 5 USING A GENETIC ALGORITHM FOR REGIONAL SURVEILLANCE

The aforementioned IP model provides a global optimal solution for the surveillance of a geographical region. However, it is necessary to note that as $n$ increases generally the amount of constraint formation and branching also increases, therefore causing large CPU times. Furthermore, genetic algorithms have been applied successfully in finding relatively good solutions to the Travelling Salesman Problem (TSP) which is NP-hard (Davis, 1991; Michalewicz, 1996). One could view the original mission planning problem as a type of TSP with an important difference related to the way in which swaths are to be scanned. The Regional Surveillance Problem incorporates an additional spatial characteristic.

Additionally, the traditional genetic algorithm is considered to be a robust optimisation procedure. In this section we introduce a genetic algorithm (RSGA) to the regional surveillance problem incorporating a simple feasible cover and efficient mission tour. The algorithm is effective and robust and able to handle non-convex regions of varying sizes.

## 5.1 REPRESENTATION OF CANDIDATE SOLUTIONS

The fundamental problem with finding a suitable representation of the candidate solutions (chromosomes) is finding a representation that can be manipulated by genetic operators. In the Regional Surveillance Problem the chromosomes are permutations of the list of active pixels associated with the given region. Each chromosome is of a fixed length ($t$), and is always a feasible solution to the problem. Thus, there are $t!$ possible solutions for a region consisting of $t$ active pixels. An example of a possible chromosome for a region consisting of 8 active pixels ($t = 8$) is

$$[\ 6 \quad 3 \quad 7 \quad 8 \quad 4 \quad 2 \quad 1 \quad 5\ ].$$

Each active pixel consists of eight nodes and has eight alternative paths. An active pixel and its eight alternative paths is illustrated in Figure 3: $a \rightarrow b$ (1), $b \rightarrow a$ (2), $d \rightarrow c$ (3), $c \rightarrow d$ (4), $e \rightarrow h$ (5), $h \rightarrow e$ (6), $f \rightarrow g$ (7), $g \rightarrow f$ (8).

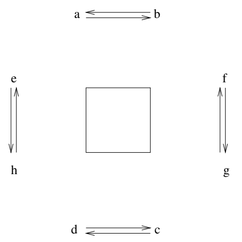


Figure 3: A pixel, its associated nodes (from $V$) and paths.

It is necessary to note that: selecting node $b$ for example automatically leads to the selection of path $b \rightarrow a$; and paths such as $a \rightarrow b$ (1) and $b \rightarrow a$ (2) are considered as two completely independent paths, although their length is identical. Each active pixel in a chromosome succeeds another, with the exception of the first pixel which is automatically connected to the starting base. Any path from the preceding pixel connecting with $a \rightarrow b$ or $b \rightarrow a$ would return two different "costs".

A chromosome is decoded to determine the swaths an aircraft should scan, in order to survey a region. Pixels that share an edge are considered to be adjacent, for example in Figure 2, active pixels 6 and 8. Adjacent active pixels using the same type of path (e.g. two adjacent pixels using path type (1)) are then combined to form a swath, with a chosen track that can be scanned.

A chromosome does not directly represent a cover and mission tour. Converting the ordering of active pixels into a cover and tour is accomplished by the evaluation function described in section 5.2.

The initial population consists of randomly generated, dissimilar chromosomes; alternative permutations of the active pixels associated with the given region.

## 5.2 CHROMOSOME EVALUATION

The evaluation function calculates the "cost" of any chromosome related to the Regional Surveillance Problem. The cost of each chromosome is then employed in the selection function.

Regional Surveillance includes two optimisation criteria that are incorporated into the evaluation function. The entire geographical region must be completely covered, that is each chromosome must contain exactly one of each active pixel from the region. The second optimisation condition is to minimise the mission tour length.

It is evident that searching for the optimal cover and associated tour of a chromosome is very time consuming. The evaluation function used in RSGA implements Dijkstra's Shortest Path Algorithm. Standard Dijkstra's Algorithm is employed, however it is important to note that edge connections only exist for nodes connecting the pixels as listed in the chromosome.

