Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

Vehicle Routing Problem with Metric Temporal Logic Specifications

Sertac Karaman Emilio Frazzoli

Abstract— **This paper proposes a novel version of the Vehicle Routing Problem (VRP). Instead of servicing all the customers, feasible solutions of the VRP instance are forced to satisfy a set of complex high-level tasks given as a Metric Temporal Logic (MTL) specification, which allows complex quantitative timing constraints to be incorporated into the problem. For the resulting Vehicle Routing Problem with Metric Temporal Logic Specifications (VRPMTL), a Mixed-Integer Linear Programming (MILP) based algorithm is provided that solves the problem to optimality. Examples for optimal multi-UAV mission planning are provided where MTL is used as a high level language to specify complex mission tasks.**

I. Introduction

The Vehicle Routing Problem and its many variants have been studied in the literature for many decades. Although several different variants have been proposed and have found applications in the real world, some important issues have not been addressed yet. The VRP by definition requires all the customers to be serviced employing all the vehicles. However, in some applications, there may be problem instances in which not all customers must necessarily be serviced, and successful completion of certain tasks depends on relative timing between servicing different customers.

For instance, a two-customer task may be completed only if the two customers are visited in a given order, within a specific interval in time (for example, when the first customer wants to deliver a package to the second one within a fixed time). Another two-customer task may be accomplished by visiting only one of the two customers; this occurs, for example, when the two "customers" are in fact suppliers of a certain commodity product. More in general, one can desire to satisfy conjunctions and disjunctions of all these requirements, which would constitute a possibly very complex specification. Besides not addressing these type of issues, one of the important drawbacks of current approaches to formulating and solving vehicle routing problems is the lack of a high level language to specify complicated tasks in a natural way. In this paper we employ Metric Temporal Logics as a natural language for specification of such complex tasks and consider applications to multi-UAV mission planning.

The VRP has been extensively studied by the Operations Research community for over half a century. Several different variants have been considered and several different techniques were employed to obtain optimal or suboptimal solutions [23]. Variants of the VRP have been applied to many real-world problems in different domains [5], [16], [20]. Applications to multi-UAV routing and mission planning problems are also extensively studied in the literature [1], [6], [21], [22], [24] and experimental demonstrations were provided [19].

VRP Algorithms consist of heuristics and exact algorithms. Even though heuristics quickly provide solutions that are often close to the optimum, they have no worst case guarantees (see for example [7], [8] for a survey of some of the heuristics). On the other hand, exact algorithms provide solutions for instances of practical sizes. We should note that many of the techniques that provide an optimal solution to VRP are based on Mixed-Integer Linear Programming (MILP), which were classified in [23]. The aforementioned multi-UAV mission planning applications of the VRP all employ MILP-based formulation to construct optimal solutions.

In this paper, we consider a form of temporal logic as a language to define complex missions. Temporal logic is a branch of philosophy and was first studied by philosophers. It was taken into the Computer Science literature by the seminal work of Pnueli [18] to reason about concurrent computer programs [15]. Since then, many different temporal logics have been considered, especially for model checking purposes [9]. In fact, computer programs can easily be modeled using state transitions without knowing when these transitions actually occur. However, for several other applications the actual time that a change occurs in the system becomes quite important and one desires to be able to reason about time in a quantitative way as well. Such types of logics have been studied extensively under the name of real-time logics, for which several decidability results are available [4], [3], [11], [14], [17], as well as model checking algorithms [2], [10]. In this paper, we concentrate on Metric Temporal Logic (MTL), one of the many real-time logics.

