The Knowledge Engineering Review, Vol. 31:5, 486–497.
doi:10.1017/S0269888916000291

A constraint-based approach for planning unmanned aerial vehicle activities

CHRISTOPHE GUETTIER[1] and FRANÇOIS LUCAS[2]

[1]*SAFRAN Electronics and Defense, 100 Avenue de Paris, 91344 Massy, France;*
e-mail: christophe.guettier@safrangroup.com;
[2]*EPEX SPOT, 5 Boulevard Montmartre, 75002 Paris, France;*
e-mail: f.lucas@epexspot.com

Abstract

Unmanned Aerial Vehicles (UAV) represent a major advantage in defense, disaster relief and first responder applications. UAV may provide valuable information on the environment if their Command and Control (C2) is shared by different operators. In a C2 networking system, any operator may request and use the UAV to perform a remote sensing operation. These requests have to be scheduled in time and a consistent navigation plan must be defined for the UAV. Moreover, maximizing UAV utilization is a key challenge for user acceptance and operational efficiency. The global planning problem is constrained by the environment, targets to observe, user availability, mission duration and on-board resources. This problem follows previous research works on automatic mission Planning & Scheduling for defense applications. The paper presents a full constraint-based approach to simultaneously satisfy observation requests, and resolve navigation plans.

1 Introduction

Using Unmanned Aerial Vehicles (UAV) has become a major trend in first responder, security and defense areas. UAV navigation plans are generally defined during mission preparation. However, during mission preparation or execution, different users can request for additional observations to be performed by the UAV. It is then necessary to insert these actions in UAV navigation plans. The user must deal with constraints that will impact the overall plan feasibility, such as observation preconditions, duration of the UAV mission or resource consumption. For example, a rotorcraft can easily perform an observation using stationary flight, but has poor endurance. In turn, a fixed wing can perform longer missions but needs to orbit around a waypoint to acquire and observe a target. This paper addresses vehicle planning issues, managing constraints composed of mission objectives, execution time and resource requirements. In this problem, UAVs can communicate with the network to transmit remote videos to ground manned vehicles on ground.

The optimization problem consists in finding the path that maximizes the overall mission efficiency while ensuring mission duration and resource consumption. The structure of consumption and observation constraints make the problem difficult to model and hard to solve. Determining the shortest path may not lead to the most efficient one, since observation requests may occur for various different places. The paper proposes a constraint model for UAV activity optimization, before and during mission execution. It is formulated as a Constraint Satisfaction Problem (CSP), and implemented using the Constraint Logic Programming (CLP) framework over Finite Domains (FD). The constraint-based model combines flow constraints over {0,1} variables, with resource constraints and conditional task activation models. A solving method is also proposed, which tends to be a very generic approach for solving these complex

problems. It is based on Branch and Bound (B&B), constraint propagation and a *probing* technique. Probing is a search strategy that manages the state space solver exploration using the solution of a low-computational relaxed problem evaluation. Results are reported using a SICStus Prolog CLP(FD) implementation, with performances that suit operational needs.

The first section introduces the problem and the second one describes our constraint-based approach, compared with the state of the art. Next section presents problem formulation as a CSP. Search algorithms are then described. We give a few results on realistic benchmarks and a general conclusion.

2 Unmanned Aerial Vehicles mission planning problem

Intrinsic UAV characteristics (i.e. maximal speed, manoeuvrability, practical altitudes) have a direct impact on operation efficiency. Figure 1 presents the *Patroller*, a UAV that has large wingspan to allow medium altitude flight, which enables performing long-range missions by minimizing energy consumption. UAV operations are not only constrained by energetic resources, but also mission time and terrain structure. Figure 2 shows a set of potential waypoints to flyby. They are defined during mission preparation, by terrain analysis, mission objectives and situation assessment. Navigation constraints are also defined by available corridors, that are provided either by navigation authorities, in civilian space, or by the Air Command Order, in military context.

