Two Agent Cooperative Search Using Game Models with Endurance Time Constraints

Sujit P B, Debasish Ghose

Two Agent Cooperative Search Using Game Models with Endurance Time Constraints

Journal:	*Engineering Optimization*
Manuscript ID:	GENO-2009-0102.R4
Manuscript Type:	Original Article
Date Submitted by the Author:	15-Sep-2009
Complete List of Authors:	P B, Sujit; University of Porto Ghose, Debasish; Indian Institute of Science, Aerospace Engineering
Keywords:	unmanned aerial vehicles, decision-making, route planning, game theory
Note: The following files were submitted by the author for peer review, but cannot be converted to PDF. You must view these files (e.g. movies) online.	
jrnlBaseGeno.tex	

Engineering Optimization
Vol. 00, No. 00, Month 200x, 1–28

RESEARCH ARTICLE

Two Agent Cooperative Search Using Game Models with Endurance Time Constraints

P.B. Sujit and Debasish Ghose
(*Received 00 Month 200x; final version received 00 Month 200x*)

In this paper, the problem of two Unmanned Aerial Vehicles (UAVs) cooperatively searching an unknown region is addressed. The search region is discretized into hexagonal cells and each cell is assumed to possess an uncertainty value that represents the information available in that cell. The UAVs have to cooperatively search these cells at every time step and reduce the uncertainty of the entire region as quickly as possible. The UAVs are assumed to have limited endurance, limited sensor and communication ranges. Due to limited endurance time, the UAVs need to return to the base station for refueling. When multiple base stations are present, selecting a base station for refueling is also an issue that needs to be addressed. Designing search strategies with these constraints is a difficult problem. In this paper, a route planning algorithm is proposed that takes endurance time constraints into account and uses game theoretical strategies to reduce the uncertainty. The route planning algorithm selects only those cells that ensure that the agent will return to any one of the available bases. Using these cells, the agents form paths and provide these paths to the game theoretical strategies to choose one of them. The game theoretical strategies take the presence of other agent into account and selects a path that will yield high uncertainty reduction. Various strategies that can be explored for the agents to enhance search effectiveness are non-cooperative Nash, cooperative and security strategies. Monte-Carlo simulations are carried out for all the search strategies and compared with greedy strategy with one, two and four steps look ahead step length paths. The results show that the game theoretical strategies outperform the greedy strategy. Within the game theoretical strategies noncooperative Nash strategy performs as well as the cooperative strategy in an ideal case. But when information about the uncertainty maps is different then noncooperative Nash performs better than the cooperative strategy. The security strategy performs the worse of the three

P.B. Sujit is a Research Scientist in the Department of Electrical and Computer Engineering, University of Porto, Portugal. India. Email: sujit@fe.up.pt
Debasish Ghose is a Professor in the Department of Aerospace Engineering, Indian Institute of Science, Bangalore 560 012, India. Email: dghose@aero.iisc.ernet.in.
This work was partially supported by the DRDO-IISc Programme in Mathematical Engineering.

ISSN: 0305-215X print/ISSN 1029-0273 online

DOI: 10.1080/0305215YYxxxxxxx
http://www.informaworld.com

strategies but is still better than the greedy strategy. With increase in the look ahead policy length the computational time increases, therefore, a heuristic based on partitioning the search space into sectors is developed. The simulation results show that the performance of game theoretical strategies with sector partitioning performs almost as well to that of without partitioning but with significant decrease in computational time.

Keywords: game theory, cooperative search, route planning.

1. Introduction

1.1. *Problem Motivation and Preliminaries*

Recently, considerable effort has been expended in developing technologies that would enable search and surveillance operations to be carried out in hostile and inaccessible environments using unmanned aerial vehicles (UAVs) and robots. The UAVs/robots/agents are equipped with sensors that collect necessary information as they pass through a region. An important requirement for these agents is to have the ability to make optimal decisions independently based on their sensor information and have minimal communication among themselves. The problem of decision making becomes complex when more than one agent is deployed to perform search operation. The search of an unknown region can be more effective when the agents can cooperate with each other. The cooperation can be either implicit, by considering the possible actions of the other agents without actual communication between them, or it can be explicit, where communication among the agents is used to achieve cooperation.

Usually the agents used for these missions have limited endurance time due to limited fuel carrying capacity or battery power. The agents need to return to the base for refuelling and also to download the information they have collected during the sortie. The agents make online route decisions based on local information sensed by them and by considering the presence of other agents in the neighbourhood. During the exploration process the searcher may not know the way back to the base. In such cases, one simple strategy would be to back track along the route which the agent has explored, but this route would not be optimal as the uncertainty on that path has been reduced during its exploration. Hence, the route used by the agents to get back to the base must also be optimal for the entire sortie to be optimal in terms of search effectiveness. To compute the optimal search route back to the base some restrictions on the paths selected by the agents need to be applied. If the number of bases stations available are more than one then the agents have to select the base station dynamically.

These types of problems, where a search mission with constraints on the endurance time and having a requirement that the agents have to return back to a base station for refuelling, are realistic and are not addressed in the search literature Benkoski *et al.* (1991), Stone (1975). This is the first paper that addresses a problem of conducting sorties with (i) limited endurance time (ii) return back to base station, and (iii) dynamic selection of base station by agents for refuelling. The only other paper that addresses the notion of generating sorties with limited endurance time is by Sujit and Ghose (2004-AES). However, the k-shortest path approach used in this paper does not address the problem of changing the base station dynamically. The source and destination base stations have to be pre-determined before the start of the sortie.

In this paper, a framework for obtaining optimal search strategies and dynamic selection of base stations for two agents operating in an unknown region with limited endurance time is presented. The framework is based on principles from game theory, namely, noncooperative Nash, security and cooperative strategies. Each agent is modeled as a player and the possible actions of the agent as strategies. The noncooperative Nash and security strategies do not require or allow any kind of communication between the agents, while cooperative strategy requires communication among the agents. An algorithm that yields an optimal route for two agents with the combination of restricted path selection is presented. Preliminary work on developing optimal search strategies using game theory was reported in Sujit and Ghose (2004-ACC).

Although in a multiple-agent search scenario there could be many agents, the results presented in this paper are restricted to two agents, both of which use game theoretical strategies for route selection. The results give crucial insight into the use of game theoretical strategies to search problems. These results can be extended to multiple agent scenarios but with increased computational burden. Hence, this is beyond the scope of the present paper.

1.2. *Motivation for using a Non-cooperative Game Paradigm*

One of the motivations of this work arises from the fact that game theoretical framework of modelling multiple decision-making situations give rise to two different solution concepts: one based on cooperation between players and the other based upon non-cooperation. Application of these notions to the economic market place had to take into account the fact that players are not inherently altruistic, thus making the cooperative framework somewhat untenable, unless the cooperation is enforced by a third party. On the other hand, the non-cooperative framework has shown that in repeated games, cooperation automatically emerges as the best non-cooperative solution and hence the notion of cooperation is inherent and enforceable in the non-cooperative framework. When cooperation between automated agents that are devoid of any selfish motive and have only a common goal in mind, it is more logical to use a cooperative framework. In this paper, it is observed that the non-cooperative framework is almost equally effective and is no more computationally time consuming than the cooperative framework.

There are other reasons too, related to the specific problem structure, which justifies the usage of the non-cooperative framework. For instance, when the sensor performance is unreliable or noisy, or due to ineffective communication the uncertainty map of each agent changes with time unknowingly to the other agents, leading to different uncertainty maps for different agents. In such situations, the cooperative decision making mechanism breaks down resulting in non-cooperative Nash strategies performing better than the cooperative strategies.

1.3. *Organization*

The remainder of the paper is organized as follow: A survey of the recent literature on cooperative search is given in Section II. In Section III, the search space, objectives and constraints for the problem is presented. Section IV describes the algorithm required for the agents to return back to base for two agents with one base station and the extension of the algorithm for multiple bases in the search space. Section V describes the formulation of game theoretical framework and various strategies. Finally, Section VI presents and discusses the simulation results on the performance of various game theoretical strategies

with variable length look ahead policies.

2. Related Work

Searching an unknown region has been a topic of interest for the past few decades. However, the main focus has been to develop decentralized algorithms for maximizing the detection of targets, optimal trajectory generation and exploring the search region efficiently.

For a given source and destination points one can design an optimal path planning algorithm. Nikolas and Brintaki (2005) present an offline path planner using differential evolution technique for a given pair of target and source locations taking obstacle avoidance between obstacles and inter-UAVs into account. Shanmugavel *et al.* (2006) developed a 3D dubin paths based algorithm for multiple UAV rendezvous. Geiger *et al.* (2006) present a non-linear programming based optimal path planner to provide maximum coverage on a target using multiple UAVs. Bortoff (2000) designed an optimal path planning algorithm in the presence of threats by initially constructing a voronoi diagram and then using virtual potential field to generate the optimal path. Ryan and Hedrick (2005) developed a path planner that takes kinematic constraints into account while following a path generated by a helicopter that can carry out steep turns for a search and rescue mission. Wong *et al.* (2004) presents a circular arc based path planning technique to tour a set of predefined targets. In the above cited paper the authors assume that sufficient fuel is present during the mission which is a strong assumption as vehicles do carry limited fuel and have to return to the base station for refueling. In this paper, the problem of route planning is posed by taking this endurance time and the need for refueling into account.

Coordinating these multiple vehicles to achieve effective search with minimal search effort is a complicated task. Cooperative control of UAVs was initially addressed by Passino *et al.* (2000), Polycarpou *et al.* (2001), in which the behavior of a team of UAVs were studied under various operational conditions, strategies, and task goals. A recursive approach for cooperative search using a multi-objective cost function, and q-step path planning was developed. Yang *et al.* (2002-GNC) propose a decentralized cooperative strategy where cooperation is achieved by having each agent take into account the other agent's possible actions. A team of agents are deployed into an environment which is cleaved into square cells. Each cell has an associated uncertainty value, and also a probability of finding a target in the cell. Every agent has a feed-forward neural network trained with reinforcement learning that predicts a reward after two look ahead steps. The path that provides the best reward is selected. The paper assumes complete communication among the agents and the objective is to have effective search and maximize the detection of targets in the given region within a specified time. Some analysis for the upper limit on search time to search fully each unknown cell in the environment with the same framework is presented in Yang *et al.* (2002-CDC). Coordinating multiple UAVs for simultaneous target intercept is described in Beard *et al.* (2002), where the authors present a heuristic approach for target assignment of aerial vehicles and then design paths through hostile environment based on Voronoi diagrams. The paths are subjected to minimizing the threat exposure and length costs with the objective that all the aerial vehicles should intercept their assigned targets simultaneously. In Beard and McLain (2003), a team of UAVs is given the task of searching a region with unknown opportunities and hazards. The regions of opportunities and hazards are identified using

the q-step look ahead policy. The objective of the team is to maximize the regions of opportunity visited by the team, while minimizing visits to regions of hazard, subject to constraints that the UAVs must remain connected by a communication network at all times and avoid collisions among themselves. All paths of length q are computed on a longitudinal front and a dynamic programming approach is used to evaluate the best rewardable path. Some heuristics to reduce the computational complexity considering the constraints on the paths are presented. In Mot *et al.* (2000), the authors present an algorithm for coordinated path planning for two vehicles acquiring a target. The terrain is divided into regions and each of these region is associated with a payoff. The terrain traversal is mapped into a graph traversal problem with a node on the graph denoting a region. The decision to move along the graph is decided using a two step look ahead policy.

Exploring unknown regions for seeking information has also been a topic of interest for researchers in robotics. In Goldsmith and Robinett (1998), a problem of exploring an unknown environment using multiple robots is posed and solved using the concept of target points being assigned to each individual robot to maximize search effectiveness. In Batalin and Sukhatme (2002), the authors address the problem of deploying a mobile sensor network into an environment with the task of maximizing sensor coverage of the environment. Two algorithms are proposed to perform the coverage task successfully using only local sensing and local interaction information between robots. In Spires and Goldsmith (1998), space-filling curves are used to define open-loop search trajectories for a team of robots carrying out an exhaustive geographical search operation. In Burgard *et al.* (2002), several mobile robots are assumed to search for targets in a search region. The paper addresses the problem of communication between two groups of robots for effective coordination between them.

Some researchers have considered path constraint in their design for search strategies, but have not studied the possibility of the agents returning back to base for refuelling. For instance, in Nygard *et al.* (2001) and Scerri *et al.* (2005), the authors address the endurance time constraint for wide area search munitions which will either destroy the target within the fuel limit or self destruct.

3. Problem Formulation

3.1. *Scenario*

Consider a two dimensional search space with some *a priori* knowledge of the area represented in the form of an uncertainty map as shown in Figure 1. A search mission consists of multiple agents performing search operation at the same time. The multiple agents perform a number of sorites in the search space. A sortie is a search route that starts from the base station (B) and terminates or returns back to base at the end of its endurance time for refuelling. After refuelling the agents start another sortie. Figure 1 shows typical sorties for an agent with a single base station. Intuitively, an effective search route would have an agent spend more time in the high uncertainty region and less time in the low uncertainty region. The agents have to take decisions autonomously while considering the presence of other agents in the search region.