Our previous work considered the VRP with Linear Temporal Logic Specifications, which employs a non-real-time logic LTL_{-X} to express complex tasks in a VRP instance [12], [13]. Indeed, LTL_{-X} is capable of expressing several logical and temporal constraints of the VRP, e.g., disjunctive servicing constraints, or precedence constraints on the customers. However, LTL_{-X} can not be used for reasoning about quantitative properties of time. For example, even though it is possible to constrain the order in which two customers are serviced in LTL_{-X}, it is not possible to specify the time elapsed between the two services. To remedy this problem, we consider a strictly more expressive real-time logic, namely MTL, and provide a different algorithm to obtain the optimal solution for VRPs with MTL specifications.

The contributions of this paper are as follows. First, we introduce a novel variant of VRP, the Vehicle Routing

Sertac Karaman and Emilio Frazzoli are with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA. sertac,frazzoli@mit,edu

Problem with Metric Temporal Logic Specifications. This new variant of VRP is a generalization of the well-known Vehicle Routing Problem with Time Windows. In addition, we present a novel MILP-based algorithm that solves this problem to optimality under some assumptions. Finally, we also present applications of VRPMTL as examples of multi-UAV mission planning of practical sizes.

The rest of the paper is organized as follows. In Section II a formal introduction to the Vehicle Routing Problem is presented by relating it to a MILP formulation. In Section III the Metric Temporal Logic language is introduced with its syntax and semantics after providing some preliminaries. Section IV is devoted to the introduction of Vehicle Routing Problem with Metric Temporal Logic Specifications (VRPMTL) for which a MILP-based algorithm that solves VRPMTL instances to optimality is presented in Section V. A multi-UAV mission planning example is provided in Section VI. The paper ends with conclusions in Section VII.

II. Vehicle Routing Problem

This section introduces the VRP with one of its common MILP formulations. This formulation was studied in [24] for UAV scheduling purposes, where loitering of UAVs as well multiple launching or landing sites were accounted for. The rest of this section presents the same formulation with slightly different notation.

From a practical point of view, consider a UAV Cooperative Control scenario with N targets, L launch sites, and C landing sites, and K UAVs. Let us denote the sets of targets, launch sites, landing sites and UAVs by $\mathcal{N}=\{N_1,\ldots,N_N\}$, $\mathcal{C}=\{C_1,\ldots,C_C\}$, $\mathcal{L}=\{L_1,\ldots,L_L\}$, and $\mathcal{K}=\{K_1,\ldots,K_K\}$, respectively. Let us also denote the set of departing nodes by $I=\mathcal{L}\cup\mathcal{N}$ and the set of approaching nodes by $\mathcal{J}=\mathcal{N}\cup\mathcal{C}$.

One class of the parameters of the optimization problem is the set of times t_{ijk} defined for $\forall i\in I$, $\forall j\in\mathcal{J}$, and $\forall k\in\mathcal{K}$. These indicate the time it takes for UAV k to travel from a departing node i to an approaching node j. This parameter can also include the servicing time at node j. A second class of parameters is the set of c_k , $k\in\mathcal{K}$. This parameter is set to indicate a relative risk per time for using UAV k in the mission. A final class of parameters is the set of r_k, $k\in\mathcal{K}$, and indicates the time that UAV k can be used in the mission before it is out of fuel.

The optimization problem also includes the binary decision variables x_{ijk}, defined for $i\in I$, $j\in\mathcal{J}$, and $k\in\mathcal{K}$. Such decision variables indicate whether or not a UAV travels from a departing node to an approaching node. More precisely, $x_{ijk}=1$ if UAV k travels from node i to node j, and $x_{ijk}=0$ otherwise. The formulation also has three types of continuous variables t_j defined for $j\in\mathcal{J}$, t_{jk} defined for $j\in\mathcal{C}$ and $k\in\mathcal{K}$, and s_{ik} defined for $i\in I$ and $k\in\mathcal{K}$ [*]. The variables t_j and t_{jk} indicate the time that target j is serviced and the time that UAV k has landed on landing site j, respectively. The variable t_{jk} is equal to zero if UAV k does not land on landing site j, but lands on another one, or if it was not launched at all. Finally the variables s_{ik} indicate the amount of time that UAV k loiters around target i after servicing it. This time can naturally be zero. If UAV k does not service target i at all, then this variable is meaningless.