2.1 Informal description

A navigation plan consists in a subset of waypoints, totally ordered, estimated flyby dates and some observations to perform. Choosing the final mission plan depends on multiple criteria (duration, available energy, exposure, objectives, initial and recovery points). Maximizing mission objectives, for instance the number of observations performed during the mission, is the primary cost objective of planning automation. The overall mission duration, exposure and on-board energy may also be maintained as low as possible. To decide a mission plan, the user must deal with the following elements:

- Initial UAV conditions: initial positions and remaining energy.
- Terrain structure: defined as a set of navigation waypoints, connected by available paths. Each waypoint has a geographical reference, and a distance metric is defined to compute the value between any couple of waypoints.
- Mission objectives: the final recovery point, and any waypoint to which a sensing observation has been associated (requested by some user).
- On-board resource consumption: resources can be consumed due to UAV mobility (from a waypoint to another one) and/or observation action.
- Exposure: in some defense missions, the UAV exposure to threats shall be mastered.

Figure 1 The patroller unmanned aerial vehicles (UAV) can detect targets at long range. With such UAV, the operator requests observation at preparation time or during mission execution

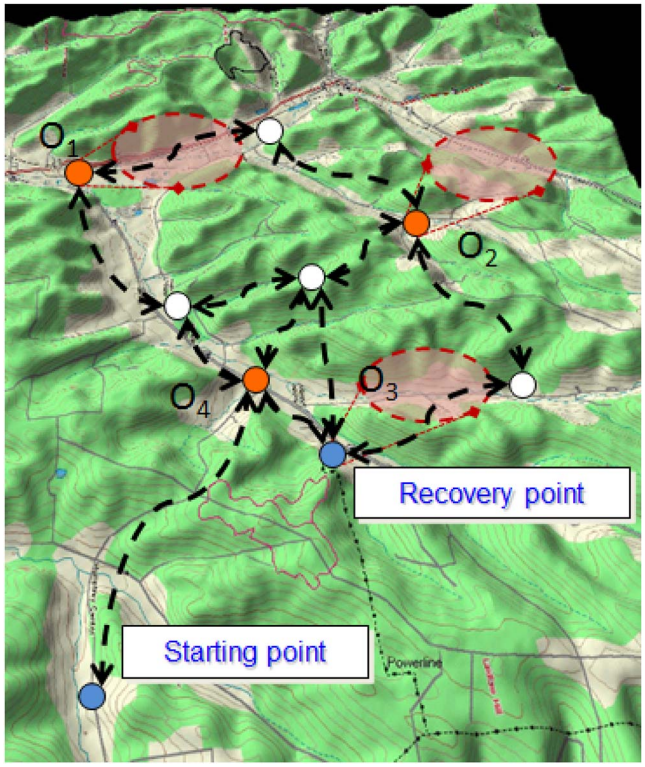

Figure 2 Navigation plan and observation requests from users. The unmanned aerial vehicles must maximize the number of requested observations (specified by field of views, represented in red), under time and energetic constraints

In general, the plan is defined at mission preparation time, but it can be redefined online due to the situation evolution:

- Situation changes: new threats might appear.
- Mission objective: sensing and observations actions can be updated. The recovery points can be updated during mission execution.
- UAV state: energy consumption is not what was expected (for instance due to wind conditions).

The remote operator receives in real time all the critical data that may require a replanning event. To be able to keep operational efficiency, it is fundamental to have fast solving algorithms that can address realistic missions plans and be able to deal with all the mission constraints.

2.2 *Example*

In Figure 2, the UAV takes off from the initial position (blue circle) and must perform a maximal set of observations among $\{O_1, O_2, O_3, O_4\}$. Each observation consumes energy and time, as for navigation between two points. In case of a defense mission, it also exposes the UAV to opponent visibility. The UAV is recovered after a last potential observation in O_3 (blue circle). To satisfy energy and UAV exposure, the user decides to only plan for observation actions $\{O_1, O_2, O_3\}$ and discards observation O_4. White circles are potential flyby navigation points.

2.3 *Complexity*

Some simplified versions of the problem are equivalent to known hard problems. If the set of observations is fixed, then the problem can be specialized as a *Travelling Salesman Problem* with multiple distance constraints. Maximizing the set of observation actions can also be relaxed as a *knapsack* problem by formulating

a path weight for each action. In both cases, these problems are known to be non polynomial (NP)-hard, and solution verification can be performed in polynomial time. Solution can be evaluated by a simple check of the navigation plan, verifying that each action is correctly scheduled and metrics are correctly instantiated. Therefore some problem instances are NP-hard on worst cases, although polynomial families may certainly be exhibited.

