2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Optimized Graph Topologies for Probabilistic Search

Christian Klaus and Timothy H. Chung

Abstract— **This paper investigates the effect on the performance of a mobile searcher caused by altering the search environment. We model the search environment as a simple connected undirected graph. By adding non-existing edges to the graph we change the search environment's model. Our objective is to optimize search performance, that is, to minimize the (expected) time needed to find the target, in the context of probabilistic search. We first analyze two different methods to generate random connected graphs, then evaluate a number of methods to augment a graph by means of the algebraic connectivity of the graph and its associated (Fiedler) eigenvector. The relationship between the graph topology and the performance of the search is highlighted, including a comparative evaluation of different search strategies employed by the mobile searcher. Extensive simulation studies and resulting statistical and theoretical models show that adding a few wisely chosen edges to a sparse graph is sufficient to dramatically increase search performance. Further, we propose a novel method for incorporating prior information about the target's likely location by defining a subgraph on which the presented approach is performed, resulting in even greater improvements in search performance.**

I. Introduction

The Laplacian matrix of a graph and its eigenvalues are used in various areas of mathematics with interpretations in several problems in physics, electrical engineering and many others. Among all eigenvalues of the graph Laplacian one of the most extensively studied is the second smallest, including seminal work by M. Fiedler [1],[2]. In terms of graph theory this eigenvalue is denoted the *algebraic connectivity* of the graph. Because of its importance and in honor of Fiedler's work, the eigenvector associated with this eigenvalue is known as the Fiedler vector. Classifications of bounds relating other graph properties as a function of algebraic connectivity are shown in [3] and more recently summarized in [4].

While the way of assigning edge weights to graphs in order to maximize the algebraic connectivity has been widely studied and applied to various problems [5], [6], [7], the problem of adding some non-existing edges to achieve the same goal remains challenging. In fact, this problem is shown to be NP-hard [8]. In this research area, [9] is of particular interest. Boyd and his co-authors identified and provided a common framework for a class of these edge-weight problems using a semidefinite program (SDP) [10]. In [11], they introduce a relaxation to the boolean constrained semidefinite program as well as an approach employing the Fiedler vector to guide the augmentation of the graph. While the relaxed SDP does provide an optimal non-integer solution, which represents an upper bound for the augmented graphs algebraic connectivity, it does not directly yield a solution to the original problem in that the simple rounding of the edge weights makes this a heuristic method. Neither the Fiedler vector method nor the relaxed SDP guarantees an optimal solution. In the case of growing a graph by only one edge, [12] defines a method that uses the Fiedler vector and provides the optimal solution.

C. Klaus is with the Operations Research Department, Naval Postgraduate School, Monterey, CA, USA cklaus@nps.edu

T. Chung is with the Department of Systems Engineering, Naval Postgraduate School, Monterey, CA, USA thchung@nps.edu

Probabilistic search theory has been extensively studied in operations research communities, stemming from the application of probability and optimization models in search of submarines [13]. Numerous works have considered optimization of searcher paths in both the OR (see [14] for a survey) and robotics communities (e.g., [15]), but few have investigated the structure of the search environment and its modification as they pertain to the probabilistic search process. The connection between algebraic connectivity and search performance has been studied in [16], leveraging a framework for decision making in probabilistic search found in [17]. To the best of our knowledge, this is the first work that provides insight in the relation between augmenting a given connected graph and probabilistic search performance.

We consider a simple connected graph without multiple edges or loops, $G = (V, E)$ with $n = |V|$ nodes (vertices, cells) and $m = |E|$ edges. The complement of the graph $\overline{G} = (V, \overline{E})$ contains $\overline{m}$ edges s.t. $e_{ij} \notin E(G)$. $G(V, E \cup \overline{E}) = K_n$, the complete graph with $\frac{1}{2}n(n-1)$ edges. The number of neighbors of a node v is called the degree of v and is denoted by $deg(v)$. Furthermore, the largest degree in the graph is denoted by $\Delta(G)$ and the smallest degree by $\delta(G)$. A simple connected graph with n vertices has at least $n-1$ edges.

The ij-entry of the $n{\times}n$ adjacency matrix, Adj_{ij}, is one if $e_{ij} \in E$ and zero otherwise. The degree matrix, Deg, is defined by $Deg_{ii} = deg(v_i)$ and zero for $i \neq j$. The edge vector, $a_e \in \mathbb{R}^n$, corresponding to the edge, e_{ij}, has two nonzero entries, $a_{e_i} = 1$ and $a_{e_j} = -1$.

The $n{\times}n$ graph Laplacian can be written as

$$L = Deg - Adj = \sum_{e=1}^{m} a_e a_e^T$$

The *algebraic connectivity*, $\lambda_2(L)$, has the following properties [1]:

- λ_2 is monotonically non-decreasing with adding edges to the graph G,
- $\lambda_2 = 0$ if and only if G is not connected,
- $\lambda_2 = m$ if $G = K_n$,
- $\lambda_2 \leq \delta(G)$.