3.2. *Discretization of the Search Region*

A two-dimensional search region is partitioned into a collection of identical regular hexagonal cells. The reason for using a hexagon as the basic unit is that it offers the flexibility to the searcher to move in six uniformly distributed directions at each time step and reach a neighboring cell while expending the same amount of energy (see Figure 2). This discretized search space model is the same as that used in Sujit and Ghose (2004-AES). Assume that the cells are large compared to the sensor range and the minimum turning radius of the UAV. Thus, the agent spends an unit time searching the cell. One way to interpret this model is to consider the cells that constitute a search route for an agent as waypoints that are used by the agent as a reference to determine its flight path and to determine in which region its search effort should be expended. The exact search pattern can be spiral or lawn mowing paths. For example, if a route is given as a sequence of cells, then the agent uses this as a command to visit the region represented by each of these cells in the same sequence, design the paths and use its sensor to search each cell through the paths, and then continue on to the next cell in the sequence.

3.3. *Uncertainty Map*

The uncertainty map constitutes real numbers between 0 and 1 associated with each cell in the search space. These numbers represent the uncertainty about the location of the target in that cell. An interpretation of the uncertainty map would be as follows: An uncertainty value of 0.6 would imply that any statement about the target's location in cell i (say) would be true only with probability 0.4. An uncertainty value of 0 would imply that everything is known about the cell (that is, one can say with certainty whether a target is located in that cell or not). On the other hand, an uncertainty value of 1 would imply that nothing can be said about the location of the target in that cell. Another interpretation of the uncertainty value is that it is the undetected mass in the cell and represents the extent of the lack of information about that cell. Hence, a successful search operation is one that manages to visit those cells that have large uncertainty values. This model of uncertainty map representation of the search space is similar to the one used in Sujit and Ghose (2004-AES).

Once a cell C_i is visited by a UAV at time t then the uncertainty value of the cell U_i reduces to

$$U_i(t+1) = U_i(t)(1-\beta) \tag{1}$$

where, $\beta \in [0, 1)$ is the uncertainty reduction factor associated with each visit to a cell by a UAV. This factor β has a similar effect as the detection function used in search theory Stone (1975), where the detection function represents the probability that a search in a given cell for a specified duration of time will detect the target provided that the target is present in that cell. An exponential detection function, which is normally used in search problems, is represented as $1 - e^{-\alpha t}$ where t is the time spent in the cell and α is a scaling factor, which is also known as the detection rate. It has the property of diminishing returns in the sense that each incremental time spent in searching a cell produces a decreasing return on the probability of detection. Note that β in (1) has a similar effect in the sense that the incremental reduction in uncertainty reduces with each subsequent visit.

3.4. *Searcher Objectives and Constraints*

The multiple agents used for search missions generally have limited fuel or battery power that allows them to search for a limited length of trajectory per sortie and then return back to base for refuelling or recharging. After refuelling, the agents perform the next sortie of the same length. The number of such sorties are not necessarily specified. However, the sorties can be stopped once the average uncertainty of the region that the agents can reach is below a certain threshold. During each sortie, the agents pass through the cluster of cells and collect information about the cells. The objective of these agents is to maximize the uncertainty reduction per sortie with a constraint on the length of the trajectory.

The search space is composed of identical sized cells, hence the energy spent by an agent in moving from one cell to another is equivalent to one unit step length. Therefore, the fuel limit for an agent can be interpreted as a maximum of N steps. The agent has to return back to the base at the end of the N^{th} step for refuelling. At every time step, the agent can move from one cell to the neighboring cell or the searcher may devote more search effort in the same cell, because of higher uncertainty value in the cell. This would mean that the agent may spend more time, in terms of multiple steps, in this cell.

It is assumed that each agent is equipped with a sensor through which it collects data or information about the cell it visits. So, an agent that spends a certain number of steps in searching any given cell is essentially using this time to collect data about the cell and thus the uncertainty in that cell reduces as a function of the time that the agent spends there.

Assume that at any given time t the agents know the position and route (up to time t) of the other agents. So, at a given time t, the searcher objective is to determine its future route based on its own perceived uncertainty map. The problem can be looked upon as a centralized one if the searchers are assumed to communicate with each other and decide upon a globally beneficial decision. This would be the cooperative solution. In the absence of any such cooperation each searcher has to decide its next search route using some other strategy.

3.5. *Uncertainty Map Dynamics: The Ideal Case*

In an ideal case, every agent A_i has information about the current location of all the other agents and also their uncertainty reduction factors. Each agent starts with the same initial uncertainty map and updates it after every time step using information about the route taken by the other agents till that time. In which case, at any given time, the agents have the same uncertainty map and also know the past route and present location of the other agents. The update is also assumed to be synchronous. Although each agent is aware of the current position, and hence the past route, of the other agents, they do not communicate with each other and thus cannot convey or decide upon their actions in a coordinated fashion.

One possible scenario, where this assumption would be valid, is the one shown in Figure 3, where several agents (UAVs) are searching an unknown region and are tracked by a satellite. The satellite broadcasts the track information of each agent to all the agents. The agents use this information to update their uncertainty map as each agent knows its own past route as well as the other agents' past routes. Since all the information relevant to compute the current uncertainty map is available to all the agents, their uncertainty maps are also the same. There is no direct communication among the agents and neither is

there any communication of information from the agents to the satellite. The information flow is from the satellite to the agents only. Alternatively, one can consider a scenario where each agent broadcasts to the others its current position at every decision step.

3.6. *Uncertainty Map Dynamics: The Non-ideal Case*

In a non-ideal case, it is possible that an agent's perceived uncertainty map may differ from that of the other agents. This can happen due to various reasons. The updating of the uncertainty map, in the ideal case, depends on the following factors: (i) Same initial uncertainty map (ii) Perfect and complete route information of the agents (iii) Perfect knowledge of the uncertainty reduction factors β of all the agents by each agent, and (iv) Perfect synchronization among agents in updating the uncertainty map. In a perfect world all these are feasible assumptions. In fact, if all these are indeed true, then the uncertainty map for all the agents would be the same, and a pre-determined protocol (which in itself may be rather complicated) can, in principle, be devised in order to enable the agent collective to produce a well-coordinated cooperative decision. In this case, there is no need to look for non-cooperative strategies.

However, consider a scenario where the uncertainty map may not be the same for all the agents due to violations of any one or more of the above factors. The most common reasons could be change in the uncertainty reduction factor of an agent due to change in performance of the sensor systems of an agent, which only the agent itself is aware of. This change in performance can happen because of various reasons such as reduced visibility, deterioration of sensor hardware performance, unexpected terrain features that hamper search, etc. Another reason for the uncertainty map to differ could be because of the detection of some features in a cell that increases or decreases the demand of search effort for that cell and is reflected in enhancing or decreasing the uncertainty value of the cell by the searcher. This information is not conveyed to the other searchers due to lack of communication between agents. Noisy broadcast, faulty communication, asynchronous updates, lack of communication with distant neighbors, etc., also contribute to the uncertainty map being different for different agents.

It is this non-ideal scenario that prompts us to explore the possibility of using non-cooperative strategies, since a pre-determined protocol for cooperation needs the underlying uncertainty map to be the same. When the perceived uncertainty maps are not the same, an agent cannot be sure of the actions of the other agents and it is more logical to treat them as (unintentional) adversaries rather than as cooperating team members.The neighboring cell chosen by an agent to move from step t to $t+1$ has to be such that the agent can return back to base from that cell. The procedure for selecting the cells which guarantee safe return to a base are discussed in the following section.

4. Route Planning

The agents have to compute the route depending on the number of available bases. If the search space contains only one base then the mission starts and ends at the same base station. This kind of routes are also known as sorties. But, if there are multiple base stations then the agent may choose any of the base stations. Initially, the route planning is considered for a single base station and then the concept is extended to multiple base stations.

4.1. *Single Base Station*

Consider a single base station located at cell C^b and let the duration of the search be N time steps. The time steps represents the fuel capacity of an agent. An agent starts the search from the base and at the N^{th} step has to return to the base cell. Thus, at every time step t, the agent needs to constrain its next step to only those cells that allow it to return to the base in $N-t-1$ steps. To find these cells a virtual graph of the uncertainty map has to be created.

Virtual Graph: A graph $G = (V, E)$ is created on the uncertainty map using the centers of the hexagonal cells. Consider every center of a cell (C^i) to be a node V_i. Each node V_i is connected to its neighboring cell centers (nodes) $V_j, V_j \in \mathcal{N}(V_i)$ by edges (V_i, V_j) which are assigned unit weights $(C_{ij} = 1)$. The virtual graph G does not contain any loops. A virtual graph of a cell and its neighbours is shown in Figure 4(a) with unity cost for all the arcs $(C_{ij} = 1)$. The dotted lines show the extension of the edges to other cells in the graph.

Cell Selection: Assume that an agent A_i is present at cell C_{s_i} at time t. The next step for agent A_i should be such that choosing that cell the agent should be able to return to a base. Consider $\hat{\mathcal{C}}$ to be a set of all the cells that are within a depth of q steps. For example, if $q = 1$, then the set $\hat{\mathcal{C}}$ consists of cell C^b and its neighboring cells $\mathcal{N}(C^b)$. Now consider the base station C^b to be the source node and a cell $C^k \in \hat{\mathcal{C}}$ to be the destination node. Then, the shortest path between these two nodes can be determined using Dijkstra's shortest path algorithm (Dijkstra (1959)). The algorithm is described in the appendix.

The shortest path gives the least number of steps required to traverse from the source node (base cell) to the destination node C^k, since every arc is equivalent to a unit step length. Let $\bar{\mathcal{C}}$ be a set, which contains cells C^k that satisfy the following condition:

$$|SP(C^b, C^k)| \leq N - (m+1), \quad \forall C^k \in \hat{\mathcal{C}} \tag{2}$$

where $|SP(C^b, C^k)|$ represents the minimum number of steps given by Dijkstra's algorithm with source cell C^b and destination cell C^k, N the number of steps for a sortie, and m the current time step of the searcher. Using these restricted cells $C^k \in \bar{\mathcal{C}}$, paths of q step length are generated. If the cells that do not satisfy (2) are selected and suppose the searcher chooses those cells then the searcher will not be able to return to the base by the end of the N^{th} step.

Algorithm 1 Return to Base Algorithm (Single base)

```
Function: return_base(x, base, q_neighbouring_cells, q, time_left)
source = base; destinations = q_neighbouring_cells;
graph = virtual_graph(cells) /* virtual graph with all cells */
safe_destinations = [ ] /* initialize */
for i = 1 : no_of_destinations do
    min_distance_from_base = shortestPath(base, destinations(i), graph)
    if (min_distance_from_base < time_left−1) then
        safe_destinations = [safe_destinations destinations(i)]
    end if
end for
restricted_paths = make_q_paths(x, safe_destinations, q )
return(restricted_paths)
```

The function *virtual_graph*, creates a virtual graph with nodes as the cell centers of the entire search space and the weight of each arc being unity. The function *shortestPath* finds the least number of cells required to traverse from the base cell to the destination cell. The function *make_q_paths* generates all possible paths of length q from the current location of the agent.

The search region may contain some forbidden regions as shown in Figure 4(b), hence it is necessary to use the shortest path algorithm to find the shortest path for the agent to return back to the base station. In the search space, if the forbidden regions were not there, then an analytical expression can be derived using the indices of the cells to get the shortest route.

4.2. *Multiple Base Stations*

In practical situations, there can be more than one base in a region and the agent can choose the base in which it may land for refuelling. After refuelling in a base station, the agent flies for another sortie. The agents can start their search from one base and land in a different base. There is no restriction on the selection of the base for the agent, but the agents start their search operations simultaneously. In this section, the route planning algorithm of a single base station is extended to addresses the problem with multiple base stations. Figure 5 shows a search mission with multiple bases (B1, B2, B3) and three different sorties.

Consider that at the m^{th} time step a UAV is at cell C^i and let $\hat{\mathcal{C}}$ be the q step neighborhood cells of the UAV from cell C^i. Similar to the single base case a virtual graph of the uncertainty map is constructed but taking the presence of all the base stations in the region. Every cell $C^k \in \hat{\mathcal{C}}$, is checked to determine if by using the cell C^k the searcher can reach any one of the bases, which may not necessarily be the base from which the UAV has started its search operation. For all the sorties it may be possible to have the choice of all the bases for the UAVs. Depending on the search route they have adopted, the UAVs may find some bases or no other base stations except the base station from which it has started the sortie. At every time step, the decision to move from one cell to its neighboring cell is carried out using a game theoretical strategy which will be described later.