3 A constraint programming approach

3.1 State of the art

Several approaches can deal with such problems, ranging from classical planning to very specific algorithms:

- Domain-independent planning (Fox & Long, 2000), using Planning Domain Description Language formalisms (Fox & Long, 2003). This language can model several complex actions.
- Dedicated planners have been developed for UAV, unmanned ground vehicle and vehicle planning: IxTeT (Laborie & Ghallab, 1995), Heuristic Scheduling Testbed System (Muscettola, 1993), Reactive Model-based Programming Language (Abramson *et al.*, 2001).
- Planning frameworks like Hierarchical Task Network (Goldman *et al.*, 2002; Meuleau *et al.*, 2009) have been developed to tackle specific operational domains.

All these framework need to be complemented with CSP formulation in order to tackle resource and temporal constraints. Linear Programming (LP) techniques can also be envisaged. However, if dealing with nonlinearity or discrete variables, constraints cannot be easily reformulated into linear ones without a massive increase of the variable set.

Many heuristic search methods are based on the well-known A* (Hart *et al.*, 1968) and also commonly used in vehicle planning. Several families have been derived, such as Anytime A* (Hansen & Zhou, 2007), or other variants, adapted to dynamic environments. They can be divided into two categories: incremental heuristic searches(Koenig *et al.*, 2009) and real-time heuristic searches (Botea *et al.*, 2004). For example, an experiment has been performed for emergency landing (Meuleau *et al.*, 2011), that uses A* algorithm, integrated into aircraft avionics. These algorithms can be efficient, but are limited to simple cost objectives or basic constraint formulations.

Advanced search techniques can also solve vehicle routing problems, using Operation Research (Gondran & Minoux, 1995) (OR) or local search (Aarts & Lenstra, 1997) techniques. Simulated Annealing (Cerny, 1985), Genetic Algorithms (Goldberg, 1989), Ant Colony Optimization (Dorigo & Gambardella, 1997), and more generally metaheuristics are also good candidates. These techniques do not necessarily provide optimality nor completeness, but scale very well to large problems. However, it may require strong effort to implement complex mission constraints.

This work follows previous research in vehicle routing using CLP in Prolog and hybrid techniques (Lucas *et al.*, 2010; Lucas & Guettier, 2010). In the field of logic programming, new paradigms have emerged such as *Answer Set Programming* leading to *A-Prolog* or, more recently, *CR-Prolog* languages (with their dedicated solvers). However, their declarative extensions are not significant in the context of this work.

3.2 Using constraint logic programming

Operational users are not only interested in performance, feasibility or scalability, but at first in mission efficiency. In this paper, we consider maximizing mission observations while taking into account time, energetic or exposure constraints. To satisfy user needs, the problem must be addressed globally, which requires composition of different mathematical constraints. This can be done using a declarative logical approach, constraint predicates and classical operators (Hentenryck *et al.*, 1998). Due to the introduction of complex navigation constraints related to actions description, other approaches cannot be used in a straightforward way. The Figure 3 describes the different constraint-based models that express the complete problem. Each model describes a piece of the solution space and comprises a set of variables with

