An Innovative Multi-Agent Search-and-Rescue Path Planning Approach

Jean Berger
DRDC Valcartier
Quebec City, Canada
jean.berger@drdc-rddc.gc.ca

Nassirou Lo
T-OptLogic Ltd.
Quebec City, Canada
nassirou.lo@t-optlogic.com

***Abstract* – Search and rescue path planning is known to be computationally hard, and most techniques developed to solve practical size problems have been unsuccessful to estimate an optimality gap. A mixed-integer linear programming (MIP) formulation is proposed to optimally solve the multi-agent discrete search and rescue (SAR) path planning problem, maximizing cumulative probability of success in detecting a target. It extends a single agent decision model to a multi-agent setting capturing anticipated feedback information resulting from possible observation outcomes during projected path execution while expanding possible agent actions to all possible neighboring move directions, considerably augmenting computational complexity. A network representation is further exploited to alleviate problem modeling, constraint specification, and speed-up computation. The proposed MIP approach uses CPLEX problem-solving technology in promptly providing near-optimal solutions for realistic problems, while offering a robust upper bound derived from Lagrangean integrality constraint relaxation. Modeling extension to a closed-loop environment to incorporate real-time action outcomes over a receding time horizon can even be envisioned given acceptable run-time performance. A generalized parameter-driven objective function is then proposed and discussed to suitably define a variety of user-defined objectives. Computational results reporting the performance of the approach clearly show its value.**

Keywords: search path planning, search and rescue, multi-agent, linear programming

I. INTRODUCTION

Search and rescue path planning is an increasingly important problem for a variety of civilian and military domains such as homeland security and emergency management. The basic discrete SAR or optimal searcher path problem involving a stationary target is known to be NP-Hard [1]. SAR may be generally characterized through multiple dimensions and attributes including: one-sided search in which targets are non-responsive toward searcher's actions, two-sided, describing target behavior diversity (cooperative, non-cooperative or anti-cooperative), stationary Vs. moving target search, discrete Vs. continuous time and space search (efforts indivisibility/divisibility), observation model, static/dynamic as well as open and closed -loop decision models, pursued objectives, target and searcher multiplicity and diversity. Early work on related search problems emerges from search theory [2], [3]. Search-theoretic approaches mostly relate to the effort (time spent per visit) allocation decision problem rather than path construction. Based upon a mathematical framework, efforts have increasingly been devoted to algorithmic contributions to handle more complex dynamic problem settings and variants [4], [5]-[7]. In counterpart, many contributions on search path planning may be found in the robotics literature in the area of robot motion planning [8] and, namely, terrain acquisition [9], [10] and coverage path planning [11],[12], [13]. Robot motion planning explored search path planning, primarily providing constrained shortest path type solutions for coverage [14], [15] problem instances. These studies typically examine uncertain search environment problems with limited prior domain knowledge, involving unknown sparsely distributed static targets and obstacles. Separate work on robot search algorithms is also referenced on the pursuit-evasion [16] theme although the nature and complexity of the problem are somewhat different. Recent taxonomies and comprehensive surveys on target search problems from search theory and artificial intelligence/distributed robotic control, and pursuit-evasion problem perspectives may be found in [17], [5], [18]-[20] and [16] respectively.

Exact problem-solving methods for sequential decision search problem formulations show computational complexity to scale exponentially. For instance, dynamic programming [5],[20],[7],[21] or tree –based search techniques [22], [23] may satisfactorily work under specific constraints and conditions but ultimately face the curse of dimensionality, showing poor scalability even for moderate size problem. A MIP-based approach/formulation has recently been proposed as well to solve a related constrained pursuit-evasion problem [24]. But problem attributes, constraints and complexity prove distinctive from target search, while the approach remains sub-optimal and problem-solving limited to small size problems. This paved the way to the development of efficient heuristic and approximate methods. Some early approaches simply reduce computational complexity by relaxing some hard constraints to keep the problem manageable. Methods inspired from search theory propose procedures mainly based on branch and bound [21], [7] or path finding A* types of techniques and variants. Despite the development of many heuristics and approximate problem-solving techniques for the SAR problem [5], [20], published procedures still deliver approximate solution and mostly fail to provably estimate real performance optimality gap for practical size problems, questioning their real expected relative efficiency.

In this paper, we propose a new exact mixed-integer linear programming formulation to optimally solve the multi-agent discrete search path planning problem for a stationary object. In the problem model, 'open-loop with anticipated feedback' refers to offline planning, while capturing information resulting from predicted agent observations (projected cell visit action outcome) as opposed to real feedback. Anticipated feedback augments pure open-loop formulations which simply ignore information feedback, while significantly improving solution quality, and mitigating computational complexity limitations traditionally associated with closed-loop problem formulations (e.g. dynamic programming, and partially observable Markov decision processes). This contribution aims at extending a single agent search path planning decision model [25] to a multi-agent environment in which feasible agent actions are further expanded to any possible neighboring move directions, while capturing anticipated feedback information resulting from possible observation outcomes occurring from projected path execution. In that setting, the open-loop with anticipated feedback information (observations) decision model involves *n* agents (searchers) with imperfect sensing capability (but false-positive observations -free) searching an area (grid) to maximize cumulative probability of success in detecting a target, given a time horizon and prior cell occupancy probability distribution. The model takes advantage of anticipated feedback information resulting from observations outcomes along the path to update target occupancy beliefs and make better decisions. A network flow representation significantly reduces modeling complexity (e.g. constraint specification) as well as implementation and computational costs. The new decision model relies on an abstract network representation, coupled to a parallel computing capability (e.g. using the CPLEX solver [26]) to gain additional speed-up. The novelty lies in a new exact linear model, and the fast computation of near optimal solutions of practical size problems, providing a tight upper bound on solution quality through Lagrangean programming relaxation. The computable upper bound constitutes an objective measure to fairly estimate and compare performance gap against various techniques. Computational results prove the proposed approach very efficient. Small computational run-time naturally enables open-loop model (with anticipated feedback) extension to a closed-loop formulation in which action outcomes from the previous episode may be explicitly incorporated in real-time to update target occupancy belief distribution. As a result, an updated solution can be dynamically computed, by periodically solving new problem instances taking advantage of feedback information (from real observation outcomes), over short rolling horizons. The idea is to readily exploit episodic feedback information whenever available. In that case, associated computational run-time corresponds to the time required to visit a cell. This way to embrace constructive dynamic planning in real time through inexpensive computational effort is largely preferable to dynamic programming techniques aimed at computing an exhaustive optimal policy, mapping suitable actions to any possible posterior states at a prohibitive computational cost. The proposed approach rather determines the best sequence of moves given the current state while updating the path solution resulting from partial path execution by repeatedly solving a new problem instance characterizing the follow-on state. Similarly, large time horizon problems can be solved efficiently, optimizing multiple problem instances over receding horizons.