The formulation of the problem without the temporal constraints is as follows.

$$\min \quad \sum_{j\in\mathcal{C}}\sum_{k\in\mathcal{K}} c_k t_{jk}; \tag{1}$$

$$\text{s. t.} \quad \sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{J},j\neq i} x_{ijk}\leq 1, \quad \forall i\in\mathcal{N}; \tag{2}$$

$$\sum_{i\in I,i\neq h} x_{ihk}-\sum_{j\in\mathcal{J},j\neq h} x_{hjk}=0, \quad \forall h\in\mathcal{N}, \ \forall k\in\mathcal{K}; \tag{3}$$

$$\sum_{i\in\mathcal{L}}\sum_{j\in\mathcal{N}} x_{ijk}\leq 1, \quad \forall k\in\mathcal{K}; \tag{4}$$

$$\sum_{i\in\mathcal{L}}\sum_{j\in\mathcal{N}} x_{ijk}-\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{C}} x_{ijk}=0, \quad \forall k\in\mathcal{K}; \tag{5}$$

$$t_i+t_{ijk}+s_{ik}-M(1-x_{ijk})\leq t_j, \quad \forall i\in I, \ \forall j\in\mathcal{N}, j\neq i \ \forall k\in\mathcal{K}; \tag{6}$$

$$t_i=0, \quad \forall i\in\mathcal{L}; \tag{7}$$

$$t_i+t_{ijk}+s_{ik}-M(1-x_{ijk})\leq t_{jk}, \quad \forall i\in\mathcal{N}, \ \forall j\in\mathcal{C}, \ \forall k\in\mathcal{K}; \tag{8}$$

$$t_{jk}\leq r_k, \quad \forall j\in\mathcal{C}, \ \forall k\in\mathcal{K}; \tag{9}$$

where M is a big enough constant; (1) is the objective function. Constraints (2) ensure that every target is serviced at most once. Constraints (3) are the flow constraints ensuring that every visited node must also be left. Constraints (4) ensure that if a UAV is launched then it is launched from a launch site, whereas Constraints (5) are to make sure that each UAV that is launched lands on a landing site. Constraints (6,7,8) are the timing related constraints. These constraints guarantee a feasible flow of time. They also prevent cyclic solutions and act as sub-tour elimination constraints. Finally, constraints (9) ensure that each UAV lands before it is out of fuel.

Notice that, by inequalities (2) and (4), not every target has to be serviced nor every UAV has to be employed in this formulation. Indeed, if (2) and (4) were both equalities, then the formulation (1-9) would be the standard VRP. The relaxation obtained by making them inequalities is a first step to generalizing the VRP to VRPMTL. In the rest of this paper, some other constraints are posed onto this formulation to ensure that targets are visited while satisfying temporal constraints posed by MTL. From now on, a solution will be called a feasible solution of VRP if it satisfies (2-9).

III. Metric Temporal Logic

This section introduces the Metric Temporal Logic (MTL) and it is developed independently from Section II. Before going into the syntax and semantics of MTL, transition systems and their timed executions, as timed state sequences, will be presented. The semantics of MTL will be described using timed state sequences at the end of this section.

[*]For the parameters t_{ijk} and variables t_j, t_{jk} we make an abuse of notation using the same variable name, and distinguishing them by the number of indices.

A. Preliminary Definitions

Informally speaking, atomic propositions are the indivisible components of reasoning. They are indivisible in the sense that they can not be represented by any MTL formula of other components and for this reason they are assigned a literal of their own. A formal definition is as follows.

Definition III.1 (Atomic Proposition) *An atomic proposition p is a statement over the problem variables and parameters that is either* `True` *or* `False` *at a given time instance.*

Atomic propositions are statements like "Target 1 is serviced by UAV 1" which will be either `True` or `False` at any time for any given feasible solution of VRP.