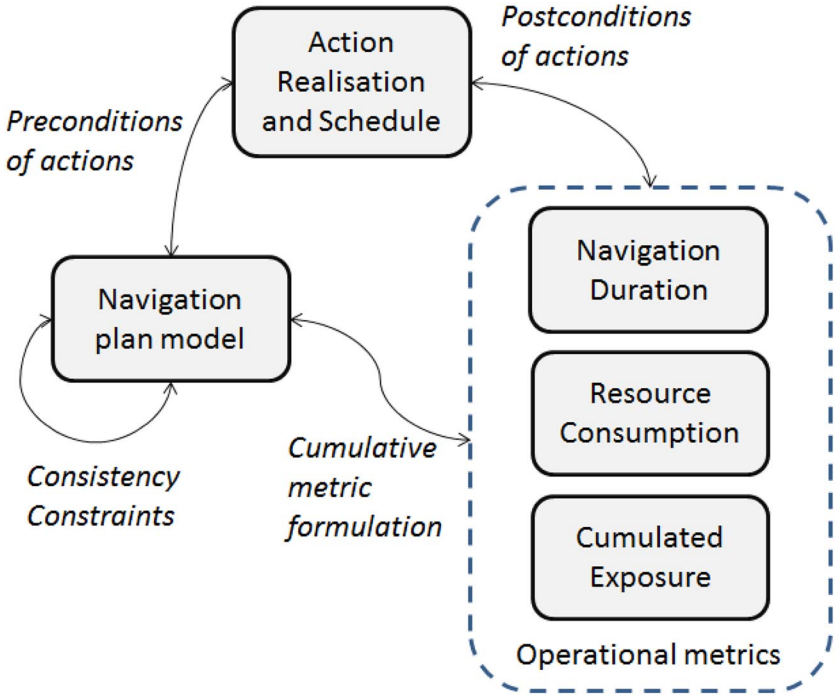

Figure 3 Problem formulation using multiple constraint models

basic applicative constraints. Models are related with constraints in order to provide consistent solutions. Five models represent the problem formulation:

- Navigation plan, as a sequence of navigation way points that compose a consistent flight plan.
- Actions realization and scheduling, that are defined thanks to post and preconditions, as well as execution dates.
- Operational metric models, which are elaborated for resource consumption, UAV exposure and flight duration. These constraints result from cumulated variables over the flight path.

The search algorithm described in the sequel, strongly rely on this model-based representation. Search techniques can be complex to design (in the case of A*) or models difficult to express (in the case of LP). Furthermore, as shown in previous works, the problem can be extended in several ways by combining different formulations. Search algorithms and heuristics must be developed or adapted without reconsidering the whole model. This can be achieved using CLP expressiveness, under a model-based development approach. CLP is a competitive approach to solve either constrained path or scheduling problems. In CLP, CSP follows a declarative formulation and is decoupled from search algorithms, so that both of them can be worked out independently. Designers can perform a late binding between CSP formulation and search algorithm. This way, different search techniques can be evaluated over multiple problem formulations. The development method also enables an easier management of tool evolutions by the designers. CSP formulation and search algorithms are implemented with the CLP(FD) domain of SICStus Prolog library. It uses the state-of-the-art in discrete constrained optimization techniques: Arc Consistency-5 (AC-5) for constraint propagation, using CLP(FD) predicates. With AC-5, variable domains get reduced until a fixed point is reached by constraint propagation.

Most of constraint programming frameworks have different tools to design hybrid search techniques, by integrating Metaheuristics, OR and LP algorithms (Ajili & Wallace, 2004). An hybrid approach is proposed to solve the mission planning problem by exploiting Dijkstra algorithm and to elaborate a meta-metric over search exploration structure. This approach, known as probing, relies on problem relaxation to deduce the search tree structure. This can be done either statically or dynamically. The CLP framework also enables concurrent solving over problem variables.

The global search technique under consideration guarantees completeness, solution optimality and proof of optimality. It relies on three main algorithmic components:

- Variable filtering with correct values, using specific labelling predicates to instantiate problem domain variables. AC being incomplete, value filtering guarantees the search completeness.
- Tree search with standard backtracking when variable instantiation fails.
- B&B for cost optimization, using minimize predicate.

Designing a good search technique consists in finding the right variables ordering and value filtering, accelerated by domain or generic heuristics. In general, these search techniques are implemented with a conjunction of multiple specific labelling predicates.

4 Problem formalization

A navigation plan is represented using a directed graph $G(X, U)$ where

- the set U of edges represents possible paths;
- the set V of vertices are navigation points. In the remaining of the paper, a vertex is denoted x, while an edge can be denoted either u or (x, x').

4.1 Navigation plan

A navigation plan is defined by the set of positive flows over edges. The set of variables $\varphi_u \in \{0, 1\}$ models a possible path from $start \in X$ to $end \in X$, where an edge u belongs to the navigation plan if and only if a decision variable $\varphi_u = 1$. The resulting navigation plan, can be represented as $\Phi = \{u | u \in U, \varphi_u = 1\}$.

4.2 Consistency constraints

From an initial position to a final one, path consistency is enforced by flow conservation equations, where $\omega^+(x) \subset U$ and $\omega^-(x) \subset U$ are outgoing and incoming edges from vertex x, respectively:

$$\sum_{u \in \omega^+(\text{start})} \varphi_u = 1, \quad \sum_{u \in \omega^-(\text{end})} \varphi_u = 1 \tag{1}$$

$$\sum_{u \in \omega^+(x)} \varphi_u = \sum_{u \in \omega^-(x)} \varphi_u \leqslant 1 \tag{2}$$

Since flow variables are $\{0, 1\}$, equation (2) ensures path connectivity and uniqueness while equation (1) imposes limit conditions for starting and ending the path. This constraint provides a linear chain alternating flyby waypoint and navigation along the graph edges.