Algorithm 2 Algorithm for Multiple Bases

```
Function: return_base(x, base, q_neighbouring_cells, q, time_left)
source = base; destinations = q_neighbouring_cells;
graph = virtual_graph(cells) /* virtual graph with all cells */
safe_destinations = [ ] /* initialize */
for j=1:no_of_bases do
   for i = 1 : no_of_destinations do
      min_distance_from_base = shortestPath(source(j), destinations(i), graph)
      if (min_distance_from_base < time_left−1) then
         safe_destinations = [safe_destinations destinations(i)]
      end if
   end for
end for
restricted_paths = make_q_paths(x, safe_destinations, q )
return(restricted_paths)
```

The k-shortest path algorithm based search as described in Sujit and Ghose (2004-AES) can be used for multiple bases, provided the destination base station are specified for refuelling before the start of the search. Since the agents do not consider the presence of neighbouring agents during the sortie, the effectiveness of the search may reduce due to search route overlaps in common regions. Also, the k-shortest path algorithm fails when multiple agents start from the same base station as all the agents choose the same path. These shortcomings of the k-shortest path algorithm based uncertainty reduction search can be eliminated with game theoretical real time search mechanism and using the path planning algorithms which guarantees safe return to any of the bases.

There could be a concern that, at each step, a cell that allows the agent to return to a base station is selected and this kind of selection mechanism can lead to a case where a finite number of sequences form a cyclic loop. The formation of such cyclic loops is not possible because the shortest path determines those cells that ensure the agent will return to any of the base stations. But choosing a particular cell for the next time step is based on the game theoretical strategies. These strategies choose those cells that yield high uncertainty reduction. Once a cell is visited, its value decreases (Equation (1)). Since, the value decreases, the strategies will not choose this cell, unless this cell still has higher uncertainty value than the other cells, or this is the only remaining cell. Therefore, the possibility of a set of cells being selected always to form a cycle is not possible. Each agent chooses its next path segment based on various game theoretical strategies that are described next.

5. Game Theoretical Model and Strategies

5.1. *A Game Model*

The agents make decisions during the search operations and these decisions are based on a game theoretical model. In this section, the game theoretical model is formulated using q-step look ahead planning, as proposed in Passino *et al.* (2000) (where q determines the depth of the exploratory search environment) to obtain optimal strategies. In Sujit and Ghose (2004-AES), the implementation of the search algorithm required the knowledge of the complete uncertainty map. Further, the route decision of an agent was determined independent of the other agents' current decisions. In the present paper, the algorithm that takes into account possible actions of the other agents with different levels of cooperation. Here, two agents are considered which are performing the search operation and formulate the game as a bimatrix game. The payoff that an agent receives by considering the actions of the other agent are expressed in terms of a search effectiveness matrix.

5.1.1. *Search Effectiveness Matrix*

Let a cell's uncertainty value be U_i. Let $\mathcal{P}_i^q(C_{s_i})$, $i \in \{1, 2\}$ be the set of all possible paths of length q for agent A_i, emanating from cell C_{s_i}. A path $P_i^j(C_{s_i}) \in \mathcal{P}_i^q(C_{s_i})$, $j = 1, 2, \ldots, |\mathcal{P}_i^q(C_{s_i})|$, is a sequence of cells

$$P_i^j(C_{s_i}) = [C^1, C^2, C^3, \ldots, C^q] \tag{3}$$

where $C^k \in \mathcal{C}$ ($\mathcal{C}$ is the collection of all cells), $C^1 = C_{s_i}$, the current position of A_i, and $C^{k+1} \in \mathcal{N}(C^k)$ where, $\mathcal{N}(C^k)$ is the set of all neighboring cells of C^k.

Let the uncertainty value of cell C^k at time t be $U(C^k, t)$. Given a path $P_i^j(C_{s_i})$ of agent A_i, suppose A_i is at cell C^l at time t, then the reduction in uncertainty associated

with C^l, and the subsequent updated value of uncertainty, is evaluated as follows:
Case 1: Only A_i is at cell C^l at time t, then

$$v_i(t) = U(C^l, t)\beta_i$$
$$U(C^l, t+1) = U(C^l, t) - v_i(t) \quad (4)$$

Case 2: A_1 and A_2 are both at cell C^l, then

$$v_i(t) = \frac{\beta_i}{\beta_1 + \beta_2} U(C^l, t)[1 - (1-\beta_1)(1-\beta_2)]$$
$$U(C^l, t+1) = U(C^l, t) - (v_1(t) + v_2(t)) \quad (5)$$

So, given two routes $P_1^j(C_{s_1})$ and $P_2^{j'}(C_{s_2})$ of the two agents, the reduction in uncertainty achieved by A_i at each step t ($t = 1, 2, \ldots, q$) is given by $v_i(t)$ and is computed using Case 1 or Case 2. Note that this computation has to be carried out simultaneously for both agents. The total benefit to A_i due to path $P_i^j(C_{s_i})$ is

$$V(P_i^j(C_{s_i})) = \sum_{t=1}^{q} v_i(t) \quad (6)$$

The search effectiveness matrix M^i has dimension $|\mathcal{P}_1^q(C_{s_1})| \times |\mathcal{P}_2^q(C_{s_2})|$ and every element $m^i_{(k,l)}$ of the matrix represents the payoff $V(P_i^j(C_{s_i}))$ obtained by the agents, when A_1 chooses the path $P_1^k(C_{s_1}) \in \mathcal{P}_1^q(C_{s_1})$ and A_2 selects the path $P_2^l(C_{s_2}) \in \mathcal{P}_2^q(C_{s_2})$.

5.2. *Solution Concepts*

The decision to choose a particular path that would provide the maximum information gain (or uncertainty reduction) can be based on various strategies. The strategies can be classified into two categories (one being those strategies that can be derived by the present framework, and the other is those that can be derived using extra information/coordination/communication). The strategies that do not require communication and uses the current state information about the environment are non-cooperative Nash equilibrium, security strategy, and greedy strategy. While cooperative and global optimal strategies require communication/extra information.

Noncooperative Nash Equilibrium: This strategy is used when the agents do not communicate with each other to decide on their future action at time t, and each agent assumes the other agent to take actions that are individually beneficial to them.

Security Strategy: This strategy becomes relevant when the agents do not communicate with each other and each agent assumes the other agent to be an adversary who is likely to take actions that are harmful to the first agent without heed to its own self interest. In such a situation the best strategy for the agent is to secure its minimal benefit. Hence, it is logical for the agent to a use security strategy that would guarantee a minimal payoff.

Cooperative Strategy: The agents communicate with each other and decide collectively (jointly) to take the best possible action. This is also the centralized case.

Greedy Strategy: The agents do not communicate among themselves and use greedy strategy for future actions. An agent does not consider the effect of the possible actions of

the other agents and selects an action that yields the maximum benefit to itself according to (4).

Globally Optimal Strategy: The game theoretical strategies are based on local information up to q steps. Hence, the solution is optimal for these q steps and are not globally optimal. The globally optimal solution can be obtained by making q equal to the largest possible number of steps in an agent's search path. This requires prohibitively large computational time and also increases the computational complexity as the domain of the search effectiveness function increases. This strategy is not considered in the present paper, but some heuristic algorithms to implement such strategies have been discussed in Dell *et al.* (1996).

5.3. *Two-Person Game Model*

The two-person game problem can be formulated as a bi-matrix game, and use it to find the equilibrium strategies.

5.3.1. *Non-cooperative Nash equilibrium strategy*

A non-cooperative bimatrix game for two players or agents Basar and Olsder (1995) is defined as a game consisting of two search effectiveness matrices, $M^1 = \{m^1_{kl}\}$ and $M^2 = \{m^2_{kl}\}$, with each pair of entries (m^1_{kl}, m^2_{kl}) denoting the payoff to each agent respectively, corresponding to a pair of decisions made by the players. The players do not cooperate with each other and arrive at their decisions independently. In such a situation the equilibrium solution can be stated as follows:

A pair of strategies {row k^*, column l^*} is said to constitute a noncooperative (Nash) equilibrium solution to the bimatrix game, if the following pair of inequalities are satisfied, $\forall\ k = 1, 2, \ldots, |\mathcal{P}^q_1(C_{s_1})|$ and $\forall\ l = 1, 2, \ldots, |\mathcal{P}^q_2(C_{s_2})|$

$$m^1_{k^*l^*} \geq m^1_{kl^*}, \quad m^2_{k^*l^*} \geq m^2_{k^*l} \tag{7}$$

The pure strategy Nash equilibrium may not exist always, in which case mixed strategies that guarantee a solution to the noncooperative game need to be computed.

Mixed Strategies: A mixed strategy for a player is a probability distribution on the space of its pure strategies. A mixed strategy for A_1 is to choose '*row* 1' with probability (w.p.) y_1, '*row* 2' w.p. $y_2, \ldots$, and '*row* $|\mathcal{P}^q_1(C_{s_1})|$' w.p. $y_{|\mathcal{P}^q_1(C_{s_1})|}$, so that,

$$\sum_{k=1}^{|\mathcal{P}^q_1(C_{s_1})|} y_k = 1 \tag{8}$$

The mixed strategy space for A_1 is denoted as Y, while for A_2 it is denoted as Z. A pair $\{y^* \in Y, z^* \in Z\}$ is said to constitute a *noncooperative (Nash) equilibrium solution* to the bimatrix game (M^1, M^2) in *mixed strategies*, if the following inequalities are satisfied $\forall\ y \in Y,\ \forall\ z \in Z$:

$$y^{*'} M^1 z^* \geq y' M^1 z^*,\ y \in Y, \quad y^{*'} M^2 z^* \geq y^{*'} M^2 z,\ z \in Z \tag{9}$$

Computation of mixed strategy equilibrium solution can be posed as a bilinear programming problem Basar and Olsder (1995) as follows:

A pair $\{y^*, z^*\}$ constitutes a mixed-strategy Nash Equilibrium solution to a bimatrix game (M^1, M^2) if, and only if, their exists a pair $(f^*,\ g^*)$ such that $\{y^*, z^*, f^*, g^*\}$ is a solution of the following bilinear programming problem:

$$\min_{y,z,f,g} [-y^{'} M^1 z - y^{'} M^2 z + f + g] \tag{10}$$

subject to

$$-M^1 z \geq -f \cdot 1_{|\mathcal{P}_1^q(C_{s_1})|},\ \ -M^{2'} z \geq -g \cdot 1_{|\mathcal{P}_2^q(C_{s_2})|}$$

$$y \geq 0,\ z \geq 0,\ y^{'} \cdot 1_{|\mathcal{P}_1^q(C_{s_1})|} = 1,\ z \cdot 1_{|\mathcal{P}_2^q(C_{s_2})|} = 1 \tag{11}$$

where, $1_{|\mathcal{P}_1^q(C_{s_1})|}$ and $1_{|\mathcal{P}_2^q(C_{s_2})|}$ are column vectors of dimensions $|\mathcal{P}_1^q(C_{s_1})|$ and $|\mathcal{P}_2^q(C_{s_2})|$, with all elements equal to 1.

The dimension of the search effectiveness matrix increases with q. Thus, computing the mixed strategy equilibrium using the bilinear programming formulation may become computationally time consuming, therefore, the dominating strategies concept as described below is used for reducing the size of the search effectiveness matrix.

Dominating Strategies (Basar and Olsder (1995)): Some rows and columns can be eliminated that have no influence on the equilibrium solution. The 'row i' of matrix M_1 is said to dominate row k if $m_{ij}^1 \geq m_{kj}^1,\ \forall\ j = 1, 2 \ldots, |\mathcal{P}_2^q(C_{s_2})|$ and if, for at least one j, the strict inequality holds. Similarly, for A_2, 'column j' of M^2 is said to dominate 'column l' if $m_{ij}^2 \geq m_{il}^2,\ \forall\ i = 1, 2, \ldots, |\mathcal{P}_1^q(C_{s_1})|$, and if, for at least one i, the strict inequality holds. The dominated strategies (row k and column l, in the above example) can be eliminated without affecting the equilibrium solution. The resultant matrix dimensions will be smaller than the original $|\mathcal{P}_1^q(C_{s_1})| \times |\mathcal{P}_2^q(C_{s_2})|$ and will need less computational time to compute the mixed equilibrium strategy.

5.3.2. *Security strategy*

When there is no communication between the agents each agent assumes the other agent behaves as an adversary. In this situation the best strategy for the agent is to secure its minimal benefit. The strategy that the agent chooses to secure its profit is called the security strategy. For a bimatrix game, agent A_1 chooses a 'row k^*' whose smallest entry is no smaller than the smallest entry of any other row, which implies

$$k^* = \arg\max_k \{\min_l m_{kl}^1\} \tag{12}$$

while agent A_2 chooses a strategy l^* given as

$$l^* = \arg\max_l \{\min_k m_{kl}^2\} \tag{13}$$

where, $k \in \{1, 2, \ldots, |\mathcal{P}_1^q(C_{s_1})|\}$ and $l \in \{1, 2, \ldots, |\mathcal{P}_2^q(C_{s_2})|\}$.