As discussed in section 5.1 a chromosome does not directly represent a cover and tour for the region. Dijkstra's algorithm takes each chromosome and locates the shortest feasible path connecting the pixels from that permutation; using the node set associated with each pixel.

Note that RSGA does not store each chromosome's associated cover and tour, only the final " best " solution to the surveillance of a given region.

### 5.3 SELECTION

To create children using the genetic operators in section 5.4, we require parents to be selected from the present population. The RSGA uses *Roulette Wheel Selection* (Davis, 1991). Each chromosome in the population is allocated a sector of the roulette wheel. The size of the sector for each chromosome is determined by its " cost "; the smaller the cost the larger the size of the sector. In order to select a parent chromosome, the wheel is in effect spun.

### 5.4 GENETIC OPERATORS

Genetic operators are executed on chosen parent chromosomes to construct improved child chromosomes. Applying a genetic operator in the RSGA depends on the probability of crossover parameter ($P_c$). For each generation of the genetic algorithm, a random number ($rn$) is generated, if $rn < P_c$ then crossover is performed, otherwise mutation is performed. A child is evaluated and its " cost " is compared to that of the parent(s) used to produce it. A child replaces a parent if it has a smaller " cost " and is not a duplicate of any chromosome in the current population. This technique of reproduction is known as " Steady-State Without Duplicates " (Davis, 1991).

#### 5.4.1 Mutation

The mutation operator incorporated into the RSGA provides diversity in the population of candidate solutions, in order to attempt to prevent premature convergence. A single parent is selected using the roulette wheel discussed in section 5.3. Two members of the parent chromosome are randomly selected and interchanged producing a new child chromosome as described in Goldberg (1989).

#### 5.4.2 The Recomb Operator

Typically a genetic algorithm's connection with the problem is the evaluation function. However, RSGA incorporates knowledge of the problem into an operator. Experimentation with the IP-model indicates that the optimal solutions start from a particular pixel and continue to link adjacent pixels. The *recomb* operator is created to assist in the production of improved chromosomes, incorporating the adjacent pixel knowledge. The operator forms a contiguous pixel set within a chromosome, which is then preserved in the crossover.

There are two forms of this operator, *recomb without order* and *recomb with order*. The following is a description of the *recomb without order* using chromosome1 from the region illustrated in figure 2 as an example.

1. Randomly generate an element ($ap$), say position 2, from the chromosome, on which to build an adjacent pixel set.

   $$\textbf{chromosome1} = [\ 6\quad 3\quad 7\quad 8\quad 4\quad 2\quad 1\quad 5\ ]$$

2. Search chromosome1 from position $(ap + 1)$, (3), until a pixel that is adjacent to chromosome1 $[ap]$, (3) is allocated. If a pixel is found, (pixel 4 in position 5) then it is swapped with chromosome1 $[ap + 1]$ and $(ap + 1)$ becomes the new $ap$.

   $$\textbf{chromosome1} = [\ 6\quad 3\quad 4\quad 8\quad 7\quad 2\quad 1\quad 5\ ]$$

   This step is repeated until no adjacent pixels to the present chromosome1 $[ap]$ can be found, or the end of the chromosome is reached.

   $$\textbf{chromosome1} = [\ 6\quad 3\quad 4\quad 2\quad 1\quad 8\quad 7\quad 5\ ]$$

If recomb is included in the genetic algorithm it is applied immediately before crossover, on the two selected parents. The initial randomly generated $ap$ is the same for both parents and is considered as crossover point1.

Once step 2 is carried out on both parents, the maximum of the final *ap* from parent1 and the final *ap* from parent2 is considered as crossover point2.