4.3 Operational metric formulations with cumulative constraints

Assuming a given date D_x associated with a position (e.g. vertex) x we use a path length formulation (3). Variable D_x is expressing the time at which the UAV reaches a position x (see example in Figure 4). Assuming that variable $d_{(x',\,x)}$ represents the time taken to perform the manoeuvre from position x' to x (at an average edge speed) and perform potential observations on x'. This time cumulates action duration and navigation between waypoints.

We have

$$D_x = \sum_{(x', x) \in \omega^-(x)} \varphi_{(x', x)} \left(d_{(x', x)} + D_{x'} \right) \tag{3}$$

$$\forall (x, x') \in U,\ d_{(x, x')} \in \mathbb{N},\ l_{(x, x')} \leqslant d_{(x, x')} \leqslant u_{(x, x')} \tag{4}$$

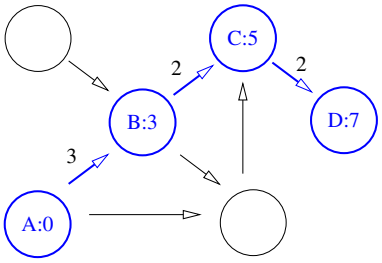

Figure 4 Illustrating manoeuvres over a graph of navigation waypoints. This graph is a spatial representation of navigation plan. A solution, representing the unmanned aerial vehicle manoeuvres, corresponds to the set of positive values (here $\Phi = \{(A, B), (B, C), (C, D)\}$). Assuming a cumulative time metric (edge values are transit times), flyby instant is $\Delta = \{(A, 0), (B, 3), (C, 5), (D, 7)\}$

Note that upper and lower speed limits (respectively $u_{(x,\,x')}$ and $l(x,\,x')$) in (4) are an edge. Other operational metrics are expressed with similar cumulative constraints. They are used to propagate resource consumption with variables $< R_x,\, r_{(x,\,x')} >$, and UAV cumulated exposure with variables $< E_x,\, e_{(x,\,x')} >$. These variables and constraints are also associated to vertices and edges as for constraint (4). In practice E_x and R_x are normalized as a percentage of consumption.

4.4 Action realization and schedule

The set of navigation points belonging to the plan P can also be expressed as follows:

$$\forall x,\, n_x = \min(1,\, D_x),\, P = \{x \in X,\, n_x = 1\} \tag{5}$$

where n_x states whether a position x is part of the navigation plan. If $D_x = 0$, the UAV does not flyby x. For simplicity, n_x is assimilated to a Boolean variable.

A set of potential observation actions O is represented by $\|V\|$ variables $O_x \in \{0, 1\}$ and

- an observation duration constant δ_x.
- a resource consumption constant ρ_x.
- a visibility exposure constant η_x.

If there is no action on vertex O_x to be performed, its default value is 0. Action activation model is defined using the following preconditions (6) and postconditions (7–9):

$$O_x \Rightarrow n_x \wedge \mathrm{E}_x \geqslant v_x \quad \forall x,\, \forall x' \in \omega^+(x) \tag{6}$$

$$d_{(x,x')} = \delta_{(x,x')} + O_x.\delta_x \tag{7}$$

$$r_{(x,x')} = \rho_{(x,x')} + O_x.\rho_x \tag{8}$$

$$e_{(x,x')} = \eta_{(x,x')} + O_x.\eta_x \tag{9}$$

and where constant $\delta(x, x')$ is the time to navigate from point x to x'.

In equation (6), the constant v_x is an exposure threshold that is tolerated and compared with the total exposure up to waypoint x. Indeed, to satisfy the action, the UAV must be incoming to the observation location, which is the role of the term n_x. This way, each observation precondition is constrained by the level of exposure. The observation resulting schedule can be defined by the waypoint flyby dates D_x.

In this paper, arrival date at the recovery point is enough to constraint the whole CSP. However, the model can be extended to express preconditions for energy and time on observation activations.

$D_{\text{end}} \leqslant D_{\max}$, where $D_{\max}$ is the maximal mission duration.

Similarly, there must be remaining energy when arriving at the recovery point.