5.3.3. *Cooperative strategy*

A pair of strategies {'row k^*', 'column l^*'} is said to be a cooperative strategy, if the following condition is satisfied.

$$m_{k^*l^*} \geq m_{kl} \tag{14}$$

$\forall\ k = 1, 2, \ldots, |\mathcal{P}_1^q(C_{s_1})|$ and $\forall\ l = 1, 2, \ldots, |\mathcal{P}_2^q(C_{s_2})|$.

5.3.4. *Greedy strategy*

The agent chooses a path P_i^k with a look ahead policy of q, using the following relation:

$$V(P_i^k) \geq V(P_i^j), \quad \forall \quad j = 1, 2, \ldots, |\mathcal{P}_i^q| \tag{15}$$

where, $V(P_i^k)$ is the benefit obtained by agents A_i using the path P_i^k only and it is evaluated using Eqn. (4).

5.4. *Selection of Strategies*

When there are multiple solutions, the selection of strategies by players becomes a crucial issue. The security strategies and cooperative strategies are straightforward to implement. If there exists multiple security strategies, any one of them will guarantee the same payoff. In fact, the actual payoff is bound to be higher for both players so long as they stick to their security strategies. In the case of multiple cooperative strategies, since players communicate with each other during the decision process, they can decide on a strategy which is beneficial to the overall goal. But, when multiple solutions occur for pure or mixed strategy Nash equilibrium, one of the solutions need to be selected. Since every agent has the search effectiveness matrix of all the agents, selecting a solution that maximizes the joint payoff is an ideal strategy. The selection of solution does not involve any communication with the other agent, but uses the available data in the search effectiveness matrix, and may be based upon some sort of prior agreement. The selection procedure is described in the algorithm for noncooperative Nash strategy.

When mixed strategy equilibrium exists, a random number may be generated according to the probability distribution of the optimal mixed strategy to select the appropriate strategy. There can be other kind of selections too, such as choosing the strategy that has the highest probability (maximum likelihood). In the simulations, the random number generation method is used.

6. Simulation Results

The performance in terms of uncertainty reduction achieved taking the endurance time constraints of the UAVs into account is evaluated using simulations. The UAVs use various game theoretical strategies to perform the mission. Through simulations the performance of each strategy is analyzed. Especially, the noncooperative Nash strategy and cooperative strategies when the variation of the uncertainty maps is high due to degrading sensor accuracy.

Table 1. Time taken by various strategies for each search step on P4 3Ghz machine

Strategy	Time for computation in msec	
	$q = 1$	$q = 2$
Cooperative	6.2	232.3
Nash	3.5	200.7
Security	3.9	26.5
Greedy	0.86	2.8

6.1. *Ideal case*

In the ideal case, each agent has complete information about the other agents' position in the search space and about its uncertainty reduction factor. Simulations are carried out on a 30×30 hexagonal grid for 50 different uncertainty maps. The search space has 3 base stations B_1, B_2, and B_3 as shown in Figure 6. The uncertainty reduction factors are $\beta_1 = 0.65$ and $\beta_2 = 0.5$. The search mission was carried out for 10 sorties with each sortie of 60 search steps. The mission is started with two agents placed at base station B_1 for all the uncertainty maps. The agents dynamically choose any base station for refuelling. Figure 6 shows the search route for two agents. From the figure it is observed that agent A_1 chooses B_1 for refueling while A_2 chooses B_2. The next sortie starts from the current base station where the UAV has landed. This process continues for 10 sorties. Then, the performance of the game theoretical strategies and the effect of increase in look ahead step length on the search performance is analyzed.

Figure 7 shows the performance of noncooperative Nash, cooperative, security and greedy strategies for look ahead step lengths of $q = 1$ and $q = 2$. The results show that the game theoretical strategies out perform greedy strategies. Among the game theoretical strategies, for $q = 1$ and $q = 2$, noncooperative Nash strategy performs as well as the cooperative strategy, while security strategy performs slightly worse.

From these results, it can be concluded that the noncooperative Nash and cooperative strategies are the best in terms of reduction in uncertainty for both $q = 1$ and $q = 2$. Although, there is an increase in uncertainty reduction with increased look ahead step length, there is a corresponding increase in the computational time. Table 1 shows the time taken by various strategies for each search step.

6.2. *Sector partitioning strategy*

From Table 1, it is observed that the computational time increases exponentially with increase in the look-ahead step size. This is because, with increase in q the number of paths increases and computing the bi-matrix for larger number of paths increases the computational time. Hence, there is a need to reduce the computational time in decision-making.

In order to reduce the computational time, the number of paths are limited by choosing those regions that have higher uncertainty. These regions are generated by partitioning of the search space into sectors. The q neighboring cells of an UAV is partitioned into six sectors as shown in Figure 8 and evaluate the average uncertainty in a sector (S_{avg}) as:

$$S_{avg}(k) = \frac{\sum_{j=1}^{|N_s|} \eta U_j}{|N_s|}, \quad k = 1, 2, \ldots, 6, \quad \eta \in \{1, 0.5\} \tag{16}$$

where, $|N_s|$ represent the number of cells present in a sector, and U_j is the uncertainty of the cell C_j. If the cell is entirely inside the sector then $\eta = 1$, else if the cell is common to two sectors (see Figure 8) then $\eta = 0.5$, that is, only half its uncertainty is considered. The values $S_{avg}(k)$ are sorted in a decreasing order of average uncertainty in the sectors. Selecting the number of sectors required for computing the search effectiveness matrix is a tradeoff between increase in computational burden and the uncertainty reduction achieved. To obtain maximum uncertainty reduction the searchers should choose different routes. Hence, allowing the searcher to choose a path from all the six sectors would be equivalent to a case without sectoring and the computational time requirement is very high. When the searchers are far apart, a sector that has the maximum average uncertainty can be selected. But selecting only one sector would imply that the searcher is forced to move in just one direction and not allowing any flexibility. Hence, to provide some freedom on the searcher movements, two best sectors from S_{avg} are selected for computing the search effectiveness matrix. When the searchers are present in the same cell there would be flexibility in allowing them to choose one or two different cells for the next search step. So, it is necessary to allow for at least two sectors for the searchers to select routes from. In the simulations, three sectors are selected when the searchers are in the same cell and two sectors when the searchers are far apart.

Figure 9 shows the relative performance of all the strategies that are similar to those shown in Figure 7. The variation in performance of the strategies with sector partitioning to that of without sector partitioning can be seen in Figure 10. The figure shows the percentage deviation between various strategies for look ahead steps of $q = 1$ and $q = 2$. The maximum deviation of the performance between non-partitioned scheme and partitioned scheme is around 4.5%. Since the deviation is small, it can be claimed that the partitioning schemes yield results that are close to non-partitioned schemes but with reduced computational time.

The performance of all the strategies for $q = 1$ and $q = 2$, with and without sector partition, is similar. Hence, to determine whether the results are persistent with larger value of q, a different experiment is conducted with ten different random maps using two searchers and two bases. The searchers need to perform five sorties and each sortie has 50 time units. The uncertainty reduction factors are $\beta_1 = 0.6$ and $\beta_2 = 0.5$ and the look-ahead length is $q = 4$. From earlier results, it is evident that the cooperative strategy and non-cooperative strategies perform better than the rest of the strategies. Therefore, this experiment was limited to these two strategies. The performance of the agents is shown in Figure 11. It can be observed from the figure, that the non-cooperative Nash performs similar to the cooperative strategy even for larger q values. Thus sector partitioning is a good strategy to reduce the computational complexity with almost similar performance to that of without sector partition.

6.3. *Variation in β*

To demonstrate the utility of the Nash strategies when the perceived uncertainty maps of the agents are different from the actual uncertainty map, a set of experiments were conducted. For this set of experiments, it was assumed that the uncertainty reduction factors (β) of the agents fluctuate with time due to fluctuation in the performance of their sensor suites due to environmental or other reasons. Each agent knows its own current uncertainty reduction factor perfectly but assumes that the uncertainty reduction factors of the other agents to be the same as their initial value. This produces disparity in the uncertainty map between agents and from the actual uncertainty map which evolves

according to the true β values as the search progresses.

The simulation was carried out for 10 different uncertainty maps with 30×30 cells. The initial uncertainty reduction factors are 0.65 and 0.5. The variation in the value of β for the two agents is shown in Figure 12. Initially, the search mission is started with both agents placed at the same base station. Figure 13 shows the performance of the two strategies. The results shown that the average uncertainty reduction obtained using Nash strategies that do not make any assumption about the other agents' actions performs better than the cooperative strategy which assumes cooperative behavior from the other agents.

7. Conclusions

In this paper, a 2-person game model is proposed for agents with endurance time constraints performing a search operation in an unknown region. The UAVs need to refuel at a base station after performing the mission. Hence, a dynamic base station algorithm for UAVs is developed so that they can return to the base station in time. Several game theoretical strategies namely, noncooperative Nash, security, and strategies, using 2-person game theory are proposed for the agents. The performance of each strategy for one step and two step look ahead policies was evaluated through simulations and was compared against greedy strategy. The simulation results show that the game theoretical strategies outperform greedy strategy. In the game theoretical strategies, cooperative, and noncooperative Nash strategies perform equally well and take almost the same order of computational times, while security strategy performs quite close to cooperative and noncooperative Nash strategies. The computational time results show that these strategies can be used for real time search in a large unknown region. The effectiveness of uncertainty reduction increases with increase in the look ahead step size but this also causes an increase in the computational time considerably. For this purpose, a heuristic based on partitioning the search space into sectors was developed, and it was found that the performance with sectors is good as without sectors but with far less computational time. In the case of imperfection in the information that causes disparity in the perceived uncertainty map information of the agents, the non-cooperative game theoretical strategies are shown to be more effective than the cooperative strategy.

References

Basar, T. and Olsder, G.J., 1995. *Dynamic Noncooperative Game Theory*. 2nd ed. California, USA: Academic press.

Batalin, M.A. and Sukhatme, G.S., 2002. Spreading out: A local approach to multi-robot coverage. *Proc. of the International Symposium on DistributedAutonomous Robotics Systems*, Fukuoka, Japan, June 2002, 373-382.

Beard, R.W., *et al.*, 2002. Coordinated target assignment and intercept for unmannedair vehicles. *IEEE Trans. on Robotics and Automation*, 18(6), 911-922.

Beard, R.W. and McLain, T.W., 2003. Multiple UAV cooperative search under collision avoidance and limited range communication constraints. *Proc. of the IEEE Conference on Decision and Control*, Maui, Hawaii, Dec 2003, 25-30.

Benkoski, S.J., Monticino, M.G., and Weisinger, J.R., 1991. A survey of the search theory literature. *Naval Research Logistics*, 38(4), 469-494.

Bortoff, S.A., 2000. Path planning for UAVs. *Proc. of the American Control Conference*, Chicago, Illinois, June 2000, 364-368.

Burgard, W., Moors, M., and Schneider, F., 2002. Collaborative exploration of unknown environment with teams of mobile robots. In: M. Beetz, J. Hestzberg, M. Ghallab, and M.E. Pollack, eds. *Advances in Plan-Based Control of Robotic Agents*. London: Springer Verlag, Lecture Notes in Computer Science: 2466, 52-70.

Dell, R.F. *et al.*, 1996. Using multiple searchers in constrained-path, moving-target search problems. *Naval Research Logistics*, 43(4), 463-480.

Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, 269-271.

Enns, D., Bugajski, D., and Pratt, S., 2002. Guidance and control for cooperative search. *Proc. of the American Control Conference*, Anchorage, Alaska, May 2002, 1923-1929.

Ganapathy, S. and Passino, K.M., 2003. Agreement strategies for cooperative control of uninhabited autonomous vehicles. *Proc. of the American Control Conference*, Denver, Colorado, June 2003, 1026-1031.

Geiger, B.R., *et al.*, 2006. Optimal path planning of UAVs using direct collocation with nonlinear programming. *Proc. of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, Colorado, August 2006, AIAA 2006-6199.

Goldsmith, S.Y. and Robinett, R., 1998. Collective search by mobile robots using alpha-beta coordination. *In:* A. Dragoul, M. Tambe, and T. Fukuda, eds. *Collective Robotics*. London: Springer-Verlag, Lecture Notes in Artificial Intelligence: 1456, 135-146.

Mot, J.D. *et al.*, 2000. Coordinated path planning for a UAV cluster. *AINS symposium*, LA, CA, May 2000.

Nikolos, I.K. and Brintaki, A.N., 2005. Coordinated UAV path planning using differential evolution. *Proc. of the Mediterranean Conference on Control and Automation*, Limassol, Cyprus, June 2005, 549-556.

Nygard, K.E., Chandler, P.R., and Pachter, M., 2001. Dynamic network flow optimization models for air vehicle resource allocation. *Proc. of the American Control Conference*, Arlington, VA, June 2001, 1853-1858.

Passino, K.M., *et al.*, 2000. Cooperative control for autonomous air vehicles. *Proc. of the Cooperative Control and Optimization Workshop*, Florida, December 2000.