The main contributions of this work include a comparative evaluation of methods for adding edges to improve the topology of a graph. The resulting approach is used to highlight the relationship between the graph structure and the performance of a mobile searcher as measured by the time until a stationary target is found. Statistical studies provide the basis for an analytic model which describes an exponential dependence of search performance on the number of edges added to the graph. Further, a novel approach to incorporate prior information on likely target locations using a subgraph is proposed.

The problem of augmenting a graph with additional edges is formulated in Section II, in addition to the models for probabilistic search. Section III offers analysis of simulation studies and provides resulting analytic models describing search performance. Finally, a summary of results and multiple avenues for future work are highlighted in Section IV.

II. Problem Formulation

A. Optimization Problem

Because of the demonstrated relationship between algebraic connectivity and search time [16], we are going to grow the graph by k edges, in order to maximize $\lambda_2(G)$. We briefly review three methods for augmenting the graph.

1) Brute Force: This method simply tries all possible combinations of edges to add and thus is an exact algorithm, i.e., it is guaranteed to find the optimal solution. Unfortunately, it takes $\binom{|\overline{E}|}{k}$ eigenvalue computations. We use this method for benchmarking purposes for cases with small k but is clearly infeasible for larger problems of practical interest.

2) Fiedler vector method: This method (FVM) uses the Fiedler vector, i.e., the eigenvector associated with the second smallest eigenvalue of the original graph, to determine an edge to add. It was first introduced as a greedy perturbation heuristic in [11]. Each of the k edges are added sequentially, that is, one at a time, choosing the edge e_{ij} from the candidate set of edges currently not in the graph with the largest $(v_i - v_j)$, where v is the Fiedler vector of the current graph Laplacian. We need to perform k eigenvector computations to find k edges to add. This method does very well in many cases but does not guarantee an optimal solution, neither for adding one edge nor for the sequence of edges.

3) Semidefinite programming: In his 1973 paper [1], Fiedler already stated and showed that $\tilde{M} = M - \lambda_2\left(\mathbf{I} - n^{-1}\mathbf{J}\right)$ is semidefinite for any $n{\times}n$ semidefinite matrix M with corresponding second smallest eigenvalue λ_2, where I is the identity matrix and J the matrix of all ones. The graph Laplacian L with $\lambda_i \geq 0,\ i = 1, 2, \ldots, n$ is always positive semidefinite. Boyd used the semidefinite property to formulate a relaxation of the original semidefinite program in [11] to maximize the algebraic connectivity by adding edges to the graph.

$$\begin{aligned}
\max \quad & s \\
s.t. \quad & s\left(\mathbf{I} - \mathbf{1}\mathbf{1}^T/n\right) \preceq L + \sum\nolimits_{e=1}^{\overline{m}} x_e a_e a_e^T \\
& \mathbf{1}^T x = k, \\
& 0 \leq x \leq 1
\end{aligned}$$

The decision variable is $x \in \{0,1\}^{\overline{m}}$, where $x_e > 0$ defines the fraction of edge e that belongs to the solution. The integer number of edges to add is smeared over all the possible edges to add. To solve the original problem, this method requires a heuristic to determine which edge to choose from the SDP solution. The authors in [11] propose a simple rounding to find the k best edges, which is not guaranteed to be optimal.

B. Probabilistic Search

The primary question in search is to determine where the target is located in a region A, given uncertainty in the target's location and in the searcher's observations. If the target is considered stationary and is known to be located somewhere in the region we have a good chance to answer the question.

Consider a discretized search region A, with cells $c \in \{1, \ldots, |A|\}$. In addition we define a virtual cell $\emptyset$ that represents all space outside the search environment. A target can be either in one of the search region cells or in the virtual cell. The presence in one of those cells is a Bernoulli random variable, denoted as X_c or $X_\emptyset$. At the beginning of the search process (time, $t = 0$) there may be a particular probability, that the target is inside the c^{th} search cell, $p_c^0 \overset{\text{def}}{=} P\left(X_c = 1\right)$ or outside the search region, $p_\emptyset^0 \overset{\text{def}}{=} P\left(X_\emptyset = 1\right) = 1 - \sum\limits_{c=1}^{|A|} p_c^0$.

The single searcher's position will be in one of the search region's cells at any time. We can define another Bernoulli random variable $Y_{s(t)}$ that determines whether the searcher detects the target in cell $s(t)$ at time t.