The *recomb with order* consists of the same steps, with one alteration; the pixels must not only be adjacent, but must all lie in the same grid row or column. Consider chromosome1 with initial $ap = 2$. Active pixel 4 is adjacent to 3 and pixel 2 is adjacent to 4, but it does not lie in the same row as pixels 3 and 4. Therefore, pixel 2 is not moved alongside 4, whereas pixel 5 is, since it is both adjacent to 4 and in the same grid row as pixels 3 and 4. Therefore, chromosome1 would become:

$$\textbf{chromosome1} = \left[\begin{array}{cccccccc} 6 & 3 & 4 & 5 & 7 & 2 & 1 & 8 \end{array}\right].$$

There are three forms of the RSGA; RSGA_1 which does not include a recomb operator, RSGA_2 which includes *recomb with order* and RSGA_3 incorporating *recomb without order*.

### 5.4.3 OX Crossover

The OX crossover operator as described in Michalewicz (1996) constructs two child chromosomes from two parent chromosomes selected using the roulette wheel. In RSGA_2 and 3 the parents are initially manipulated by the recomb operator incorporated in the algorithm. A section of a chromosome from one parent is selected and replicated in the child chromosome (step 1), while the relative order of active pixels from the other parent is preserved (step 2). In RSGA_1 the section is randomly chosen. However, in RSGA_2 and 3, the section is selected according to the two crossover points established in the relevant recomb operator.

Consider the following two parents from the region illustrated in Figure 2, with $t = 8$. Both parents have been manipulated by "recomb without order".

$$\textbf{parent1} = \left[\begin{array}{cccccccc} 6 & 3 & 4 & 2 & 1 & 8 & 7 & 5 \end{array}\right]$$

$$\textbf{parent2} = \left[\begin{array}{cccccccc} 1 & 4 & 5 & 6 & 8 & 3 & 7 & 2 \end{array}\right]$$

Using the two crossover points selected in "recomb without order" (i.e. 2 and 6) and step 1, the chosen section of parent1 and 2 is replicated in child1 and child2 respectively.

$$\textbf{child1} = \left[\begin{array}{cccccccc} o & 3 & 4 & 2 & 1 & 8 & o & o \end{array}\right]$$

$$\textbf{child2} = \left[\begin{array}{cccccccc} o & 4 & 5 & 6 & 8 & 3 & o & o \end{array}\right]$$

Applying step 2 completes the crossover operation. Beginning from the second crossover point, the order of the active pixels (that do not already exist in child1(2)) in parent2(1) are replicated in child1(2). Note that the first element of the chromosome follows the last.

$$\textbf{child1} = \left[\begin{array}{cccccccc} 6 & 3 & 4 & 2 & 1 & 8 & 7 & 5 \end{array}\right]$$

$$\textbf{child2} = \left[\begin{array}{cccccccc} 1 & 4 & 5 & 6 & 8 & 3 & 7 & 2 \end{array}\right]$$

## 6 RESULTS

A number of experiments were conducted on a diverse range of regions of varying sizes; using the IP-model and the three forms of the RSGA. Experimental regions and their corresponding data were generated to be consistent with aircraft ranges. A final cover and tour for a region is known as a mission.

Summarising the results of the three RSGA's applied to various regions, we found that RSGA_2 produced more efficient missions than RSGA_1 and 3. Mission efficiency is measured by the total distance the aircraft travels in the final mission and the amount of CPU time the algorithm requires to produce the mission.

Consider a region with a 5 by 4 pixel grid; region1, illustrated in Figure 4. The region consists of 14 active pixels. All missions associated with this region begin and end at the same base.

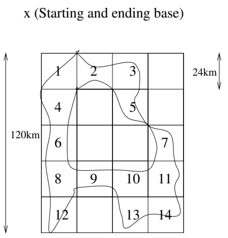


Figure 4: Region1, its associated 120km by 96km rectilinear grid, 14 active pixels and starting and ending base.

The IP and RSGA models were implemented on region1 to investigate their efficiency. All the RSGA models use a crossover probability of 0.953 and maintain a population of 30 chromosomes. These parameters were selected by experimentation.

Each execution of the three RSGA's employed an allocated total number of generations (gen). For every set number of generations, each RSGA was executed five times with different random seeds. The mean distance and CPU time over the five executions were recorded as the final mission distance and required CPU time respectively, for each generation. Table 1 exhibits the performance of the genetic algorithms and the IP-model on region1. The two columns displayed represent the final mission distance and the required CPU time.