Polycarpou, M., Yang, Y., and Passino, K.M, 2001. A cooperative search framework for distributed agents. *Proc. of the IEEE International Symposium on Intelligent Control*, Mexico City, Mexico, Sept 2001, 1-6.

Rajnarayan, D.G. and Ghose, D., 2003. Multiple agent team theoretic decision-making for searching unknown environments. *Proc. IEEE Conference on Decision and Control*, Maui, Hawaii, Dec 2003, 2543-2548.

Ryan, A. and Hedrick, J.K., 2005. A mode-switching path planner for UAV-assisted search and rescue. *Proc. of the IEEE Conference on Decision and Control, and the European Control Conference*, Seville, Spain, December 2005, 1471-1476.

Scerri, P., *et al.*. Coordinating very large groups of wide area search munitions. *Theory and Algorithms for Cooperative Systems*. World Scientific Publishing, 2005.

Shanmugavel, M., *et al.*, 2006. 3D Dubins sets based coordinated path planning for swarm of UAVs. *Proc. of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, Colorado, August 2006, AIAA 2006-6211.

Spires, S.V. and Goldsmith, S.Y., 1998. Exhaustive geographic search with mobile robots along space filling curves. *In:* A. Dragoul, M. Tambe, and T. fukuda, eds. *Collective Robotics*. London:Springer-Verlag, Lecture Notes in Artificial Intelligence: 1456, 1-

12.
Stone, L.D., 1975. *Theory of Optimal Search*. New York:Academic Press, 1975.
Sujit, P.B. and Ghose, D., 2004. Search using multiple UAVs with flight time constraints. *IEEE Transactions on Aerospace and Electronic Systems*, 40(2), 491-509.
Sujit, P.B. and Ghose, D., 2004. Multiple agent search of an unknown environment using game theoretical models. *Proc. of the American Control Conference*, Boston, June 2004, 5564-5569.
Wong, H., Kapila, V., and Vaidyanathan,R., 2004. Optimal path planning using CCC class paths for target. *Proc. of the IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas, December 2004, 1105-1110.
Yang, Y., Minai, A.A., and Polycarpou, M.M., 2002. Decentralized cooperative search in UAV's using opportunistic learning. *Proc. of the AIAA Guidance, Navigation and Control Conference*, Monterey, CA, August 2002, AIAA-2002-4590 .
Yang, Y., Minai, A.A., and Polycarpou, M.M., 2002. Analysis of opportunistic method for cooperative search by mobile agents. *Proc. of the IEEE Conference on Decision and Control*, Las Vegas, Nevada, December 2002, 576-577.

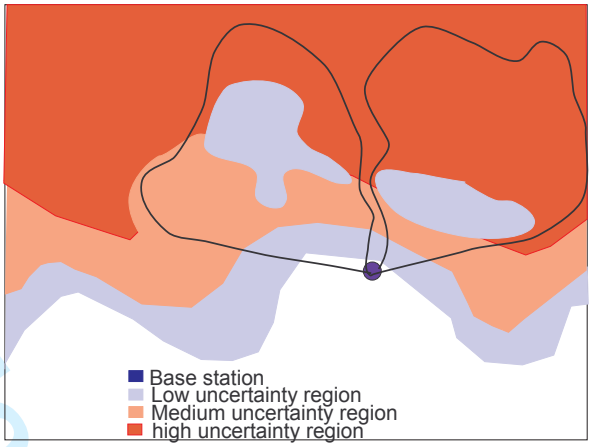


Figure 1. A typical search scenario

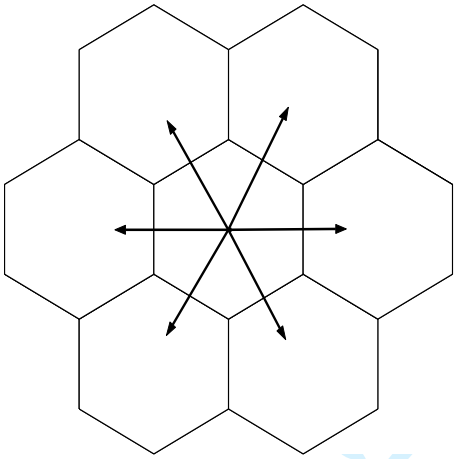


Figure 2. Partitioning of the search space into hexagonal cells

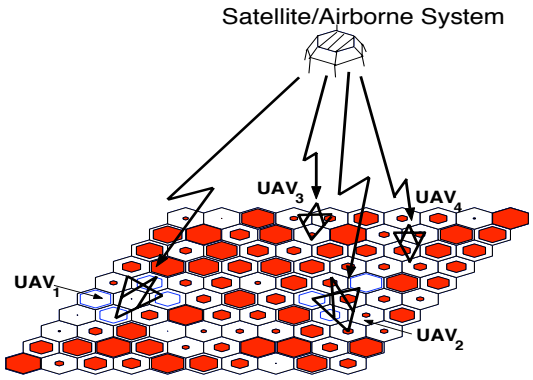
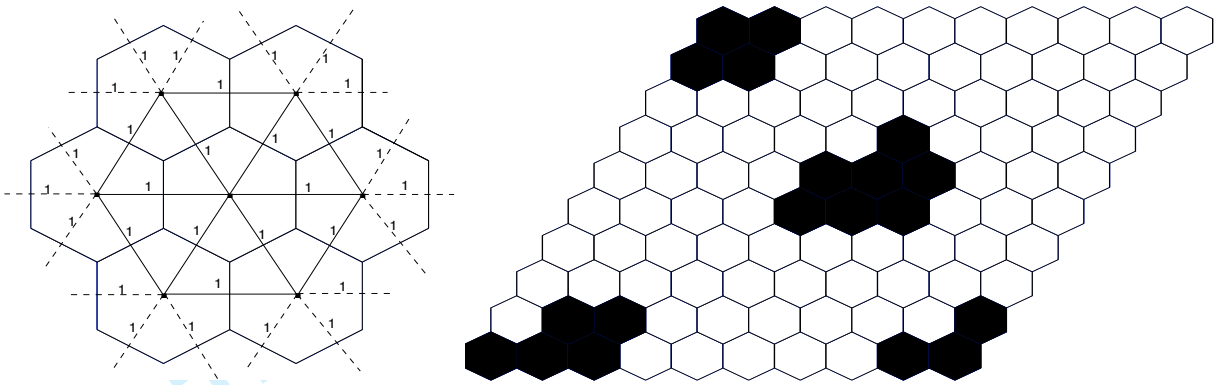


Figure 3. Ideal case scenario for the application of noncooperative Nash or security strategies



(a) (b)

Figure 4. (a) Virtual Graph with unity cost (b) Search region with forbidden areas

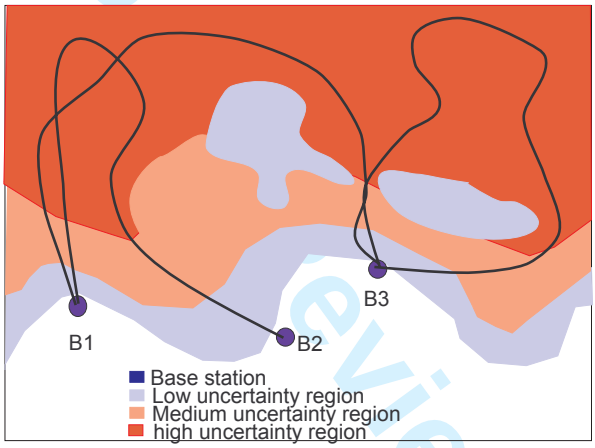


Figure 5. A typical search scenario with multiple bases

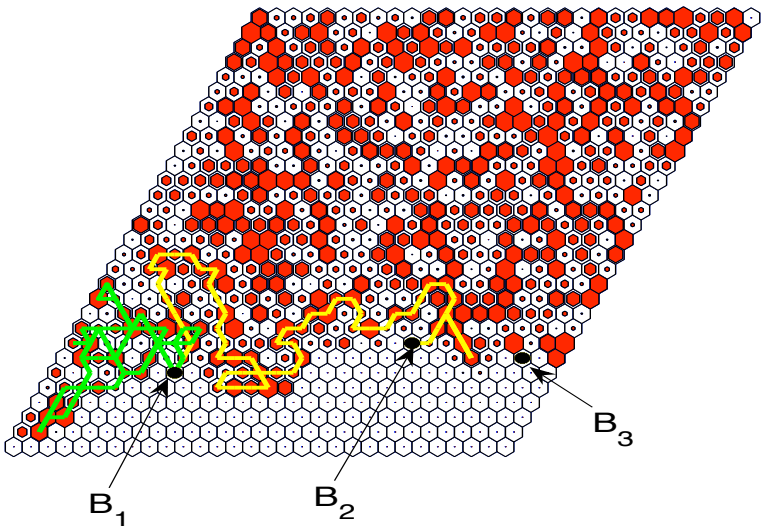


Figure 6. An uncertainty graph with 3 base stations. A route traversed by agents A_1 and A_2 for one sortie using cooperative strategy with look ahead policy $q = 2$

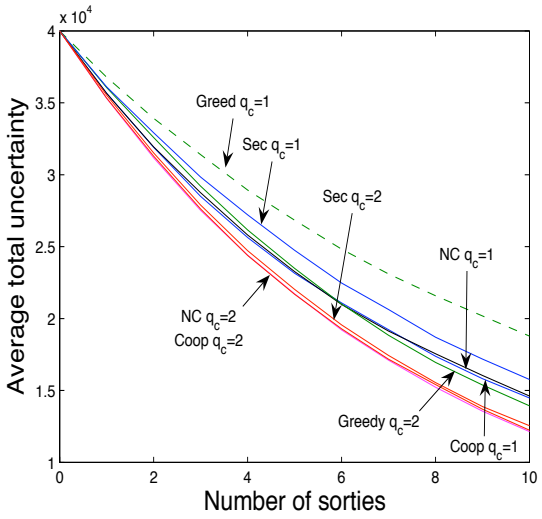


Figure 7. Performance in the ideal case with β constant

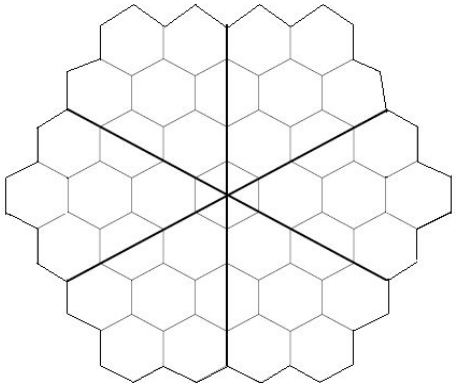


Figure 8. Partitioning the search space into sectors with $q = 3$ neighborhood cells

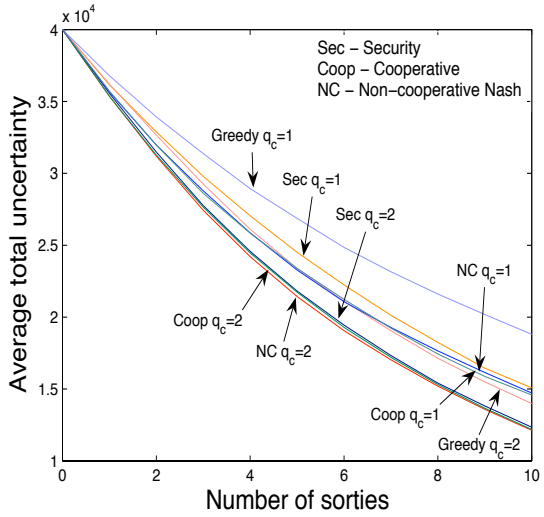
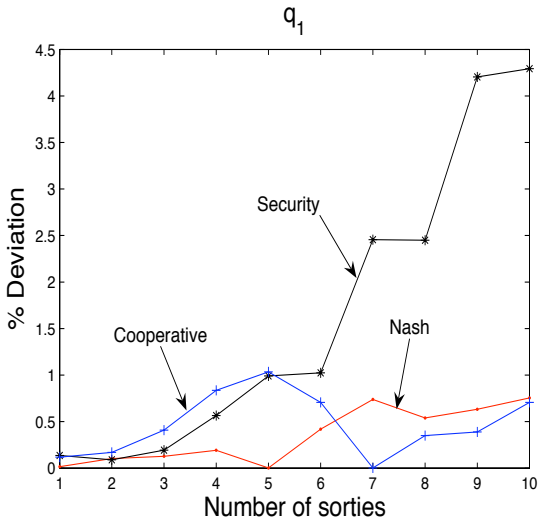
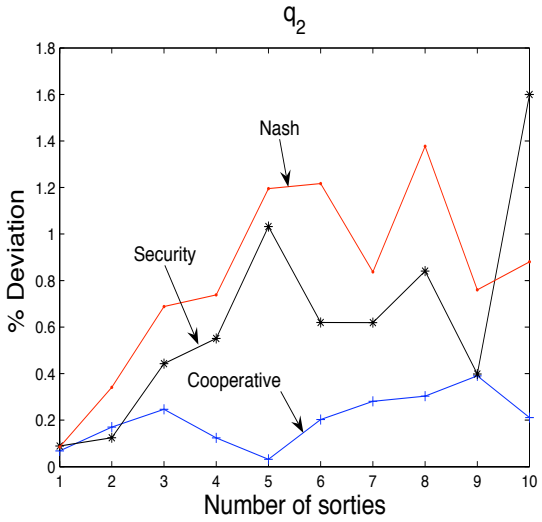