The structure of the paper is organized as follows. Section II first introduces problem definition, describing the main characteristics of the open-loop search path planning problem with anticipated feedback. Then the main solution concept for the problem is presented in Section III. It describes a new mixed-integer linear programming network flow formulation combined with network representation to efficiently compute a near-optimal solution. The proposed CPLEX-based problem-solving technique and some implementation issues are then briefly reported in Section IV. Section V reports and discusses computational results depicting the value of the proposed method. Finally, a conclusion is given in Section VI.

II. PROBLEM

A. General Description

The discrete centralized search and rescue path planning problem involves a team of *n* homogeneous stand-off sensor agents searching a stationary target in a bounded environment over a given time horizon. From a search and rescue mission perspective, the goal consists in maximizing the cumulative probability of success in detecting a target within a given region. Represented through a grid, the search region characterizes an area defined as a set of cells *N*, describing possible target locations. Presumably occupying a single cell, the precise location of the target is assumed unknown. A prior target location probability density distribution for which cell occupancy probabilities sum up to one can be derived from domain knowledge. It reflects possible individual cell occupancy, defining a grid cognitive map or uncertainty grid. Should the target be located outside the search areas of interest, a special inaccessible, and invisible virtual cell would simply be added to the basic problem description to preserve the sum of probability property. The cognitive map constitutes a knowledge base describing a particular world state, including variables such as target occupancy belief distribution, time, agent position and orientation. An example of a cognitive map is illustrated in Fig. 1 at a specific point in time.

The duration of a cell visit or service time is assumed constant, specifying the period of each episode. Vehicles are assumed to visit different cell locations at the same time, and fly at slightly different altitudes to avoid colliding with one other. A search path solution consists in constructing an agent path plan selecting base-level control action to maximize target detection.

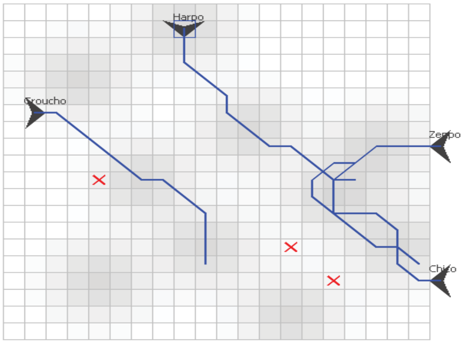

Figure 1. Uncertainty grid /cognitive map at time step *t*. The 4-agent team beliefs are displayed through multi-level shaded cell areas. Projected agent plans are represented as possible paths.

B. *Agent Path Planning*

Episodic agent search path planning decision is based on agent's position (cell location), specific orientation $\{N,S,E,W,NE,SE,SW,NW\}$ and speed determining possible legal moves to adjacent cell locations. For example, the 3-move agent investigated in [25] is limited to three possible moving directions with respect to its current heading, namely, *ahead, right* or *left* as depicted in Fig. 2. In this work, agent movement

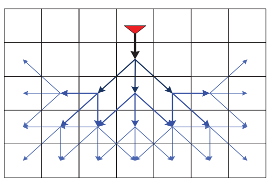

Figure 2. Agent's region of interest displayed as forward move projection span (possible paths), for a 3-move agent over a 3–step time horizon.

or manoeuvring capability is generalized to all degrees of freedom, permitting free motion along any possible directions to explore its neighborhood. An agent can therefore legally move toward its neighbouring cells offering eight alternate possible directions at each time step. This additional capability expands an agent path solution space by a factor $(8/3)^T$ over a 3-move planning agent for a given time horizon *T*, significantly increasing computational complexity.

The primary goal consists in planning base-level control action moves to maximize probability of success (target detection) over the entire grid.

C. *Cumulative Probability of Success*

In the proposed open-loop SAR model, the probability to successfully detect the target resulting from *n* agent path solution executions on the grid is defined as the sum over cells of the product of the probability of detection reflected from cell visits and target cell occupancy belief dictated by the cognitive map (grid) [5], [27], [28]. Cumulative probability of success (*CPOS*) for team path solutions (sequence of cell visits) represents the probability that detection occurs for the first time over one of the time intervals defining horizon *T*. It relates probability of first detection to binary observation outcome z_{ct} (1: positive, 0: negative) from cell *c* visit over time interval *t*, target cell *c* occupancy state X_c (1: positive, 0: negative) and past observation outcomes (history) $Z_{t\text{-}1}$ up to the end of interval *t*-1:

$$CPOS = \sum_{c\in N}\sum_{t} p(z_{ct}=1, X_c=1, Z_{t-1}=0) \qquad (1)$$

where $Z_{t\text{-}1}=0$ corresponds to a negative observation outcomes history $\{z_{c(0)0}=0, z_{c(1)1}=0, \ldots, z_{c(t-1)\,t-1}=0\}$ before time interval *t*, meaning that the target has not been observed so far.