Table 1: Performance comparison of RSGA_1, 2 and 3 with different generations (results averaged over five executions) and the IP-model.

| **MODEL** | **DISTANCE** (km) | **CPU TIME** (hrs:mins:secs) |
|---|---|---|
| IP-Model | **2870.502** | **2:34:24.9** |
| RSGA_1: | | |
| gen=100 | 3801.90 | 0:0:27.22 |
| gen=200 | 3661.62 | 0:0:54.50 |
| gen=500 | 3426.67 | 0:2:16.92 |
| gen=1000 | 3236.26 | 0:4:41.42 |
| RSGA_2: | | |
| gen=100 | 3510.06 | 0:0:30.70 |
| gen=200 | 3244.50 | 0:1:03.36 |
| gen=500 | 3059.06 | 0:2:43.26 |
| gen=1000 | **2951.16** | **0:5:30.20** |
| RSGA_3: | | |
| gen=100 | 3608.18 | 0:0:29.80 |
| gen=200 | 3378.46 | 0:1:03.80 |
| gen=500 | 3179.61 | 0:2:33.84 |
| gen=1000 | 3100.53 | 0:5:45.22 |

The results demonstrate that for each RSGA the distance of the final mission decreases gradually as the number of generations increase. Additionally, it can be seen that the placement of adjacent pixels using RSGA_2, produces a smaller mission distance and employs less CPU time than RSGA_3. This, indicates the interdependence between the placement of adjacent pixels that also belong to the same grid row or column, and the quality of the mission.

RSGA_2 and 3 force adjacent pixel sets to form within chromosomes from the first execution of the crossover operator, and then continue as part of the crossover to preserve these sets. This enables the genetic algorithm to produce a shorter mission in less CPU time. RSGA_1 which incorporates the OX crossover alone takes more generations to compile and preserve adjacent pixel sets. Consequently, RSGA_1 produces longer final missions using more CPU time.

Although the IP-model allocates the optimal mission for a region, the CPU time required to obtain "good" suboptimal missions from the genetic algorithms is considerably less, which is an operational advantage.

## 7 CONCLUSION

In this paper we have described and analysed three variations of a genetic algorithm for the Regional Surveillance Problem (RSP). The objective was to introduce a genetic algorithm to the RSP and to investigate its performance. The major problem was to establish a representation for the problem that operators (existing and future) could manipulate. The results indicated that the genetic algorithms produced "good" suboptimal tours for the problem within reasonable time. Although the magnitude of the experimental region described in section 6 is of realistic proportions (around 100km x 100km), the number of active pixels can increase. Results from region1 and additional experimentation on simulated regions; consisting of a larger number of active pixels, indicated that the genetic algorithms are operationally superior to the IP-model. Future work will include executing the genetic algorithms for a set number of generations greater than 1000, and developing and comparing a simulated annealing model to the genetic algorithms.


**References**

L. Davis (1991). *Handbook of Genetic Algorithms.* Van Nostrand Reinhold.

D.E. Goldberg (1989). *Genetic Algorithms in Search, Optimisation, and Machine Learning.* Addison-Wesley Publishing Company, Inc.

J. H. Holland (1992). *Adaptation in Natural and Artificial Systems.* Massachusetts Institute of Technology Press Edition.

M. John, D.M. Panton, and K. White (1998). Models For Regional Surveillance. *Proceedings of the International Conference of Optimisation Techniques and Applications, pp 819-826.* Curtin University of Technology.

Z. Michalewicz (1996). *Genetic Algorithms + Data Structures = Evolution Programs.* Springer.

D.M. Panton, and A.W. Elbers (1999). Mission Planning For Synthetic Aperture Radar Surveillance. *Interfaces* (to appear).

GAMS a user's guide release 2.25: Brooks, Kendrick, Meeraus, Boyd and Fraser 1992.

CPLEX, CPLEX Optimisation, Inc. 1998.