Figure 9. Performance of various strategies with search space partitioned into sectors



(a)



(b)

Figure 10. Performance variation with and with sectoring for different strategies

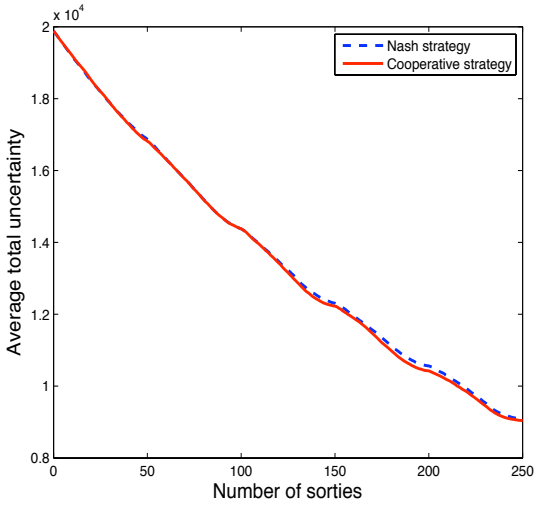


Figure 11. Performance of non-cooperative Nash strategy and cooperative strategy for $q = 4$

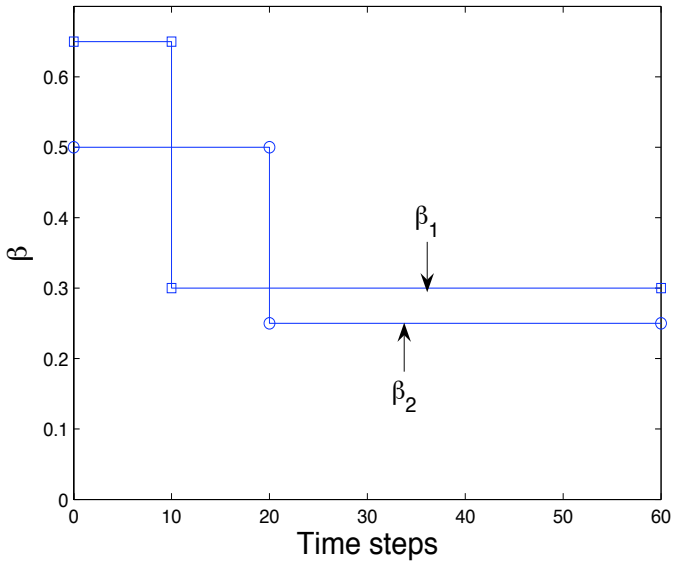


Figure 12. Variation in the uncertainty reduction factors

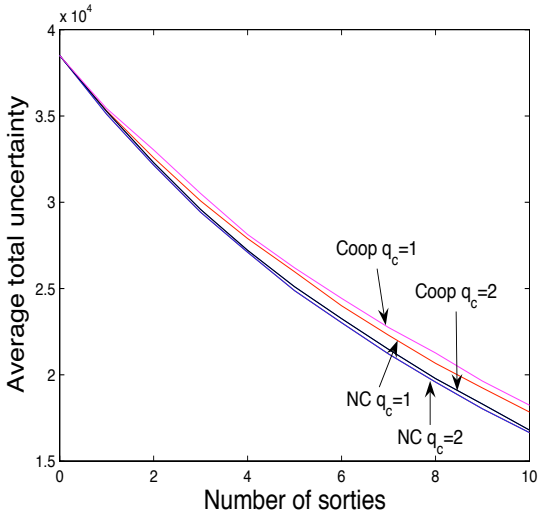


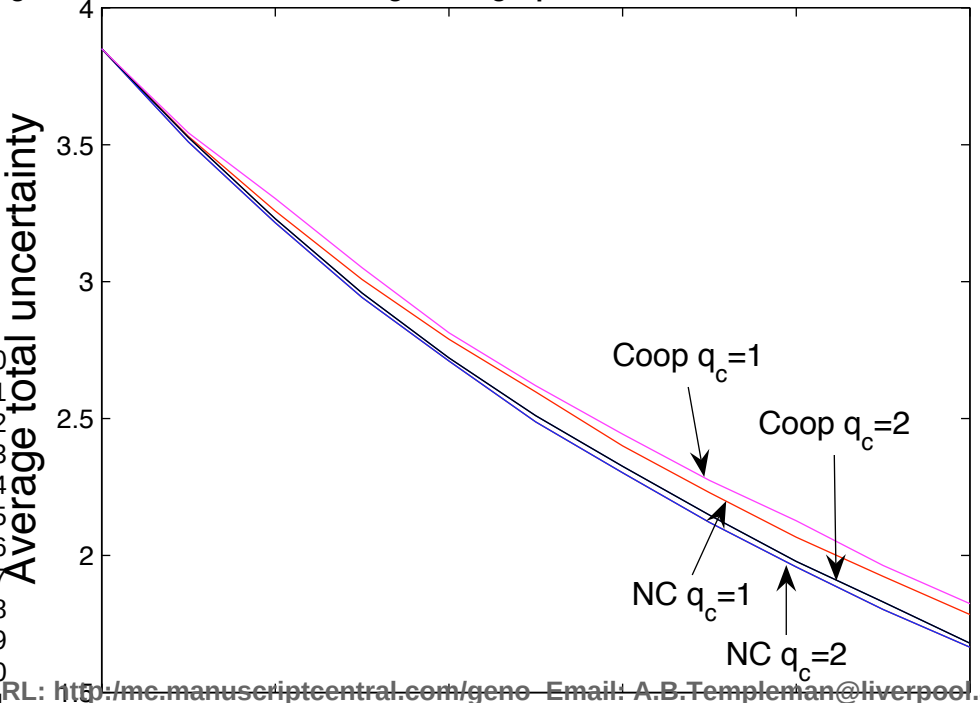
Figure 13. Performance in the non-ideal case with varying β

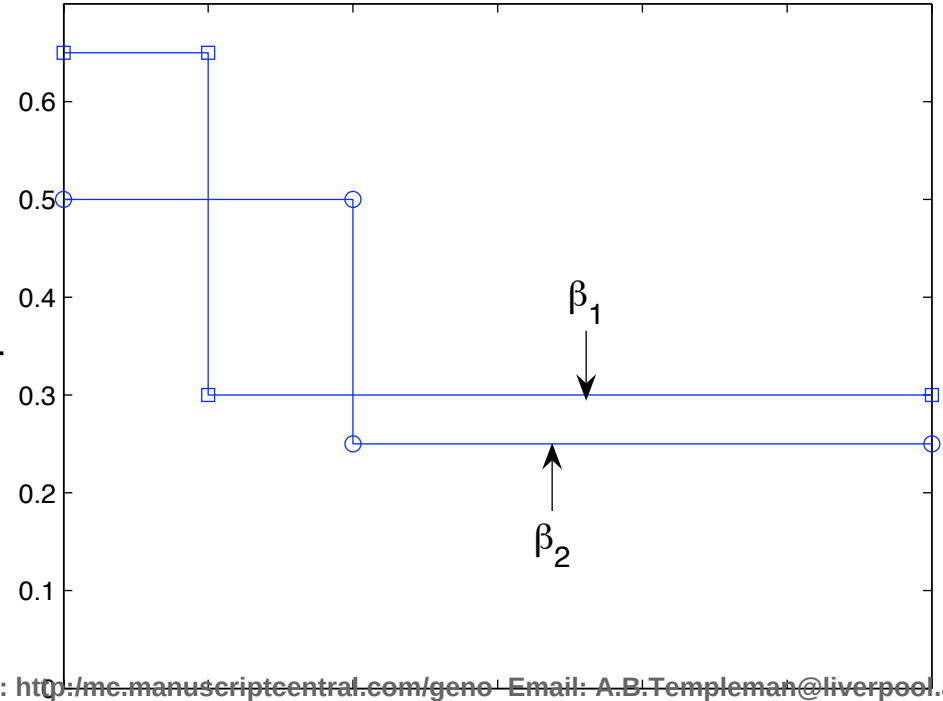
Appendix A. Dijkstra's Shortest Path Algorithm:

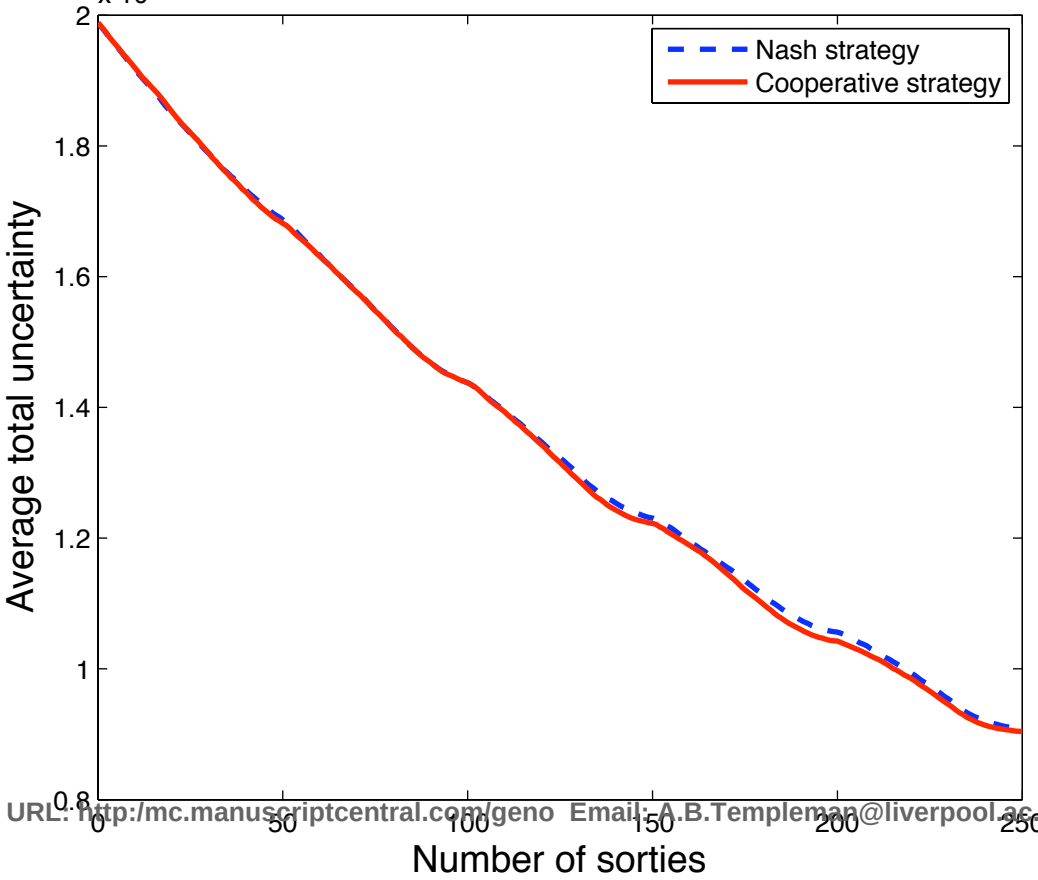
Given a directed graph $G = (V, E)$, where V is the set of nodes and E is the set of arcs, and a specified source node $s \in V$, where each edge $(i, j) \in E$ has a specified non-negative cost C_{ij}, the problem is to find, for each node $i \in V$, the shortest path from s to i. The algorithm created a directed shortest path tree T rooted at the source node s. When $(i, j) \in T$ then $pred(j) = i$, where i is said to be the predecessor of j. The algorithm is as follows:

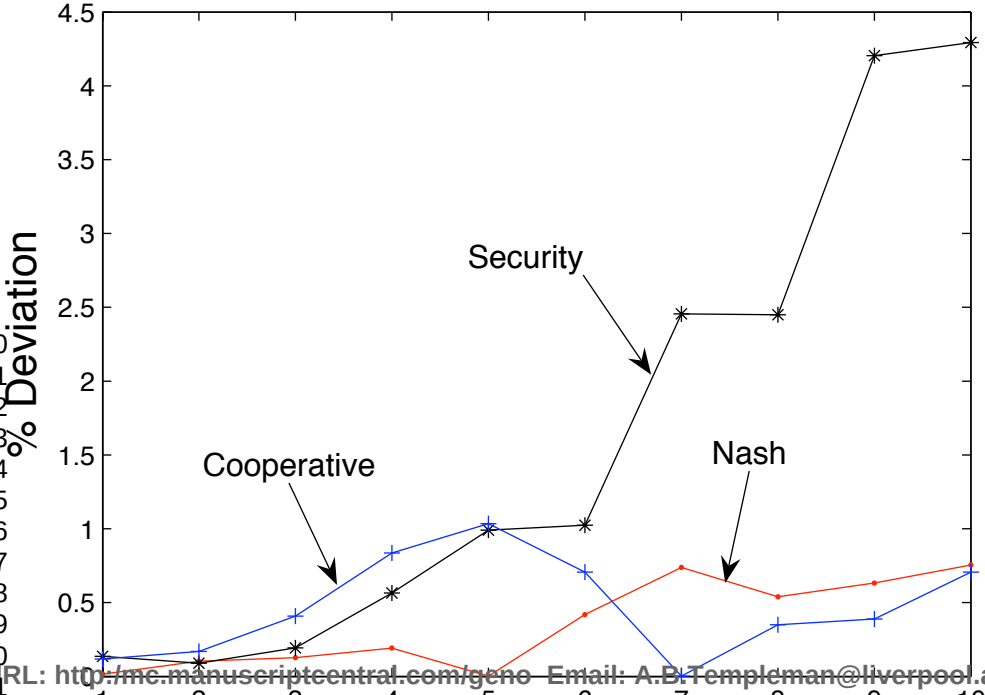
Algorithm A1 Dijkstra shortest path algorithm

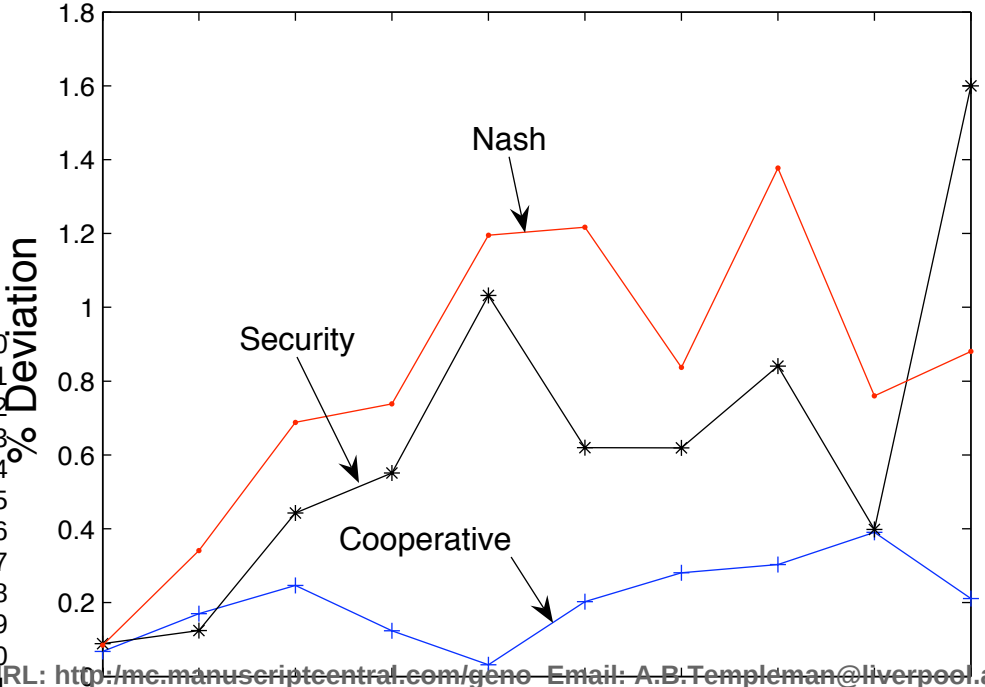
Initialization:
Set $S = \emptyset$ and $\bar{S} = V$
$d(i) = \infty$ (Actually a very large number) for each node $i \in V$ where $d(i)$ is a label that will ultimately store the distance of the shortest path from s to the node i.
$d(s) = 0$ and $pred(s) = 0$.
Main Loop:
while $|S| < n$ **do**
  let $i \in \bar{S}$ be a node such that $d(i) = \min\{d(j) : j \in \bar{s}\}$
  $S = S \bigcup i$
  $\bar{S} = \bar{S} \backslash i$
  for each $(i, j) \in E$ **do**
    if $d(j) > d(i) + C_{ij}$ **then**
      $d(j) = d(i) + C_{ij}$ and $pred(j) = i$
    end if
  end for
end while

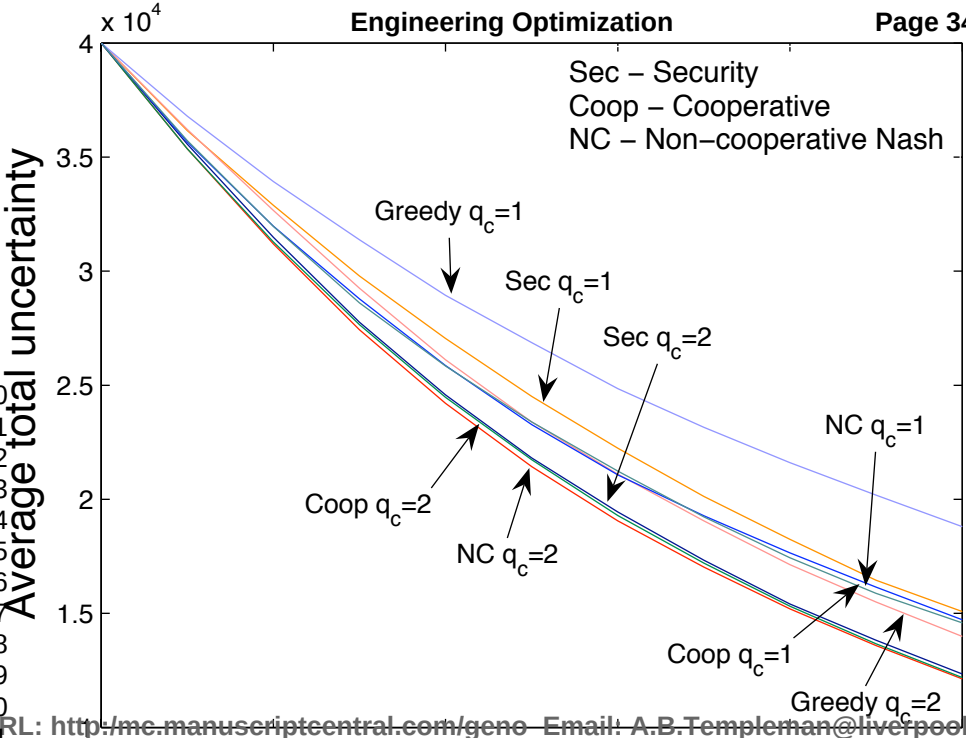

Average total uncertainty
Number of sorties
Coop q_c=1
Coop q_c=2
NC q_c=1
NC q_c=2
4
3.5
3
2.5
2
0
2
4
6
8
10

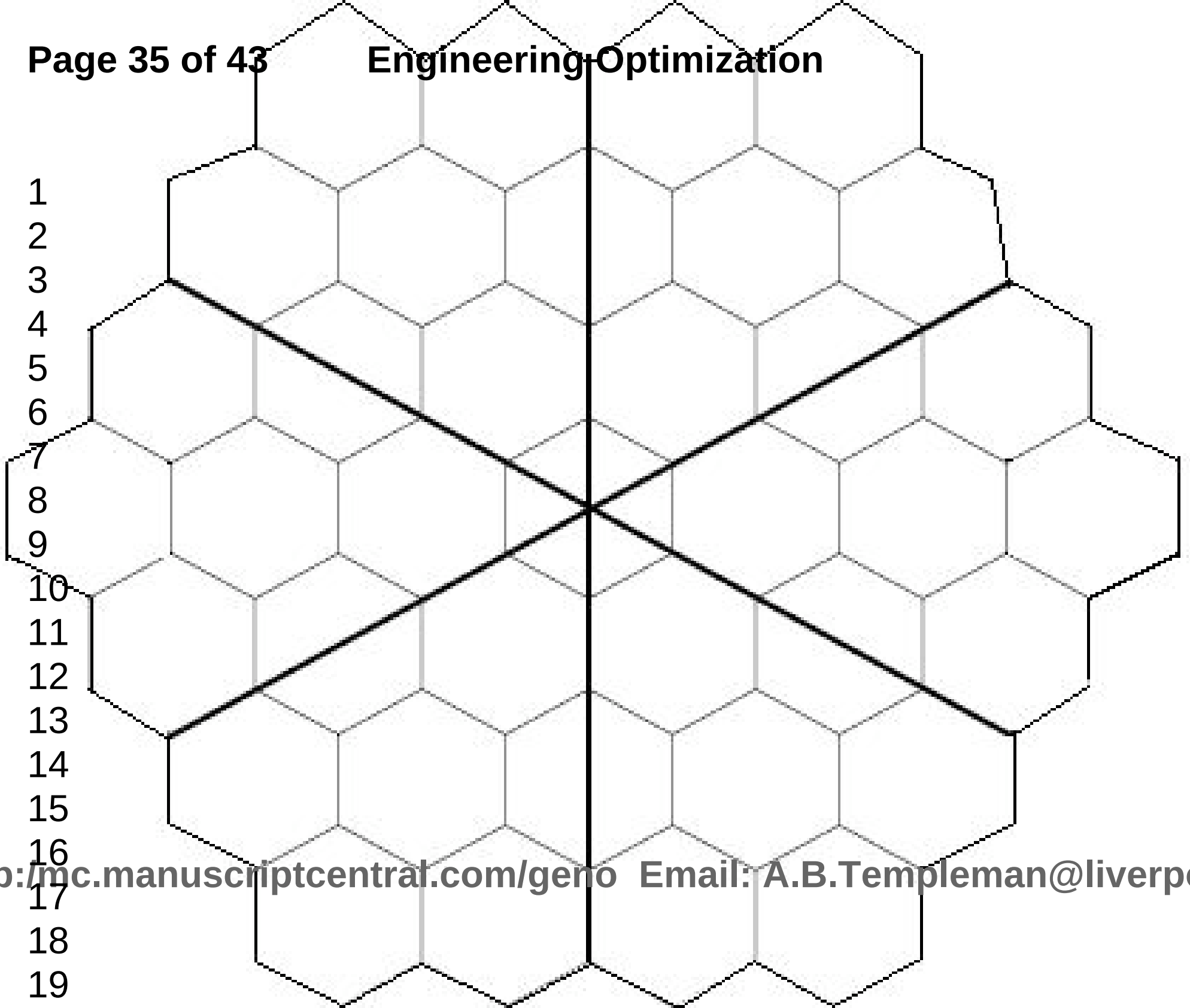

β
0.6
0.5
0.4
0.3
0.2
0.1
0
β1
β2
0
10
20
30
40
50
60
Time steps

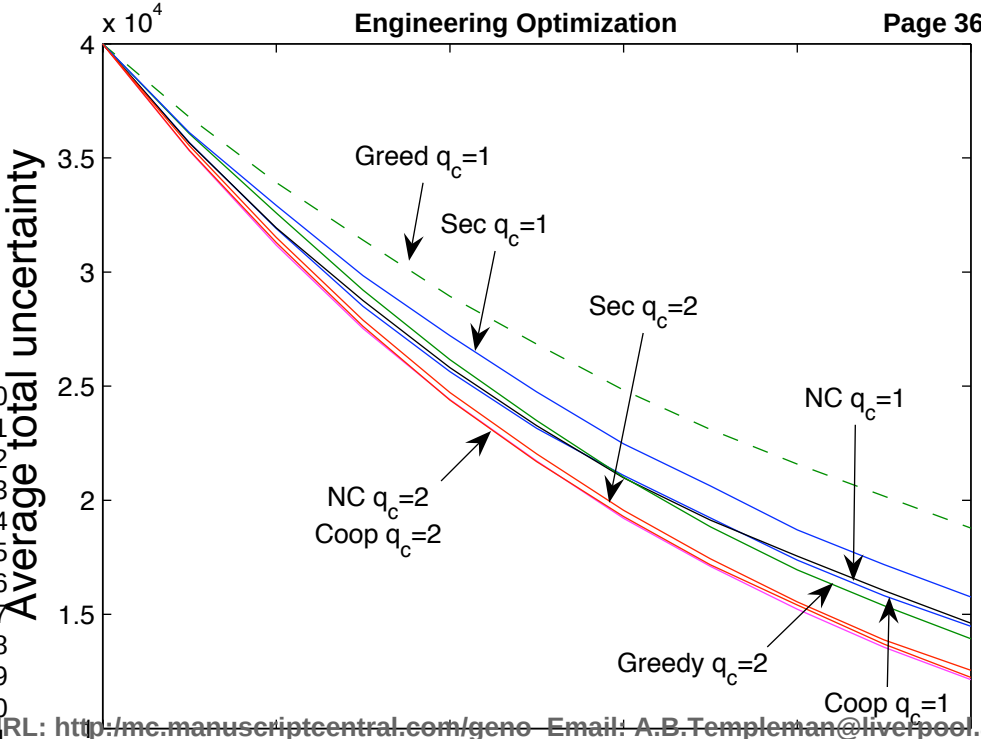

x 10^4
2
1.8
1.6
1.4
1.2
1
0.8
Average total uncertainty
0
50
100
150
200
250
Number of sorties
Nash strategy
Cooperative strategy