Exploiting conditional independence probability property ($p(A,B)=p(A|B)p(B)$) and, the fact that current cell visit observation outcome given target occupancy is independent of past observation outcomes, equation (1) is further developed leading to the expression:

$$CPOS = \sum_{c\in N}\sum_{t} p(z_{ct}=1 \mid X_c=1)\; p(X_c=1 \mid Z_{t-1}=0)\times p(Z_{t-1}=0) \qquad (2)$$

Using Baye's theorem, posterior probability/belief of target cell *c* occupancy given past negative observation outcomes is given by:

$$p(X_c=1 \mid Z_{t-1}=0) = \frac{p(Z_{t-1}=0 \mid X_c=1)\,p(X_c=1)}{p(Z_{t-1}=0)} \qquad (3)$$

By substituting expression (3) in (2), *CPOS* can be revisited as follows:

$$\begin{aligned} CPOS &= \sum_{c\in N}\sum_{t} p(z_{ct}=1 \mid X_c=1)\; p(Z_{t-1}=0)\times \frac{p(Z_{t-1}=0 \mid X_c=1)\,p(X_c=1)}{p(Z_{t-1}=0)} \\ &= \sum_{c\in N}\sum_{t} p(z_{ct}=1 \mid X_c=1)\times p(Z_{t-1}=0 \mid X_c=1)\,p(X_c=1) \end{aligned} \qquad (4)$$

CPOS can then be finally expressed in a more convenient form as:

$$CPOS = \sum_{c\in N}\sum_{t} p_{cc}\, p_{ct} = \sum_{c\in N}\sum_{t} pos_{ct} \qquad (5)$$

where pos_{ct} represents the probability of successfully detecting the target for the first time over the period *t* during a visit in cell *c*. p_{ct} refers to the 'non-normalized' posterior probability/belief of cell target occupancy during time interval *t* which incorporates "anticipated" information feedback that would result from past visits, as derived from (3) and (4). As for p_{cc}, it is the probability on a specific agent visit *c* to correctly detect the target in cell *c* given that the target is present in cell *c* ($p(z_{ct}=1|X_c=1)$). p_{cc} depends on cell *c*. It should be emphasized that *CPOS* definition referring to first target detection assumes no 'false positive' detection from sensors (i.e. $p(z_{ct}=1|\, X_c=0) = 0$) to make sense, otherwise one could not claim that the target has been found for sure. Accordingly, an agent sensor is assumed to be false positive free, meaning that a vacant cell visit always results in a negative observation outcome by the sensing agent. Conversely, based on this assumption, a positive observation confirms that the target has been found and that the search task may be interrupted. This condition does not however preclude the occurrence of false negative outcomes ('miss') as agent sensors are not perfect. In the current setting, sensor range defining visibility or footprint (coverage of observable cells given the current sensor position) is limited to the cell being searched.

III. Mixed-Integer Linear Programming Model Formulation

A. Network Representation

A network representation is used to simplify modeling and constraint specification as well as problem-solving, as it eliminates the need to explicitly capture all constraints. These include maximum path length or deadline, admissible/legal move, and disconnected subtours elimination which may significantly impact run-time when handled explicitly.

Let $\mathcal{G}_k=(\mathbb{V}_k, \mathcal{A}_k)$ be the grid network, a directed acyclic graph associated with agent $k \in \eta=\{1,...,n\}$, where $\mathbb{V}_k = \bigcup_{t\in T} \mathbb{V}_{kt}$ is the set of vertices associated to agent states (i.e. position and orientation state variables during a given episode $t \in T=\{0,1,2,..,|T|-1\}$), and $\mathcal{A}_k$ the set of arcs (i,j) where $i,j \in$

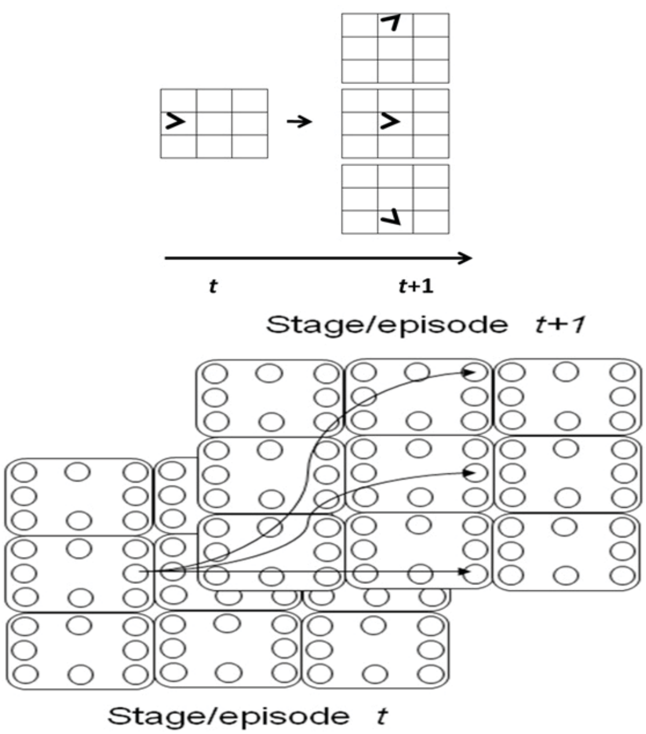

Figure 3. Agent grid network (directed acyclic graph) excerpt, over consecutive episodes t and t+1 for a 3x3 -cell grid. Nodes depict agent state (position, orientation, episode) whereas arcs capture node transition between episodes defined by possible legal moves. Squares refer to grid cells enclosing 8 possible agent orientations. A $|T|$-move path may be constructed by moving along arcs from stage 0 to stage $|T|$-1.

$\mathbb{V}_k$, reflecting possible agent state transition between consecutive episodes over the grid, corresponding to a legal move m selected from the action set A=$\{left, ahead, right\}$. $N_{kt} = N$ is the set of possible cell locations $\{1,...,|N|\}$ over the grid during episode t whereas $O_{kt} = O$ refers to the set of possible agent orientations/headings $\{E,NE,N,NW,W,SW,S,SE\}$ during episode t. As a result, $\mathbb{V}_k = \bigcup_{t\in T} \mathbb{V}_{kt} = \bigcup_{t\in T} (N_{kt} \times O_{kt})$. The nodes o and d are additional fictitious origin and destination location vertices defining legal path ends in graph. An excerpt from the abstracted representation for the agent network over two consecutive episodes is given in Fig. 3. An integer binary flow decision variable x_{ijk} is associated to each arc $(i,j) \in \mathcal{A}_k$. Agent k path solution include arcs $(i,j) \in \mathcal{A}_k$ for which $x_{ijk} = 1$. Given an initial agent state $i_0(k)$, path may be defined over the grid network traveling along arcs connecting o to d instantiating flow decision variables to build feasible paths and then, consequently, assigning visit decision variables involved in the objective function. Agent state vertex duplication over $|T|$ episodes is aimed at eliminating disjoint solution subtours otherwise difficult to handle explicitly, and provides a directed acyclic graph to represent a legal solution through binary integer flow decision variables including a multi-cycle path (possible occurrence of many visits on the same cell). Duplication implicitly satisfies path length constraint as well. The significant gain obtained through duplication clearly exceeds the cost incurred by slightly degraded model readability due to the utilization of more complex notations. The agent network includes $|O|\,|N|\,|T|$ nodes and $|O|\,|N|\,|T|\,|A|$ arcs. It is assumed that a cell c can be visited at most V_c times.