Besides correct detections the searcher can make false detections:

$$\begin{aligned}
P\left(Y_c = 1 \middle| X_c = 0\right) &\overset{\text{def}}{=} \alpha \qquad \text{false positive} \\
P\left(Y_c = 0 \middle| X_c = 1\right) &\overset{\text{def}}{=} \beta \qquad \text{false negative}
\end{aligned}$$

Applying the law of total probability and Bayes' Theorem:

$$P(B) = \sum_{i=1}^{k} P(B|A_i)P(A_i)$$

$$P(A_j|B) = \frac{P(A_j \cup B)}{B} = \frac{P(B|A_j)P(A_j)}{\sum\limits_{i=1}^{k} P(B|A_i)P(A_i)} \quad j = 1 \ldots k$$

we can update the probabilities conditioned on the search result after each search step. As an example, the probability or *belief* that the target is present in cell c at time t (i.e., after t observations) is given by the recursion:

$$p_c^t = (X_c^t = 1 | Y_c^t = 1) = \frac{(1-\beta)p_c^{t-1}}{(1-\beta)p_c^{t-1} + \alpha(1 - p_c^{t-1})}$$

Depending on the false detection probabilities, α and β, it takes more or less effort (that is, the number of observations) to find the target, and time increases to infinity if our initial belief of the target's presence $(1 - p_\emptyset^0)$ is incorrect and the target is not located inside the search region. We define a lower B_l and upper B_u threshold and stop searching at time t if for any cell $p_c^t > B_u$ or if $1 - p_\emptyset^0 < B_l$.

This defines a search termination criterion studied in other contexts, including [18], [19].

Fig. 1 shows the aggregate belief history, that is, the sum belief that the target is present in search area A, for a search on a random graph with $|A|=n=50$ and $m=60$ for searchers equipped with different sensors. Target location and searcher position at $t=0$ are the same. We do not know if the target is present in A at the beginning of the search process, which is reflected in an aggregate belief of $\frac{1}{2}$. Given the target is present, suppose it could be located in any cell with equal likelihood, which corresponds to the searcher's lack of any prior information. Thus, the initial cell belief in each cell is $p_c^0 = \frac{1}{100}$. We say the initial probability map is uniform with belief $\frac{1}{2}$. The remaining 50 percent of the probability mass is assigned to the virtual cell, $p_\emptyset^0 = \frac{1}{2}$. At this point we refer to [16] for detailed information, including closed-form recursion expressions and further studies on sensitivity of the search performance on sensor characteristics (α, β).

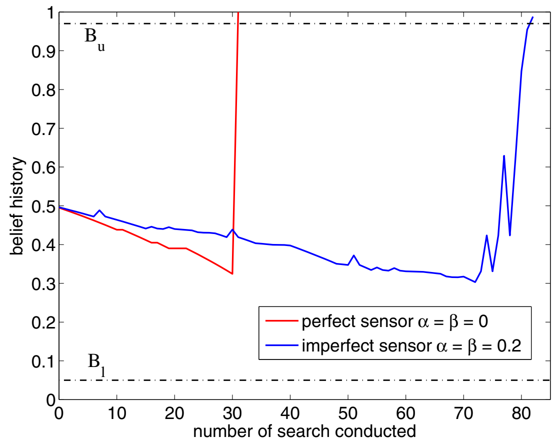


Fig. 1. **Belief of target presence** - Evolution of two searches on a graph with n=50 and m=60 with different sensor characteristics are illustrated. The initial probability map is assumed uniform with aggregate belief $\frac{1}{2}$. Target is assumed stationary at the same location for both searches as well as the initial searcher's position. The search decision thresholds are set to be B_l=0.05 and B_u=0.97.

III. Analysis and Results

A. Random Graph Generation

In order to evaluate the proposed methods for augmenting a graph in a statistically rigorous manner, we require methods for quickly generating random simple connected graphs, including those with large node number. We investigate and contrast the following two methods, in which we find the former is faster in creating random graphs, but produces graphs with higher maximum degree values, Δ, which can bias the resulting search performance. The latter approach is seen to mitigate this bias and offers a more uniform sample set of random graphs on which to perform the statistical studies for optimizing the algebraic connectivity, and thus the search performance, of the resulting graphs.

1) The Spanning Tree Method: Every connected graph has a spanning tree, a subgraph containing n nodes and $(n-1)$ edges that is connected. Conversely, a graph having such a spanning tree must be connected. We start with an empty graph and mark the nodes as available. We choose an available node and an already connected node, place an edge between those and mark the available nodes as connected. After repeating this n times we have found a spanning tree and can place more edges to the graph without destroying the connectivity.

This is a fast algorithm and we can produce graphs with a specific number of edges. The downside is that the first chosen nodes to become connected are more likely to end up with higher degree. They get more chances to be chosen from the already connected node set. Simulation shows that this effect is rather influential for sparse graphs and vanishes with increasing number of edges.

2) Acceptance-Rejection Method: Suppose we want to generate a connected graph with n nodes and m edges. The number of possible edges is $\frac{n(n-1)}{2}$. The algorithm draws a binomial random variable with probability of success $p = \frac{2 \cdot m}{n(n-1)}$ for any of those possible edges. It adds an edge to the edge set $E(G)$ if the coin flip succeeded. In a second step we have to check if the graph is connected, an easy task calculating $\lambda_2(L)$ and we have to check if the number of actual edges is in the desired range.

This method is unbiased in producing random connected graphs but it can take many trials to finally find a graph that is accepted, especially for sparse graphs with a larger number of nodes.