$E_{\text{end}} \geqslant 0$.

Other preconditions can be defined, depending on the type of action to perform (including time windows, communication, target mobility). Using our model, it is easy to overload the conjunction. However, the problem can become very complex and there is not necessarily a need as long as we consider a unique UAV. Moreover, we notice that the set of preconditions is predominant compared with postconditions.

4.5 Optimization problem

The final cost function is the total amount of observations to perform.

$$\Omega(V) = \sum_{x \in V} O_x \tag{10}$$

The sets of decision variables are Φ and O such that the CSP can then be formulated in Prolog as follows (1):

Algorithm 1 *Optimizing observations*

Instantiate variable sets Φ, O
- **Satisfying** navigation constraints (1), (2), (5),
- **Satisfying** metric constraints (3), (4) and
- **for all** actions $\{O_1, \ldots, O_x, \ldots, O_n\}$
  - **satisfying** preconditions (6)
  - **satisfying** postconditions (7), (8) and (9)

Maximizing $\Omega(V)$

5 Search algorithms

5.1 Overview

The goal of hybridizing global solving with stochastic approaches is to save the number of backtracks by quickly focussing the search towards good solutions. It consists in designing the tree search according to the problem structure, revealed by the probe. The idea is to use the prober to order problem variables, as a pre-processing. Instead of dynamic probing with tentative values such as in (Sakkout & Wallace, 2000), this search strategy uses a static prober which orders problem variables to explore according to the relaxed solution properties. Then, the solving follows a standard CLP search strategy, combining variable filtering, AC-5 and B&B. As shown in Figure 5, the probing technique proceeds in three steps (the three blocks on the left). The first one is to establish the solution to the relaxed problem. As a reference, we can, for example, compute the shortest path between starting and ending vertices, abstracting away mandatory waypoints. The next step is to establish a minimal distance between any problem variable and the solution to the relaxed problem. This step can be formally described as follows. Let $X_s \subset X$ be the set of vertices that belong to the relaxed solution. The distance is given by the following evaluation:

$$\forall x \in X,\ \delta(x) = \min_{x' \in X_s} ||(x, x')|| \tag{11}$$

where $||.||$ is a specific distance metric (in our case, the number of vertices between x and x'). The last step uses the resulting partial order to sort problem variables in ascending order. At global solving level the relaxed solution is useless, but problem variables are explored following this order.

5.2 Properties

Two interesting probe properties can be highlighted:

- *probe complexity*: since computation of minimum distance between a vertex and any node is polynomial thanks to Dijkstra or Bellman-Ford algorithms, the resulting probe construction complexity

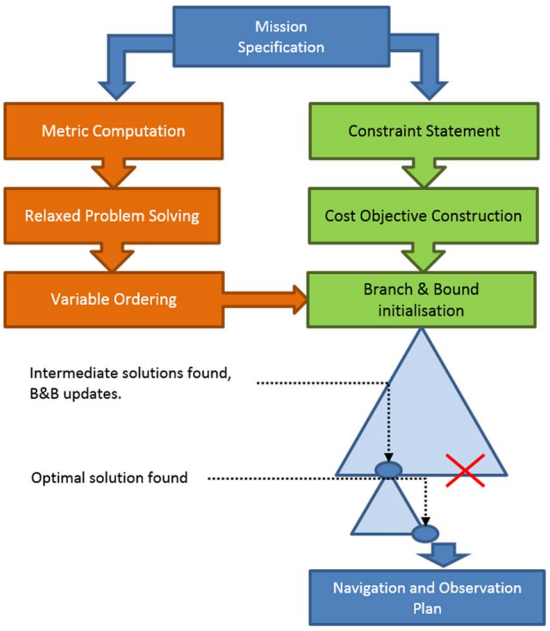

Figure 5 Diagram of the complete solver using probing techniques

is still polynomial in worst cases. The complexity of quicksort can in practice be neglected (see below for further details).

- *probe completeness*: since the probe does not remove any value from variable domains and the set of problem variables remains unchanged, the probe still guarantees global solving completeness.

Complexity analysis: let γ be the cardinality of V_s and n the one of V. The complexity of probe construction is

- in worst case performance: $O(n^2)$;
- in average case performance: $O(\gamma.n.\log(n))$.