B. Mathematical Modeling

A mathematical mixed-integer linear programming (MIP) formulation is proposed for the discrete stationary target search and rescue (SAR) path planning problem. It extends the single agent model [25] to a multi-agent setting while incorporating any possible agent action moves.

The open-loop decision model captures explicitly ahead of time anticipated information feedback resulting from projected action execution to update target cell occupancy probability (belief). Accordingly, based on the completion of a projected visit in cell c during time interval t, the posterior probability of cell containment $p_{c'\,t+1}$ for any cell c' is related to its prior belief $p_{c't}$ by:

$$p_{c'\,t+1} = (1 - p_{cc}\delta_{cc'})p_{c'\,t} \tag{6}$$

where $\delta_{cc'} = 1$ if $c'=c$ and 0 otherwise. $p_{c't}$ refers to the probability/belief of cell c' target occupancy during time interval t which incorporates "anticipated" information feedback that would result from past visits. Equation (6) derives from (4) and (5) while exploiting conditional independence property in computing $p_{c't+1}$.

The variables and parameters defining the decision model are given as follows:

η : set of homogeneous agents $\{1,2,\dots,n\}$

N: set of cells defining the grid search area $\{1,2,..,|N|\}$

T: set of time intervals defining the time horizon {0,1,...,|T|-1}

V_c: maximum number of visits on cell c

p_{cc}: conditional probability of 'correct' target detection on a visit in cell c given that the target is located in c.

β_c: $1/(1-p_{cc})$

p_{ct}: 'non-normalized' belief of cell c target occupancy during time interval t. $\{p_{c0}\}$ refers to the initial belief distribution of target occupancy over the grid.

pos_{ct}: probability of success (finding the target) resulting from the observation of cell c at the end of time interval t

$CPOS$: objective function defining cumulative probability of success

v_{clt}: binary decision variable corresponding to cumulative number of visits l on cell c at the end of time interval t – v_{clt}=1 (otherwise 0)

y_{ct}: binary decision variable reflecting agent position in episode t. It indicates that cell c is visited during time interval t - y_{ct}=1 (otherwise 0)

x_{ijk}: state transition binary variable. x_{ijk} = 1 reflects agent k network state transition from state i to j between consecutive episodes. Agent k path solution includes arcs $(i,j)\in\mathcal{A}_k$ for which x_{ijk} = 1

The MIP decision model may be formulated as follows:

$$\max CPOS = \max_{\{pos_{ct}\}} \sum_{c\in N}\sum_{t\in T} pos_{ct} \qquad (7)$$

Subject to the linear convex constraint set:

Cell visits:

$$\sum_{0\le l\le V_c} v_{clt} = 1 \qquad \forall c\in N, \forall t\in T \qquad (8)$$

$$\sum_{0\le l\le V_c} l v_{clt} = \sum_{0\le t'\le t} y_{ct'} \qquad \forall c\in N, \forall t\in T \qquad (9)$$

Belief update:

$$p_{c\,t+1} = \sum_{0\le l\le V_c} \frac{p_{c0}}{\beta_c^l} v_{clt} \qquad \forall c\in N, \forall t\in T \qquad (10)$$

Probability of success:

$$pos_{ct} - p_{cc}p_{ct} \le M(1-y_{ct}) \;\; \forall c\in N, \forall t\in T, M>1 \qquad (11)$$

$$pos_{ct} \le y_{ct} \qquad \forall c\in N, \forall t\in T \qquad (12)$$

Initial probability:

$$p_{c0} = p_c(t=0) \qquad \forall c\in N \qquad (13)$$

Network coupling:

$$y_{ct} = \sum_{k\in\eta}\sum_{i_t(c)\in\mathbb{V}_k}\sum_{j_{t+1}\in\mathbb{V}_k} x_{i_t(c)\,j_{t+1}\,k} \quad c\in N, t\in T, \quad (i_t(c), j_{t+1})\in\mathcal{A}_k \qquad (14)$$

Initial agent position:

$$x_{o\,i_0(k)\,k} = 1 \qquad \forall k\in\eta, i_0(k)\in\mathbb{V}_k \qquad (15)$$

$$y_{c0} = \sum_{k\in\eta}\delta_{c\,y_0(k)} \qquad \forall c\in N, \forall k\in\eta \qquad (16)$$

Initial/final path condition:

$$\sum_{i\in\mathbb{V}_k} x_{oik} = 1 \qquad \forall k\in\eta \qquad (17)$$

$$\sum_{i\in\mathbb{V}_k} x_{idk} = 1 \qquad \forall k\in\eta \qquad (18)$$

Flow conservation:

$$\sum_{i\in\mathbb{V}_k\cup\{o\}} x_{ijk} - \sum_{i\in\mathbb{V}_k\cup\{d\}} x_{jik} = 0 \quad \forall k\in\eta, \forall j\in\mathbb{V}_k, (i,j)\in\mathcal{A}_k \qquad (19)$$

Maximum path length:

$$\sum_{i\in\mathbb{V}_k}\sum_{j\in\mathbb{V}_k/\{i\}} x_{ijk} = T \qquad \forall k\in\eta, (i,j)\in\mathcal{A}_k \qquad (20)$$

Decision variables

$$pos_{ct}, p_{ct}\in[0,1] \quad y_{ct}, v_{clt}\in\{0,1\} \quad c\in N, t\in T, \forall l\in\{0,V_c\}$$
$$x_{ijk}\in\{0,1\} \qquad \forall k\in\eta, (i,j)\in\mathcal{A}_k \qquad (21)$$