B. How to Grow the Graph

Knowing the algorithm presented in [12] that always finds the optimal edge to add in order to increase $\lambda_2(L)$, we analyze if we could get some advantage from this fact. We grow a graph sequentially by putting the optimal edge for the current state and run brute force to find the optimal solution for the same number of edges. Sequential adding of current state optimal edges is just another heuristic method without guarantee to return optimal solutions.

We compare the results from the Fiedler vector method (FVM) (sequential) to the SDP result (simultaneous). Utilizing brute force, we can rank all missing edges according to the resulting value of λ_2 if added. We assess which edge is chosen by FVM and SDP and match it to the ten best optimal edges. A plot is shown in Fig 2. We have not incorporated the fact that the SDP is capable of finding more than one edge simultaneously with one calculation. We now add more than one edge to graphs with different edge densities. The result is shown in Fig. 3 This analysis shows that the greedy algorithm, that is, the sequential Fiedler vector method, outperforms the SDP approach for random graphs of varying sizes. The algebraic connectivity of the grown graph, though not optimal, is better than a SDP result after rounding to integer values. For each edge that we want to add to a graph, we must compute the graph Laplacian. The eigenvalue and eigenvector calculation takes approximately $O(\frac{4}{3}n^3)$ operations. To lower the amount of computational effort, we also propose a method for selecting more than one edge, that is, a block of edges, after each Fiedler vector calculation. Fig. 4 illustrates the use of block sizes of three

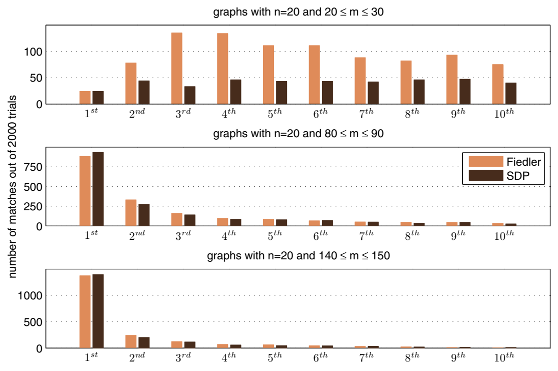


Fig. 2. **Fiedler vs. SDP, add one edge to a random graph** - The edges found by the Fiedler vector method and by a semidefinite program were matched to the ten best known edges (calculated by brute force). The plot shows the number of times a method hit a specific edge. The simulation was repeated 2000 times for each different edge density with $n = 20$.

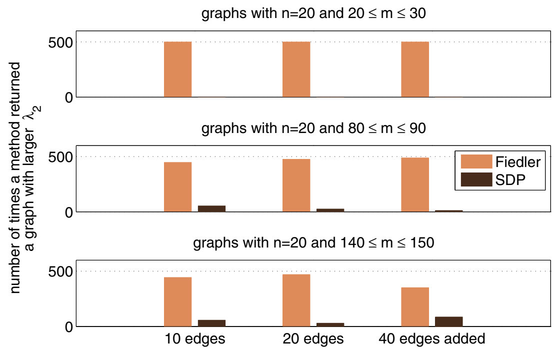


Fig. 3. **Fiedler vs. SDP, add more than one edge to a random graph** - A random graph was grown by a certain number of edges using the Fiedler vector method and a SDP. The plot shows the number of times a method returned a graph with higher λ_2 than the other method. The simulation was repeated 500 times for each different edge density and each different number of edges to add.

or ten edges, found by computing the three or ten pairs of valid nodes (represented by elements in the Fiedler vector) yielding the greatest differences. The algebraic connectivity for augmented graphs using this approach is also compared with the nominal Fiedler vector method (i.e., with block size 1) and random selection of a valid edge to add.

C. Search on Augmented Graphs

Let us grow a randomly generated graph and analyze the influence on the time to find the target or to abort the search. Consider a set of randomly generated graphs, initially sparse with $n = 20$ and $m \leq 50$. We can use the greedy algorithm based on the Fiedler vector using a block size of three edges for each iteration. We can repeat this augmentation up to 46 times (for $m = 50$) before the resulting graph is nearly a complete graph on 20 nodes, K_{20}. For each iteration, we conduct 2000 replications of the search process for the stationary target, which is assumed uniform among all nodes. For all runs, the searcher possesses a perfect sensor $(\alpha = \beta = 0)$ and behaves myopically, that is, at time t the searcher moves to the adjacent cell with highest probability p_c^t. Note that the current search cell is considered self-adjacent, which allows the searcher to repeatedly search the same location if necessary. If the searcher can choose