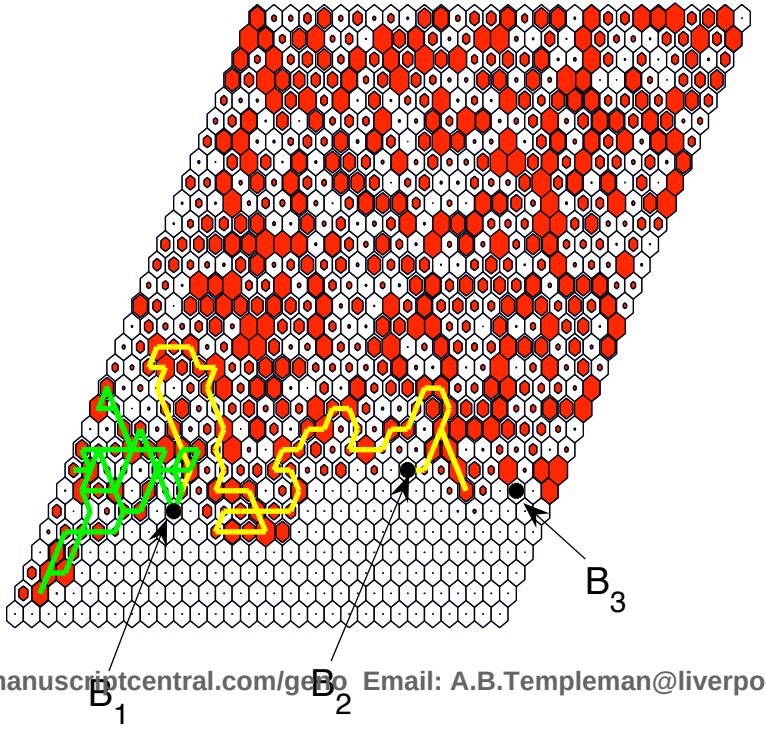


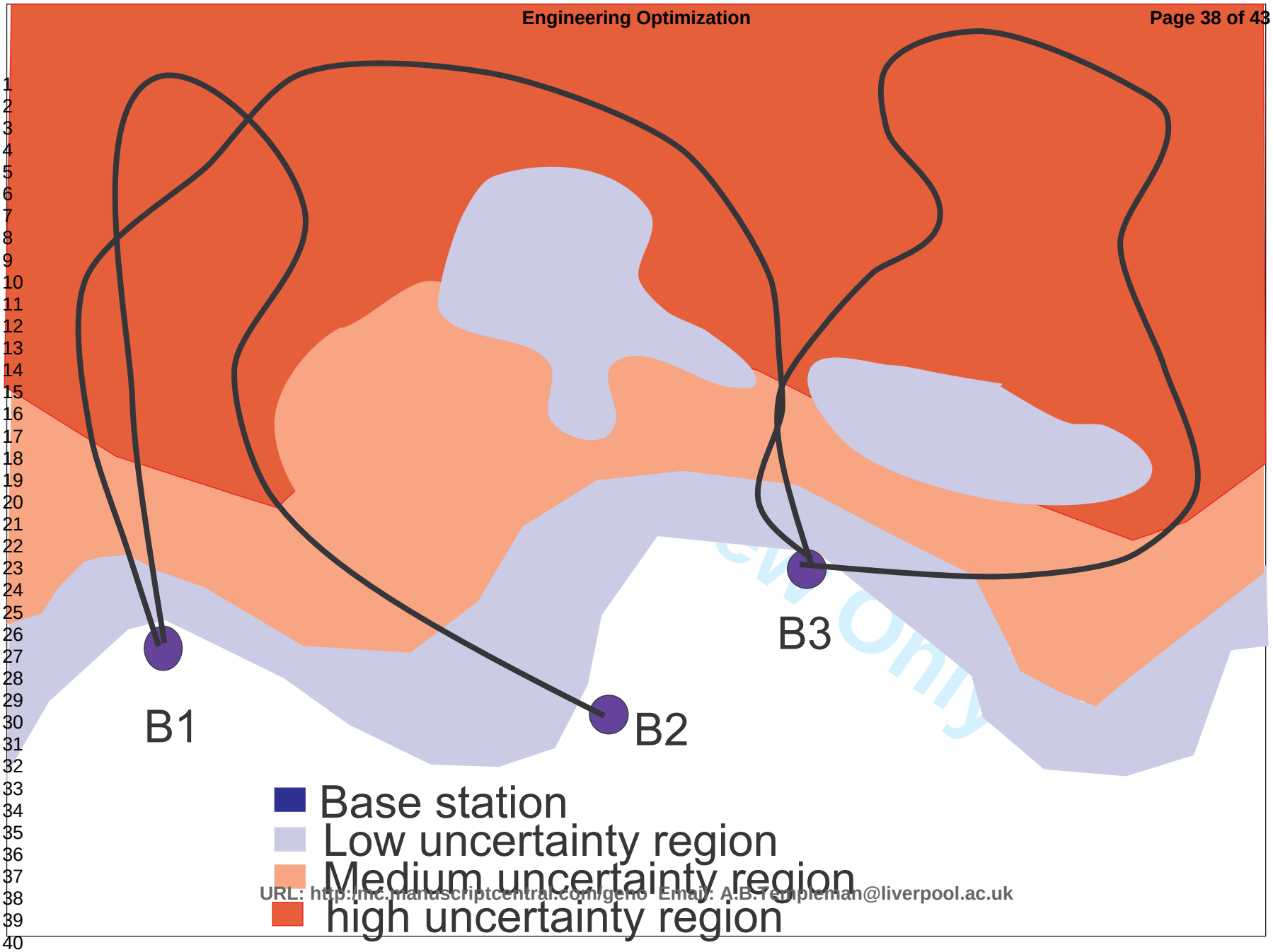


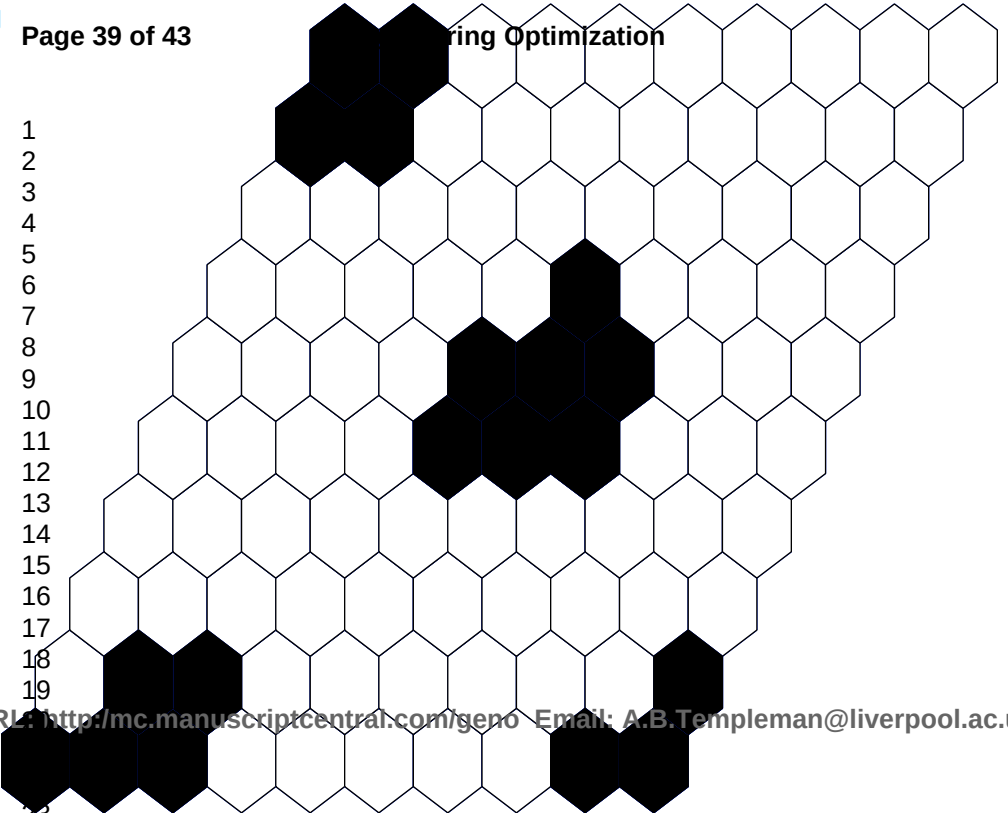

x 10^4
Sec – Security
Coop – Cooperative
NC – Non–cooperative Nash
Greedy q_c=1
Sec q_c=1
Sec q_c=2
NC q_c=1
Coop q_c=2
NC q_c=2
Coop q_c=1
Greedy q_c=2
Average total uncertainty
Number of sorties
4
3.5
3
2.5
2
1.5
0
2
4
6
8
10

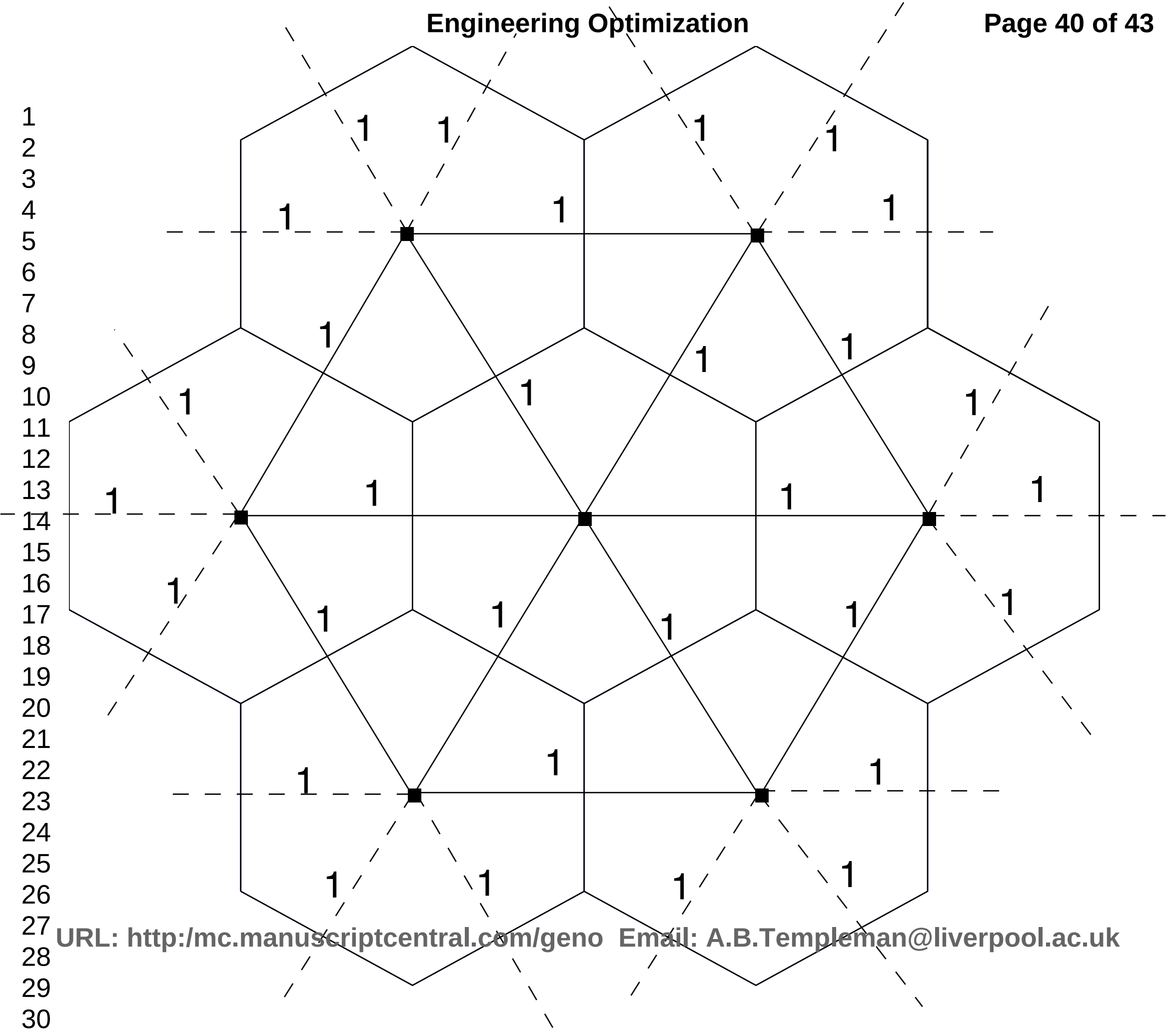





B1
B2
B3






1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1