The objective function shown in equation (7) defines cumulative probability of success over the agent path solution and time horizon |T|. Constraints are governed through equations (8)-(21). For a given path solution, constraints (8) represent the cumulative number of visits paid on site c by the end of time interval t. Constraints (9) simply link that number to past visits on c so far. It should be noticed that simultaneous visits by multiple agents on a specific cell over a given time interval is implicitly prevented and reinforced by the fact that $y_{ct}\le 1$, limiting to at most one, the number of visits a cell can receive during an episode. For cell coverage purposes, we assume a maximum number of visits V_c to be performed on site c. The bound V_c can be pre-computed or selected arbitrarily large. Target occupancy probability update is governed by constraint set (10). It is the explicit form of equation (6) relating belief and number of conducted visits. Constraint sets (11) and (12) determine probability of success contributions. Both inequations mutually reflect a visit requirement to a cell to ensure a feasible observation and an admissible success contribution aligned with the objective function. M is a constant. Initial probability distribution is specified in (13). Constraint sets (14)-(20) reflect model and network coupling as well as flow constraints imposed on/by the agent network. Constraints (14) link cell visits to the agent path network, connecting outgoing arcs from network nodes (states) on stage t to the cell c being visited during episode t. Accordingly, arcs $(i_t(c),j_{t+1})$ relate to any agent state transition starting from position c at stage t. Agent k initial state $i_0(k)$ and position $y_0(k)$ as well as its related network connection are captured in constraints (15)-(16). Constraints (17)-(18)

guarantee path solution departure and final arrival points to be uniquely defined. Flow conservation governed by constraints (19) aims at balancing the number of incoming and outgoing arcs respectively for a given node. Constraints (20) guarantee a $|T|$-move path solution for an agent, but turn out to be unnecessary as solution constraints are implicitly satisfied by agent network construction. Binary and continuous domain variables are then defined in (21).

C. Single Team Network Simplification

Given agent homogeneity, a single 'team' (n agent) T-stage network $\mathcal{G}=(\mathbb{V},\mathcal{A})$ representing possible team paths may alternatively be used, requiring minor network adjustments to concurrently incorporate agent action multiplicity subject to non-simultaneous visits on a same cell. The resort to a single team network rather than multiple network-agent mapping provides additional speed-up, number of decision variable reduction and significant computer savings (by a factor n). The resulting team directed acyclic graph $\mathcal{G}=(\mathbb{V},\mathcal{A})$ captures agent multiplicity substituting x_{ijk} integer flow decision variables for x_{ij}, slightly modifying some key flow constraints:

$$\sum_{i\in\mathbb{V}} x_{ij} - \sum_{l=0}^{V_c} l v_{cl} = 0 \quad \forall c\in N, (i,j(c))\in\mathcal{A}$$

$$x_{oi} = \sum\nolimits_{k\in\eta} \delta_{i\,i_0(k)} \quad \forall i\in\mathbb{V}$$

$$\sum\nolimits_{i\in\mathbb{V}} x_{oi} = n\,, \sum\nolimits_{i\in\mathbb{V}} x_{id} = n$$

$$\sum_{i\in\mathbb{V}\cup\{o\}} x_{ij} - \sum_{i\in\mathbb{V}\cup\{d\}} x_{ji} = 0 \quad \forall j\in\mathbb{V}, (i,j)\in\mathcal{A}$$

$$x_{ij}\in\{0,1\} \quad \forall (i,j)\in\mathcal{A}$$

$$\delta_{ij} = \begin{cases} 1 & if\ i=j \\ 0 & otherwise \end{cases}$$

The expected computational gain comes at the low cost expense of reconstructing individual agent paths from the computed agent-free decision variables of the team network solution. The agent path reconstruction procedure is described next.

1) Agent Path Reconstruction

A particular agent path is reconstructed using the team network and its instantiated integer flow decision variables x_{uv}. A legal T-move agent k path is simply generated by moving along the computed team solution arcs from its departure state node $i_0(k)$ (combining initial cell and orientation) in stage 1 adding the related cell to the evolving path, up to stage T, before finally converging to the destination node d. Decision variables are progressively decremented as the path expands. The agent path reconstruction algorithm is straightforward and fast ($O(nT)$), as summarized below:

```
For  k = 1..n  do   -- cycle over agents
   u = i_0(k);  path_k = φ;  t = 1
   While (t ≤ T)
      select state transition (u,v) ∈ A such that x_uv > 0
      path_k.cell(t) = cell_u
      t = t + 1
      x_uv = x_uv − 1,  u = v
   end While   -- T − move path constructi on
end For        -- agent_k path solution
```

The path solution $path_k$ in the above procedure is composed of a sequence of T cell visits. The path element $path_k.cell(t)$ refers to the specific cell ($cell_u$) visited by agent k in period t.

D. Dynamic Planning and Time Horizon

Dynamic problem solution can be computed constructively over receding horizons by repeatedly exploiting real information feedback as it becomes available and a new optimization to progressively improve solution quality. Aside the explicit inclusion of real information feedback, large time horizon problems are similarly solved through repeated fast subproblem optimizations over receding horizons as pictured in Fig. 4. Time horizon is divided in time intervals and corresponding subproblems sequentially solved over respective episodes of period ΔT. Accordingly, a subproblem solution periodically expands the overall current partial path solution progressively incorporating a small fraction of its solution moves (subperiod δT), while updating the objective function with new path contributions. Limited move insertions define overlapping episodes, mitigating the effects of myopic path planning. A new subproblem is then periodically solved subject to the revisited objective function updated from the previous episode accounting for the partial solution being progressively built. The process is then reiterated until the time horizon has been covered. The strategy consists in taking advantage of the fast computation of reasonable time horizon subproblems over a limited number of episodes to quickly compute a near optimal solution to the original problem.

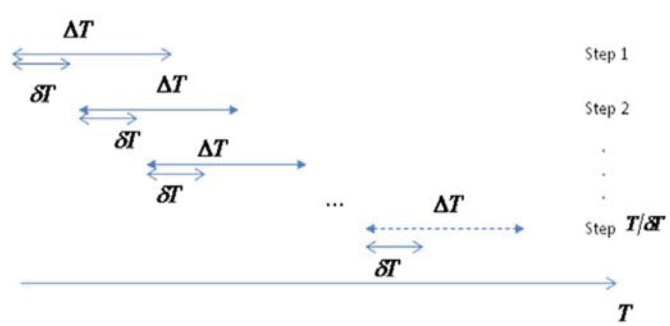

Figure 4. A large time horizon T is defined over $T/\delta T$ receding horizons of period ΔT. Moves computed in subperiods δT form the final path solution to the original problem.