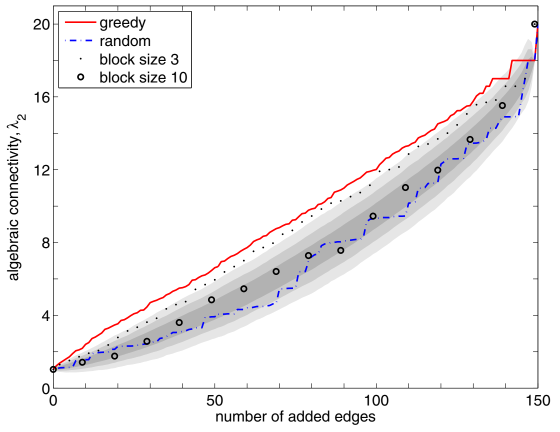


Fig. 4. **Different methods to grow a graph** - The initial random simple connected graph with $n = 20$ and $m = 40$ grew to a complete graph using different methods to choose the next edge or block of edges respectively. The shaded areas represent one, two and three standard deviations for the randomly grown graph (1000 replications used).

from more than one equally likely cell to move to, it will choose arbitrary. The plot in Fig. 5 shows the results for an illustrative example graph starting with $m = 27$ edges. The expected number of time steps till the target is found, i.e., the mean of the 2000 search times for every iteration of augmenting the graph can be seen to decrease with increasing number of edges in the graph. For comparison, we grow the same graph 46 times with three randomly chosen edges and compare the algebraic connectivity and search times for the graphs resulting from these random additions, noting that both approaches will eventually achieve the known upper bound for $\lambda_2 = n$.

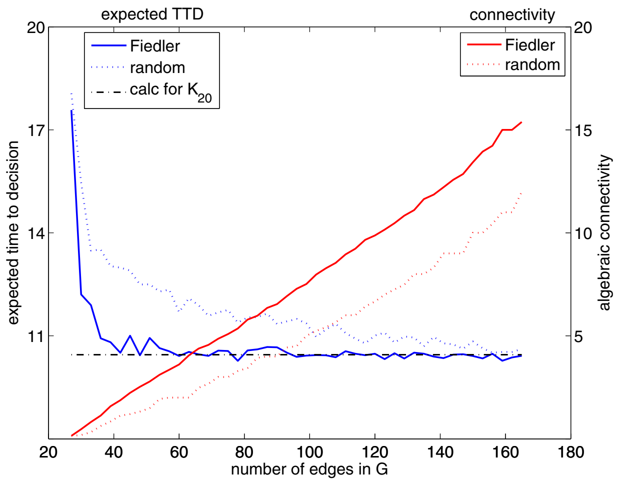


Fig. 5. **Change of TTD with adding edges by greedy algorithm** - The initial graph with $n = 20$ and $m = 27$ was grown in steps of three edges by the greedy algorithm. 2000 myopic searches were conducted for each growing state. The searcher was considered perfect and the target was located uniformly. The same graph was grown by the same number of edges randomly chosen. Searcher and target properties stayed the same. We included the calculated expected number of steps for a K_{20}.

The time to find the target levels off and hits a minimum value for a complete graph, for which we derive a closed-form expression for the expected time and its variance as follows. The myopically-behaving searcher will visit the nodes of a complete graph of n nodes, K_n, according to the probability $\{p_s^0, p_{1*}^0, \ldots, p_{k*}^0\}$, where s is the start cell, $p_{1*}^0 \geq p_{2*}^0 \geq \ldots \geq p_{k*}^0$ and $k^*{=}n{-}1$. If the searcher is perfect, it will never visit the same cell twice. We use this information to calculate the expected number of cell searches, $E[S_{n,\frac{1}{n}}]$ on a K_n by means of a recursion where the target is located in any cell with uniform probability $\frac{1}{n}$.

$$E[S_{n,\frac{1}{n}}] = \begin{cases} 1 & ,p = \frac{1}{n} \\ 1 + E[S_{(n-1),\frac{1}{n-1}}] & ,p = 1 - \frac{1}{n} \end{cases}$$

For a K_2, we can search one of the two cells to know the target's location, provided the target is present, since the result from a single observation will either declare the target present in the observed cell (for a positive detection) or in the other cell (for a negative detection). Setting the initial conditions $E[S_{2,\frac{1}{2}}] = 1$ and solving the recurrence gives the closed form:

$$E[S_{n,\frac{1}{n}}] = \frac{\sum_{k=2}^{n} k}{n} = \frac{n\,(n+1)-2}{2n}$$

The same idea can be used for any initial nonuniform target distribution, conditioned on the target's presence.

$$\begin{aligned} E[S_{n,\,p_n}] &= p_s^0 + \sum_{i=1}^{n-2}(i+1)\cdot p_{i^*}^0 + (n-1)\cdot p_{k^*}^0 \\ Var[S_{n,\,p_n}] &= (1 - E[S_{n,\,p_n}])^2 \cdot p_s^0 \\ &\quad + \sum_{i=1}^{n-2}(i+1-E[S_{n,\,p_n}])^2 \cdot p_{i^*}^0 \\ &\quad + (n-1-E[S_{n,\,p_n}])^2 \cdot p_{k^*}^0 \end{aligned}$$

To validate this result for all graphs, we repeat the simulation several times for different numbers of nodes. Fig. 6 and 7 are two examples studied extensively in simulation.