Sketch of the proof: the probing method first determines the minimal distance between all vertices $X' \in X'$ where $X' = X \backslash X_s$ and any vertex $x_s \in X_s$. A Dijkstra algorithm runs over a vertex x_s allows to compute the distance to any point of X' with $O(n.\log(n))$ worst case complexity where n is the number of nodes in X. This has to be run over each vertex of X_s and a comparison with previous computed values must be done for every vertex x', to keep the lowest one. Thus, the resulting complexity is $O(\gamma.n.\log(n))$. Variables must finally be sorted with a quicksort-like algorithm. The worst case complexity of this sort is $O(n^2)$, but is generally computed in $O(n.\log(n))$ (average case performance). Hence, the worst case complexity of the probing method is $O(n^2)$, but in practice behaves in $\max\{O(\gamma.n.\log(n)), O(n.\log(n))\} = O(\gamma.n.\log(n))$.

5.3 Pseudocode

Algorithm 2 synthesizes probe construction mechanisms. First, a vector L_d of size n (n being the number of nodes in X) is created and initialized with infinite values. At the end of the execution, it will contain a value associated to each vertex, corresponding to the minimal distance between this vertex and the solution to the relaxed problem. To do so, a Dijkstra algorithm is run over each node of the solution. During a run, distances are evaluated and replaced in L_d if lower than the existing value (in the pseudo code, comparison are made at the end of a run for easier explanation). Once minimal distances are all computed, they are used to rank the set of vertices X in ascending order (to be used by the complete solver).

Algorithm 2 *Probe construction*

1: Initialize a vector L_d of distances (with infinite values)
2: Get P the best solution of the relaxed problem
3: **for** each node x_i of P **do**
4: $L_d^{'} \Leftarrow$ Run Dijkstra algorithm from x_i
5: $L_d = \min(L_d, L_d^{'})$ (value by value)
6: **end for**
7: Sort X using L_d order
8: **return** the newly-ordered X list

6 Preliminary results

Experiments on four benchmarks are presented. They are representative of modern peace keeping missions or disaster relief. Missions must be executed in <30 minutes. Areas range from 5×5 to 20×20 km:

1. Recon villages: observing different villages after a water flooding event.
2. Reinforce UN: bring support to a United Nations mission by observing an unsecure town.
3. Sites inspections: observing different parts of a town during inspection of suspect sites.
4. Secure humanitarian area: observing different threats before securing refugees, over a large area.

For each benchmark, four experimentations are run. Two sets of runs are performed, one with the simple branch and bound, the other one with the probing method. For each set, two different constraints are preconditions to observation actions cumulated energetic resource and cumulated UAV exposure as defined by the generic constraint (3). In practice, the exposure threshold is set between 10 and 20% for each observation action. This overconstrains the problem, allowing us to observe performance differences.

Table 1 reports the time to find the optimal solution, as well as for proving optimality. It also shows the maximal number of observations that can be executed. Simple problems can be solved fairly quickly, but

Table 1 Results overview on benchmark scenarios, maximizing the number of action to perform

Experiments			Results		
Problem	Algorithm	Actions	Time (ms) for opt.	Proof	Best Value (#actions)
1. Recon villages (22 nodes, 74 edges, 702 vars, 2251 constraints)					
Energy	Probing	3	250	560	1
Exposure	Probing	3	234	609	1
Energy	Simple	3	274	1092	1
Exposure	Simple	3	358	982	1
2. Reinforce UN (23 nodes, 76 edges, 723 vars, 2312 constraints)					
Energy	Probing	3	93	93	3
Exposure	Probing	3	296	702	2
Energy	Simple	3	1045	1061	3
Exposure	Simple	3	5460	11 139	2
3. Site inspection (22 nodes, 68 edges, 654 vars, 2081 constraints)					
Energy	Probing	4	109	249	3
Exposure	Probing	4	187	312	3
Energy	Simple	4	717	1575	3
Exposure	Simple	4	1451	2261	3
4. Secure area (33 nodes, 113 edges, 1069 vars, 3447 constraints)					
Energy	Probing	3	2371	4977	2
Exposure	Probing	3	7566	10 234	2
Energy	Simple	3	8237	15 944	2
Exposure	Simple	3	22 074	29 375	2

the last benchmark is more computation demanding, which is certainly due to a large area to cover. On the second benchmark, exposure constraints prevent from performing all observations. Again for all the problem instances, the probing method improves drastically the solver performances, which confirm former researches (Lucas *et al*., 2010; Lucas & Guettier, 2012). By comparing with energetic constraints, exposure preconditions makes the problem really harder to solve.