It should be mentioned that the approach would be suitable if and only if the planning time horizon (in general) or period ΔT (receding horizon) is larger or equal to $\sqrt{N}$, the dimension of the grid. This condition allows total cell belief visibility over the entire grid to permit optimal planning over a given planning time horizon (the agent always perceives the whole grid). However, despite this condition, when the problem time horizon exceeds the planning time horizon, an optimal solution is not guaranteed as local optimizations myopically carried out over limited periods ΔT may still slightly degrade real optimal path solution. However, the execution of that path solution would anyway be very limited in practice, since intermediate observed outcomes would invalidate that solution and likely call for path re-planning well before the time horizon deadline. The proposed near optimal approach over receding horizons nonetheless remains simple and easy to operationalize in practice if large problem time horizons must be considered.

E. Discussion

The proposed formulation confers many advantages over alternate modeling procedures, as the linear model allows to efficiently compute a bound on the optimal solution quality through Lagrangrean programming relaxation. This provides a comparative measure to carry out performance gap analysis over alternate solutions, as well as the ability to trade-off solution quality and run-time for heuristic methods operating under tight temporal constraints. Problem-solving may be naturally achieved using well-known efficient techniques from the IBM CPLEX software [26] package.

In other respect, objective function (7) as advocated in [27],[28],[5] is quite legitimate in principle to reflect an acceptable measure of performance for target detection. However, a naive utilization may nonetheless lead to undesirable situations, raising some questionable legitimacy concerns in practice for search-and-rescue domains. In effect, equation (7) fails to discriminate different solutions demonstrating either a similar sum of success contributions, or an objective-invariant property over some feasible move permutations. This is partly due to the invariance property of the objective function against cell visits ordering. Solution symmetry may naturally occur for paths presenting multiple cell visits (cycles) or subpaths proximity in which directions are reversible or specific cell visits are interchangeable. Assuming a constant for p_{cc}, a trivial example is the circular path solution where an agent achieves a round-way trip, performing single visits (e.g. c_1 p_1=0.2, c_2 p_2=0.8). In that case, the agent trajectory may be clockwise (e.g. 0.8, 0.2) or counter-clockwise (e.g. 0.2, 0.8). Based on (7), both solutions show the same quality (p_{cc}), when in fact one of them (e.g. 0.8, 0.2) might be clearly preferable in the context of a search-and-rescue mission. As a result, the clockwise sequence of visits with steadily increasing beliefs (0.8, 0.2) might suitably lead to earlier detection and then improve the chance of target survival over the so-called 'equivalent' counter-clockwise path plan. Therefore, a more general objective function might be more appropriate to suit particular needs in further discriminating solutions (tie-breaking), such as optimizing cumulative probability of success, time-weighted cumulative probability of success or expected target detection time. In that respect, a generalized parameter-driven objective function to suitably define a variety of objectives suited by the user is proposed ($|T|$>1):

$$\max_{\{pos_{ct}\}} \frac{1+\gamma}{|T|-1} \sum_{c\in N}\sum_{t\in T}\left(1-\gamma\frac{t}{|T|}\right)pos_{ct} \qquad (22)$$

subject to the same constraint sets (8)-(10), (12)-(21) except for inequation (11) to be revised as follows:

$$(1+\gamma)(pos_{ct}-p_{cc}p_{ct}) \le M(1-y_{ct}) \quad \forall c\in N, \forall t\in T, M>|T|$$

The latter formulation is necessary to support both minimization and maximization problems. The discount parameter $\gamma \in [0,1] \cup \{-|T|\}$ in (22) tends to reduce probability of success objective contributions over time. It biases time-weighted objective definition toward specific problem dimensions. When $\gamma = 0$, the generalized function mimics the cumulative probability of success objective introduced in (7), while $\gamma = \varepsilon$ (e.g. 0.01) proposes a slightly time-weighted probability of success contributions variant to ultimately discriminate *CPOS*-based solutions with identical visits (but different ordering), in maximizing target detection earlier. The latter form corresponds to the dominant *CPOS* objective, modulated by average *CPOS(t)* values over intermediate time periods t. It provides a tie-breaking mechanism modifying the basic objective function to reduce the impact of the original *CPOS* objective function multimodality and path solution symmetry. Alternatively, $\gamma = -|T|$ specifies an expected detection time minimization problem. When $|T|$=1, the solutions are virtually equivalent for all the aforementioned objectives.

IV. MIP ALGORITHM - CPLEX SOLVER

The IBM ILOG CPLEX parallel Optimizer version 12.2.0.0 [26] was used, essentially exploiting various optimized problem-solving techniques for large size problems. CPLEX solves the (exact) mixed integer programming (MIP) problem model implicitly computing an upper bound on solution quality through integrality constraint relaxation referred as Lagrangean programming relaxation (LP).

Additional speed-up can be contemplated for implementation efficiency purposes. Accordingly, simplifications involve further reduction of the number of decision variables and constraints. This includes the suppression of the belief update p_{it} (by virtue of its explicit form described by equation (10)) and probability of success pos_{ct} variables and their respective related constraints from the model. It consists in substituting the content of those variables directly in the revisited objective function through the introduction of a set of new binary integer variables w_{clt} (and related constraints) expressed as the product of two binary integer variables, namely, the cumulative number of visits variable v_{clt} and the agent position variable y_{ct}:

$$\max_{\{w_{clt}\}} \sum_{c\in N}\sum_{t\in T} \underbrace{\sum_{0\le l\le V_c} p_{cc}\frac{p_{c0}}{\beta_c^{l-1}} w_{clt}}_{pos_{ct}}$$

$$\left(w_{clt}=v_{clt}\,y_{ct}\right)\leftrightarrow\begin{cases} w_{clt}\le v_{clt}\\ w_{clt}\le y_{ct}\\ w_{clt}\ge v_{clt}+y_{ct}-1\end{cases}$$

But, as during problem-solving LP integrality constraint relaxation on new variables tends to violate the intended quadratic relationship and then initially deteriorate solution quality by increasing both optimality gap and run-time, constraints on new variables have been rather specified as logical constraints, a feature option offered by the CPLEX solver. As a result, the approach significantly reduced the search space during the branching process of the algorithm reporting an order of magnitude gain in run-time.