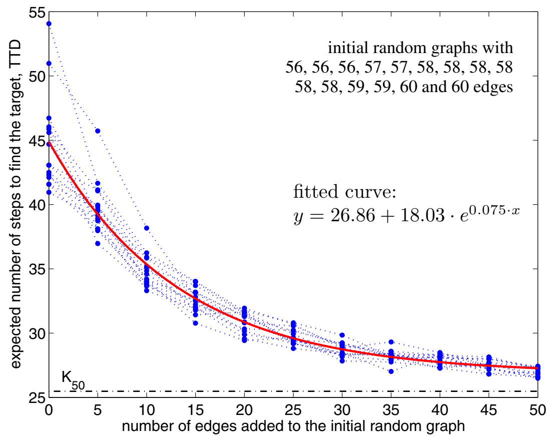


Fig. 6. **Conducting searches on 15 growing sparse graphs with 50 nodes** - The initial graphs with $n = 50$ and $55 \leq m \leq 60$ were grown in steps of five edges by the greedy algorithm. 2000 myopic searches were conducted for each growing state. The searcher was considered perfect and the target was located uniformly. We included the calculated expected number of steps for a K_{50}.

We can fit an exponential curve of the form $a + b \cdot e^{c \cdot x}$ to the simulation data. Parameter a is the expected time to decision on the complete graph K_n and b turns out to be consistently about half of that value. Additional studies are necessary with graphs of different node numbers to develop a statistical relation for the parameter c, and is left for near term future study. Using this fitted curves equation we can estimate the number of edges one must add to a sparse graph in order to increase search performance by a desired amount. The denser the initial graph, the more edges are needed to get a significant decrease in expected time to find the target, demonstrating the effect of diminishing returns by additional edges. The actual number of edges to add is seen to be dependent on the number of nodes in the graph, though common among all graphs, independent of number of nodes, is the fact that at some point, there is no further improvement with the addition of more edges.

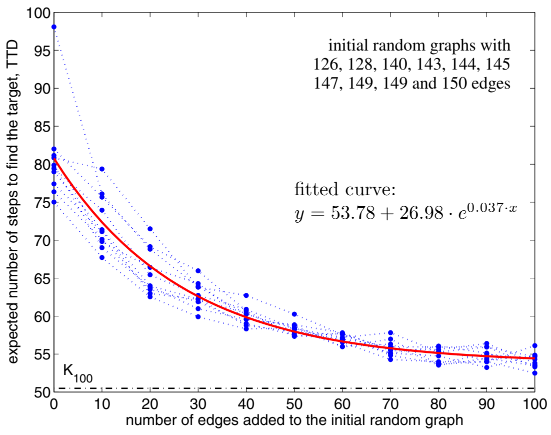


Fig. 7. **Conducting searches on 10 growing sparse graphs with 100 nodes** - The initial graph with $n = 100$ and $120 \leq m \leq 150$ were grown in steps of ten edges by the greedy algorithm. 2000 myopic searches were conducted for each growing state. The searcher was considered perfect and the target was located uniformly. We included the calculated expected number of steps for a K_{20}.

To illustrate the value of the fitted model, consider that for a sparse graph with $n{=}50$ (refer to Fig. 6), one can add just ten edges to decrease the search time by *half* relative to the lowest possible time, i.e., the theoretically derived lower bound. In other words, we gain 50% of the possible search improvement by adding just 0.857% of the total possible edges available for addition! The analytic formulation that captures this relationship is one of the main contributions of this work.

D. Other Search Behavior

The already described myopic search behavior offers a simple method without much computational expense. A shortest path behavior, leveraging global information over merely local data, seems to offer advantages if the initial target distribution is other than uniform. For the proposed alternate search strategy, a shortest path algorithm, such as Dijkstra's, dictates that the searcher follows the shortest route to search the highest probability cell as soon as possible. During transit from the current cell to this highest probability cell, the searcher can be assumed to either collect observations en route or not conduct any searchers at all. We can easily find practical applications for both behaviors, for example, the use of a towed sonar array (for observations en route) or the use of a dipping sonar (for observations only at the goal location).

Simulations show that search using these shortest paths perform worse than using the myopic search behavior. Since the searcher does not care about cells via transit while calculating the next search leg, it may visit previously searched cells. As such, the searcher often spends too much time in transit and possibly searching low probability cells.

Another effect that increases search time is an alternating search of adjacent search cells. The idea was to avoid revisiting of search cells after a short time. We coded an aggressive myopic search algorithm that searches a cell as long as one of its neighbors has a lower probability. The searcher will leave the cell after its probability is squeezed to the lowest among the adjacent cells. This becomes the threshold to leave the next cell. That means we have to push down the next cell's probability even further, which takes more search time. The result of this behavior is to increase the time to decision dramatically, especially in the event the target is not present in the search area to begin with.