7 Conclusion

This paper shows the development of the mission planning framework, that can be used either for C2 systems or for unmanned systems. Introducing actions with complex preconditions and postconditions increases the practical complexity of problem instances. In particular, with the existing design, the solving approach does not scale huge numbers of observation or large graph structures. Nevertheless, as expected by previous results, the probing approach improves drastically solving performances. Using the modeling approach, the formulation of action preconditions and postconditions can be extended in several ways. Further works will focus on scalability as well as different forms of probing, relying on action definition in the relaxation process.

Acknowledgement

We acknowledge Pr. Arnaud de la Fortelle and Pr. Patrick Siarry for their constant support.

References

Aarts, E. & Lenstra, J. 1997. *Local Search in Combinatorial Optimization*. Princeton University Press.

Abramson, M., Kim, P. & Williams, B. 2001. Executing reactive, model-based programs through graph-based temporal planning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Ajili, F. & Wallace, M. 2004. Hybrid problem solving in ECLiPSe. In *Constraint and Integer Programming Toward a Unified Methodology*, volume 27 of Operations Research/Computer Science Interfaces Series, Chapter 6. Springer, 2004.

Botea, A., Mller, M. & Schaeffer, J. 2004. Near optimal hierarchical path-finding. *Journal of Game Development* **1**(1), 7–28.

Cerny, V. 1985. A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. *Journal of Optimization Theory and Applications* **45**, 41–51.

Dorigo, M. & Gambardella, L. 1997. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation* **1**(1), 53–66.

Fox, M. & Long, D. 2000. Automatic synthesis and use of generic types in planning. In *Proceedings of the Artificial Intelligence Planning System*, AAAI Press, 196–205.

Fox, M. & Long, D. 2003. PDDL 2.1: an extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research* **20**, 61–124.

Goldberg, D. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley.

Goldman, R., Haigh, K., Musliner, D. & Pelican, M. 2002. MACBeth: a multi-agent constraint-based planner. In *Proceedings of the 21st Digital Avionics Systems Conference*, 2, 7E3:1–8.

Gondran, M. & Minoux, M. 1995. *Graphes et Algorithmes*. Editions Eyrolles.

Hansen, E. & Zhou, R. 2007. Anytime heuristic search. *Journal of Artificial Intelligence Research* **28**, 267–297.

Hart, P., Nilsson, N. & Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems, Science and Cybernetics* **4**(2), 100–107.

Hentenryck, P. Van, Saraswat, V. A. & Deville, Y. 1998. Design, implementation, and evaluation of the constraint language CC(FD). *The Journal of Logic Programming* **37**(1–3), 139–164.

Koenig, S., Sun, X. & Yeoh, W. 2009. Dynamic Fringe-Saving A*. In *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2, 891–898.

Laborie, P. & Ghallab, M. 1995. Planning with Sharable Resource Constraints. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Lucas, F. & Guettier, C. 2010. Automatic vehicle navigation with bandwidth constraints. In *Proceedings of MILCOM 2010,* November.

Lucas, F. & Guettier, C. 2012. Hybrid solving technique for vehicle planning. In *Proceedings of Military Communication Conference (MILCOM)*.

Lucas, F., Guettier, C., Siarry, P., de La Fortelle, A. & Milcent, A.-M. 2010. Constrained navigation with mandatory waypoints in uncertain environment. *International Journal of Information Sciences and Computer Engineering (IJISCE)* **1**, 75–85.

Meuleau, N., Plaunt, C., Smith, D. & Smith, T. 2009. Emergency landing planning for damaged aircraft. In *Proceedings of the 21st Innovative Applications of Artificial Intelligence Conference*.

Meuleau, N., Neukom, C., Plaunt, C., Smith, D. E. & Smithy, T. 2011. The emergency landing planner experiment. In *21st International Conference on Automated Planning and Scheduling*.

Muscettola, N. 1993. HSTS: integrating planning and scheduling. In Technical Report CMU-RI-TR-93-05, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA.

Sakkout, H. E. & Wallace, M. 2000. Probe backtrack search for minimal perturbations in dynamic scheduling. *Constraints Journal* **5**(4), 359–388.