V. COMPUTATIONAL EXPERIMENT

A computational experiment has been conducted to test the approach for a variety of scenarios. The value of the proposed MIP approach is assessed in terms of optimality gap and run-time. Computed solutions are reported against the relative target probability detection optimality gap shown at the end of horizon $|T|$:

$$Opt\ gap = \frac{CPOS^{*} - CPOS_a}{CPOS^{*}} \qquad (23)$$

where $CPOS^*$ is the optimal cumulative probability of success defined in (1) or a tight upper bound (LP solution), and $CPOS_a$ the performance of our approach for a given scenario. The closer (smaller) the optimality gap the better the performance.

A. Simulations

Computer simulations were conducted under the following conditions:

- Prior cell occupancy belief distribution for grid size N: exponential, uniform, cluster; N = 10x10
- Homogeneous sensor agents:
  - Actions: 8 moves
  - V_c=5 for all cells c
  - Sensor parameters: p_c=0.8 for all cells
- Hardware Platform:
  - Intel (R) Xeon (R) CPU X5670
  - Shared-memory multi-processing: 8 processors, 2.93 GHz
  - Random Access Memory: 16 Go, 64 bits binary representation (double precision)

It should be noted that as target cell occupancy probability sum up to one, performance analysis for large grid turns out to be less attractive. Accordingly, the larger the grid in general, the smaller (arbitrarily negligible) the related target cell occupancy belief, inevitably conducting either to significant visit payoffs for a limited number of prominently noticeable cells sparsely distributed over a large area, or alternatively in near similar cell visit rewards, for which any sub-optimal algorithms would likely demonstrate highly competitive (near similar) performance behavior. In both cases, this would result in a large and costly fraction of the total effort and time dedicated to the planning and construction of long and unimportant subpath segments, leading ultimately to marginal or insignificant gains. Consequently, grid instances larger than 10x10 should be further downsized and aggregated to embrace minimal belief coverage, to ensure substantial analysis and solution performance evaluation. This is why this study limited its investigation to the exploration of 10x10 grid instances.

B. Results

A sample of random simulation results is reported in Table I for a few 10x10 grid 8-move multi-agent scenarios over horizon *T*. Each entry corresponds to a separate problem instance. The subscript 'CL' to an instance identifier refers to a clustered belief distribution. Team size (number of agents) and time horizon are specified in second and third column respectively. Performances in terms of cumulative probability of success (*CPOS*) and optimality gap for the optimal CPLEX solver – MIP, are reported in the fourth column. Run-time expressed in seconds is shown in the last column.

TABLE I. PERFORMANCE OF CPLEX SOLVER (MIP) FOR A SAMPLE OF 8-MOVE 2,5 –AGENT DATA SET (10X10 GRID)

Instance	**# Agents**	**Time Horizon |*T*|**	**CPLEX Solver - MIP** *CPOS*	*Opt gap%*	**CPLEX Solver Run-time (*s*)**
A_{CL}	2	20	0.3623	0	3.24
	5	10	0.7582	0	35.9
A	2	20	0.3630	0	4.9
	5	10	0.7523	0	42.9
B	2	20	0.4021	0	6.1
	5	10	0.7650	0	58.1
C	2	20	0.3692	0	2.4
	5	10	0.7473	0 (1)	142.7(28.8)
D	2	20	0.4021	0	6.3
	5	10	0.7651	0	57.8
E_{CL}	2	20	0.5545	0	4.0
	5	10	0.8905	0	22.0
F_{CL}	2	20	0.7560	0	29.8
	5	10	0.9780	0	10.4*
G	2	12	0.6595	0	1.4
	5	10	0.9468	0	2.8*
H_{CL}	2	12	0.7087	0	2.4
	5	10	0.9547	0	2.4*
I_{CL}	2	12	0.8208	0	8.9
	5	10	0.9872	0	1.6*
J	2	20	0.3165	0	5.4
	5	10	0.6138	0	63.3
K	2	20	0.2521	0	4.4
	5	10	0.5736	0	38.13

Computational results show that an optimal solution is computable approximately in a minute run-time, except for instance C (5 agents) where 142.7 seconds were necessary against 28 seconds to get less than a 1% gap. Solutions reported for 5-agent starred instances F-I are computed much faster using the static model [29], in which maximum belief coverage ($\sum_{c\in N} p_{c0}\left(1-\sum_{l=0}^{V_c}(1-p_{cc})^l v_{cl|T|}\right)$) obtained in searching over the grid reaches more than 95%, meaning that most promising cells have already been covered and that best solutions from both decision models are nearly similar to one another, making unnecessary extensive path solution computation for the proposed decision model (no expected gain). Complete computation for the decision model nonetheless shows a 0% gap for those instances.

Computational results surprisingly indicate that 8-move near optimal multi-agent solution may generally be computed on a second timescale. It is interesting to generally note an order of magnitude (approximately 10) run-time ratio for 5 and 2 -agent problem instances respectively, despite their relative solution space size which is exponential ($8^{nT}/8^{n'T'}\sim 10^9$). Providing best or near optimal solution and measurable gain (upper bound through Lagrangean integrality constraint relaxation) for practical size problems, the approach may be repeatedly reused in dynamic settings exploiting intermediate sensor readings, given its small run-time. However, even though 5-agent scenarios involving a time horizon T larger than 12 are generally computationally prohibitive and might require several minutes to ensure convergence to solution optimality, T=10 -move planning scenarios are sufficient to dynamically build a path plan one step at a time, as the grid remains always entirely visible to the planner during planning. It should also be noticed that reported path solutions for the 5-agent T=10 scenarios mostly cover a significant portion of interesting cells as illustrated by *CPOS* performance results. Upgrading computational power technology through faster hardware and augmented parallel processing might further extend computable T.

VI. CONCLUSION

An innovative mixed-integer linear programming (MIP) approach has been proposed to solve a probabilistic open-loop multi-agent search and rescue path planning problem with anticipated feedback, in which agent actions are subject to any neighbouring move directions. Small computational cost naturally allows dynamic planning through a closed-loop environment settings where real information feedback resulting from past sensor agent observations is exploited to compute a revisited solution over a rolling horizon during the next cycle (episode). The novelty of the approach lies in a revisited combination of an extended problem formulation, an original network representation, and a refined problem-solving procedure based on linear programming CPLEX technology to efficiently compute near-optimal solution for practical size problems, usually handled through heuristic methods. For the first time, an upper bound estimate on the optimal solution naturally derived from the approach may be used for convergence or performance comparison analysis purposes, and/or trading-off solution quality and execution time. Experimental results demonstrate the value of the proposed approach for practical size problems, proving problem-solving to be feasible in reasonable time.