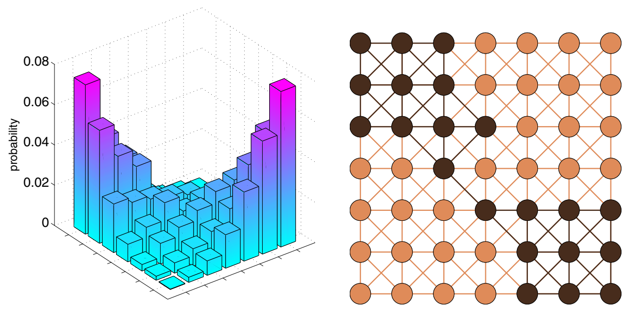
Though far from exhaustive, the study of different search strategies identifies the myopic search strategy as providing the best performance among them, where utilizing just local information rather than global information offers the highest "bang for the buck."

E. Incorporating Probability Information

The initial uniform target distribution is a suitable model whenever we do not have information about the target's location within the search area. While the previously described greedy algorithm for selecting edges to add did so among all candidate edges in a uniform manner, incorporating the probability information to guide the augmentation of the graph could improve the search performance even more. Intuitively, for nonuniform target distributions, the searcher is more likely to find the target in regions of higher target probability, and therefore should stay in and search these higher probability areas more frequently. The following extension of the proposed work recognizes and incorporates the advantage of additional information, and represents another novel contribution.

We extract a subgraph from the original graph that contains only high probability cells and all corresponding incident edges. This subgraph is then augmented according to the procedure proposed in the previous sections (i.e., the Fiedler vector method) to optimize its algebraic connectivity. Consider an example of a 7-by-7 grid graph to illustrate this idea, as depicted in Fig. 8(b). In this example, there are two areas in the search environment where the target is most likely to be. We incorporate this information into the initial probability map, shown in Fig. 8(a) as a bimodal distribution. Selection of nodes to be included in the subgraph is done by finding the smallest set of nodes such that the sum of the nodes' cell beliefs represents a specified percentage of the total probability. This procedure amounts to generating an ordered list of nodes, from highest cell belief to lowest, and sequentially adding nodes to the subgraph until the given percentage is met. Fig. 8 shows the original graph and the induced subgraph, where the darkened nodes collectively represent 75% of the probability mass of the target's presence. The graph growing algorithm is performed on this subgraph, such that the algebraic connectively of the induced subgraph is maximized (e.g., for the allowable number of additional edges), after which we embed the augmented subgraph back into the original graph. By decreasing the time to find the target while searching on the augmented subgraph alone, this procedure necessarily decreases the time to find the target searching on the entire graph. The enhanced connectivity of the embedded augmented subgraph ensures that the searcher remains longer in the areas of higher probability, effectively prioritizing search in these regions. In the limit where the subgraph is augmented until it itself is complete, the behavior of the searcher in the overall graph is to exhaustively search the high probability cells before it leaves to search the parts of the overall graph with lower probability of containing the target.

Specification of the probability threshold that dictates the selection of nodes to form the subgraph is a tuning parameter, and is likely related to the entropy H of the initial target distribution. The higher the entropy, one can expect the larger the subgraph. For example, for a uniform distribution, since no node is preferential over any other node, i.e., it does not make any sense to exclude any nodes, the subgraph is G itself. Increasing certainty of the target's location results in a smaller induced subgraph, and thus reduces the total number of edges to add to this subgraph to achieve significant improvements. Empirical studies show that generating subgraphs comprising 80% of the total probability mass and growing these subgraphs generally produced improved search effectiveness. More detailed analyses of this threshold specification and its relationship to the search performance is subject of ongoing research.



(a) (b)

Fig. 8. (a) Probability map incorporating information about the target location, which is used to extract (b) a subgraph (darker colored nodes) from the initial 7-by-7 grid graph which represents 75% of the probability. The subgraph is augmented by the greedy algorithm before it gets merged back to the graph

IV. Conclusions and Future Works

This paper investigated the relationship between the addition of edges to a graph representing a search environment and improvement in the search performance, as measured by the time necessary to complete the search. This work leverages existing results pertaining to the second smallest eigenvalue of the graph Laplacian, also known as the algebraic connectivity of the graph, and performs comparison studies to identify a greedy method based on the Fiedler vector as an effective and efficient choice for augmenting a graph. Statistical simulation studies and analysis using randomly generated graphs validate this finding.

Further, this work enhances the understanding of the positive correlation between λ_2 and the search time until the target is found in problems of probabilistic search. In fact, search performance on sparse graphs can be significantly improved by adding only a few wisely chosen edges, and the relation between number of edges to add and the search time is modeled as exponential. This analytic model offers guidance on the necessary additional number of edges needed to achieve a specified improvement in search for a target. This result also highlights the fact that adding edges beyond a certain number does little to improve search performance, and that knowledge of the relationship between number of additional edges and enhancement of search will limit the expense of adding unhelpful edges.