Future research directions will consist in considering generalized sensor footprint, and increasingly complex observation models (e.g. false-positive) while extending search to moving targets. Alternate research work will explore search problem modeling variants involving heterogeneous sensing agents.

REFERENCES

[1] K.E. Trummel and J.R Weisinger, “The complexity of the optimal searcher path problem”, Operations Research, 34 (2), 1986, pp.324-327.

[2] S. Benkoski, M. Monticino, and J. Weisinger, “A survey of the search theory literature”, Naval Research Logistics 38, 1991, 469–494.

[3] L. Stone, “What’s happened in search theory since the 1975 Lanchester Prize?”, Operations Research, 37 (3), 1989.

[4] R. Hohzaki, and K. Iida, “Optimal search plan for a moving target when a search path is given”, Math. Japonica, 41(1), 1995, pp. 175-184.

[5] H. Lau, Optimal Search in Structured Environments, PhD Thesis, University of Technology, Sydney 2007.

[6] H. Lau, and G. Dissanayake, “Probabilistic search for a moving target in an indoor environment”, In Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, 2006, pp. 3393-3398.

[7] H. Lau, and G. Dissanayake, “Optimal search for multiple targets in a built environment”, In Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, Edmonton, Alberta, Canada, 2005, pp. 228-233.

[8] J.-C. Latombe, Robot Motion Planning, ser. International Series in Engineering and Computer Science; Robotics: Vision, Manipulation and Sensors, vol. 124, Boston, Kluwer Academic Publishers, 1991.

[9] C. Choo, J. Smith, and N. Nasrabadi, “An efficient terrain acquisition algorithm for a mobile robot”, in Proc. IEEE Int. Conf. Robotics and Automation, Sacramento, CA, 1991, pp. 306–311.

[10] A. Sankaranarayanan, and I. Masuda, “Sensor based terrain acquisition: A new, hierarchical algorithm and a basic theory”, in Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, Raleigh, 1992, pp. 1515–1523.

[11] J. Svennebring, and S. Koenig, “Building terrain-covering ant robots: A feasibility study”, Auton. Robots, 16 (3), 2004, pp. 313–332.

[12] S. Wong, and B. MacDonald, “A topological coverage algorithm for mobile robots”, in Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, Las Vegas, 2003, pp. 1685–1690.

[13] S. Yang, and C. Luo, “A neural network approach to complete coverage path planning”, IEEE Trans. Syst.,Man, Cybern. B, Cybern., 34(1), 2004, pp. 718–724.

[14] I. Rekleitis, et al., “Efficient boustrophedon multi-robot coverage: an algorithmic approach”, Ann Math Artif. Intell. 52, 2008, pp. 109–142.

[15] W. Wagner, et al., “Distributed covering by ant-robots using evaporating traces”, IEEE Trans. Robot. Autom., 15(5), 1999, pp. 918-933.

[16] T. Chung, G. Hollinger, and V. Isler, “Search and pursuit-evasion in mobile robotics: A survey”, Autonomous Robots (AURO), 31(4), 2011, pp. 299-316.

[17] J.O. Royset and H. Sato, “Route Optimization for Multiple Searchers”, Naval Research Logistics, 57 (8), 2010, pp. 701-717.

[18] T. H. Chung, J. Burdick, “Analysis of search decision making using probabilistic search strategies”, IEEE Transactions on Robotics, 28 (10), 2012, pp. 132-144.

[19] Y. Jin, Y. Liao, A. Minai, and M. Polycarpou, “Balancing search and target response in cooperative unmanned aerial vehicle (UAV) Teams”, IEEE Trans on Sys Man and Cybern. Part B, 36(3), 2006, pp. 571-587.

[20] G.A. Hollinger, Search in the Physical World. CMU-RI-TR-10-20, Robotics Institute, PhD thesis, Carnegie Mellon University, 2010.

[21] T. Chung, “On Probabilistic Search Decisions under Searcher Motion Constraints”, Workshop on Algorithmic Foundation of Robotics VIII, Guanajuato, Mexico. 2009, 501-516.

[22] G.A. Hollinger, and S. Singh, "GSST: Anytime Guaranteed Search with Spanning Trees", Autonomous Robots, 29(1), 2010, pp. 99-118.

[23] A. R.Washburn, "Branch and Bound Methods for a Search Problem", Naval Research Logistics, 45, 1998, 243-257.

[24] J. Thunberg and P. Ögren, "A Mixed Integer Linear Programming approach to pursuit evasion problems with optional connectivity constraints", Autonomous Robots, 31(4), 2011, pp. 333-343.

[25] J. Berger, N. Lo, and M. Noel, "Exact Solution for Search-and-Rescue Path Planning", Proceedings of the 2nd International Conference on Information Computer Applications, Rome, Italy, February 2013.

[26] IBM ILOG CPLEX Optimizer, Available: http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/cplex-optimizer/cplex-performance/ and IBM ILOG CPLEX V12.1, 2009. Available: ftp://public.dhe.ibm.com/software/websphere/ilog/docs/optimization/cplex/ps_usrmancplex.pdf

[27] M. Morin et al. "The Optimal Searcher Path problem with a Visibility Criterion in Discrete Time and Space", Int. conference on Information Fusion, Seatle, USA, 2009.

[28] M. Morin, A.P. Papillon, F. Laviolette, I. Abi-Zeid, and C.G. Quimper, "Constraint Programming for Path Planning with Uncertainty: Solving the Optimal Search Path problem," in Proceedings of the 18th Conference on Principles and Practice of Constraint Programming, Québec, Qc, Canada, 2012, pp. 988-1003.

[29] N. Lo, J. Berger and M. Noel, "Toward Optimizing Static Target Search Path Planning", Proceedings of the 5th IEEE Symposium on Computational Intelligence for Security and Defence Applications, Ottawa, Canada, July 2012.