Simulation studies also shows that it is sufficient for the searcher to utilize local information in conducting its adaptive search of the environment. Such a myopic search strategy is seen to outperform other approaches that may involve global information but are wasteful in forcing the searcher to transit through previously visited or low probability regions in the search environment.

This work also proposed a novel method for further enhancing the improvement in search performance by partitioning the regions of high and low probability of target presence. By extracting the former region as a subgraph and adding edges using the greedy method found to be most effective, this approach offers a means of incorporating prior information on the target's likely locations into the graph augmentation process.

There are numerous avenues for future research, including extensions to account for heterogeneous edge weightings, which can represent, e.g., physical distances between locations. Searcher trajectories which address the constrained paths will enhance the relevance of the proposed models to practical applications. Additional modifications to the model for realistic scenarios address the need to account for the time it takes to search a cell in addition to transit times, where a variant of the shortest path search algorithm investigated herein might become more attractive.

The inclusion of correct information pertaining to the target location is seen to be beneficial to the searcher; however, using incorrect information, either by misdirection or by mistake, is significantly detrimental to the search process. In other words, if the searcher erroneously attributes a low likelihood to the true target location, the myopic searcher will search for a long time until that cell is finally searched. However, the myopic searcher may have an advantage over a searcher with global information by restricting the search to local regions. Such advantages may further be accentuated in the case of mobile targets, which is another area of future study. Addition of edges may improve the searcher's ability to localize the target, but also hinders the search by providing greater access for a moving target to maneuver on the graph. Investigation of this trade off is subject of active and ongoing research efforts.

V. Acknowledgments

The authors would like to thank Dr. Nedialko Dimitrov for useful discussions and helpful feedback.

References

[1] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak Mathematical Journal*, vol. 23, no. 98, pp. 298–305, 1973.

[2] ——, "A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory," *Czechoslovak Mathematical Journal*, vol. 25, no. 4, pp. 619–633, 1975.

[3] B. Mohar, "Eigenvalues, diameter, and mean distance in graphs," *Graphs and Combinatorics*, vol. 7, no. 1, pp. 53–64, 1991.

[4] N. D. Abreu, "Old and new results on algebraic connectivity of graphs," *Linear Algebra and its Applications*, vol. 423, no. 1, pp. 53–73, 2007.

[5] A. S. Ibrahim, K. G. Seddik, and K. J. R. Liu, *Improving Connectivity via Relays Deployment in Wireless Sensor Networks*. IEEE, Nov. 2007.

[6] ——, *Connectivity-Aware Network Maintenance via Relays Deployment*. IEEE, Mar. 2008.

[7] Y. Wan, S. Roy, X. Wang, A. Saberi, T. Yang, M. Xue, and B. Malek, "On the Structure of Graph Edge Designs that Optimize the Algebraic Connectivity," in *Proceedings 2008. 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 805–810.

[8] D. Mosk-Aoyama, "Maximum algebraic connectivity augmentation is NP-hard," *Operations Research Letters*, vol. 36, no. 6, pp. 677–679, Nov. 2008.

[9] S. P. Boyd, "Convex optimization of graph Laplacian eigenvalues," in *Proceedings of the International Congress of Mathematicians*, vol. 3, no. 1-3. Citeseer, 2006, pp. 1311–1319.

[10] S. P. Boyd and L. Vandenberghe, *Convex optimization*, 7th ed. Cambridge University Press, 2004.

[11] A. Ghosh and S. Boyd, "Growing Well-connected Graphs," *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 6605–6611, 2006.

[12] Y. Kim, "Bisection Algorithm of Increasing Algebraic Connectivity by Adding an Edge," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 170–174, Jan. 2010.

[13] B. O. Koopman, "Search and Its Optimization," *The American Mathematical Monthly*, vol. 86, no. 7, pp. 527–540, 1979.

[14] S. J. Benkoski, M. G. Monticino, and J. R. Weisinger, "A Survey of the Search Theory Literature," *Naval Research Logistics*, vol. 38, no. 4, pp. 469–494, Aug. 1991.

[15] F. Bourgault, T. Furukawa, and H. F. Durrant-Whyte, "Optimal Search for a Lost Target in a Bayesian World," *Field and Service Robotics*, vol. 24, pp. 209–222, 2006.

[16] T. H. Chung, "On Probabilistic Search Decisions under Searcher Motion Constraints," in *Star*, ser. STAR, G. S. Chirikjian, H. Choset, M. Morales, and T. Murphey, Eds., vol. 57. Springer Berlin Heidelberg, 2008, pp. 501–516.

[17] T. H. Chung and J. W. Burdick, "A Decision-Making Framework for Control Strategies in Probabilistic Search," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 4386–4393.

[18] M. C. Chew, "Optimal Stopping in a Discrete Search Problem," *Operations Research*, vol. 21, no. 3, pp. 741 – 747, 1973.

[19] M. Kress, R. Szechtman, and J. S. Jones, "Efficient Employment of Non-Reactive Sensors," *Military Operations Research*, vol. 13, no. 4, pp. 45–57, 2008.