Definition III.2 (Transition System) *A Transition system is a tuple $\mathcal{TS} = (Q, Q_0, \rightsquigarrow, \Pi, \models)$ where Q is a set of states, $Q_0 \subset Q$ is a set of initial states, $\rightsquigarrow \subset Q \times Q$ is a transition relation, Π is a set of atomic propositions, and $\models: Q \to 2^{\Pi}$ is a labeling function.*

Every state $s_i \in Q$ assigns a unique value to an atomic proposition $p \in \Pi$. A standard notation is to use $s_i[p]$ to indicate the value that the atomic proposition p is assigned when a state s_i is active. For formal verification of computer programs, transition systems are used as an abstract model for the program under consideration. Similarly, in this paper, the transition system constitutes a model for a given VRP instance. The following example illustrates this idea.

Example III.3 *Consider a simple VRP instance with two targets and a single UAV. To model this VRP instance, let us define the atomic propositions "Target 1 is serviced" and "Target 2 is serviced" which we denote as p_1 and p_2, respectively. In this case the set of atomic propositions is $\Pi = \{p_1, p_2\}$. The set of states can be written as $Q = \{s_0, s_1, s_2\}$, where we have $s_0[p_1] = s_0[p_2] =$* `False`*; $s_1[p_1] =$* `True`*, $s_1[p_2] =$* `False`*; and $s_2[p_1] =$* `False`*, $s_2[p_2] =$* `True`*. The set of initial states is $Q = \{s_0\}$. Now the transition relation can be given as $\rightsquigarrow = \{(s_0,s_1),(s_0,s_2),(s_1,s_2),(s_2,s_1),(s_1,s_0),(s_2,s_0)\}$. Finally the labeling function can be defined as $\models(s_0) = \emptyset$, $\models(s_1) = \{p_1\}$, and $\models(s_2) = \{p_2\}$.*

In the rest of this section, let us note the following definitions similar to the ones in [3]. An interval is any convex subset of the real line $\mathbb{R}$. The left and the right limits of an interval I are defined as $l(I) = \inf_{x \in I} x$ and $r(I) = \sup_{x \in I} x$, respectively. An interval I is said to be left closed if $l(I) \in I$, and right closed if $r(I) \in I$. Two intervals I_1 and I_2 are said to be adjacent if the right limit of I_1 is equal to the left limit of I_2. Given $t \in \mathbb{R}_{\geq 0}$, the interval, which has its left and right limits equal to $t + l(I)$ and $r(I) + t$ and which is left closed if I is left closed and right closed if I is right closed, is denoted as $t + I$. The notation $I - t$ will be used in a similar way.

Definition III.4 (State and Interval Sequences) *A state sequence $\sigma = (s_0 s_1 s_2 \dots)$ is a possibly infinite sequence of states with $s_i \in Q$ for $\forall i \geq 0$, $s_0 \in Q_0$ and $(s_i, s_{i+1}) \in \rightsquigarrow$ for $\forall i \geq 0$. Similarly an interval sequence $\kappa = (I_0, I_1, I_2)$ is a possibly infinite sequence of intervals which satisfy the following*

- *I_0 is left closed and $l(I) = 0$;*
- *for all $i \geq 0$, the intervals I_i and I_{i+1} are adjacent;*
- *every time $t \in \mathbb{R}_{\geq 0}$ belongs to an interval in κ.*

Definition III.5 (Timed State Sequence) *A timed state sequence $\omega = (\sigma, \kappa)$ is a pair consisting of a state sequence σ and an interval sequence κ.*

Informally, a timed state sequence indicates the time interval during which a state of the transition system has been active, i.e., the state s_i has been active in the interval I_i for $\forall i$. The function $s(t) : \mathbb{R}_{\geq 0} \to Q$ will denote the state that is active at a given time t, i.e., $s(t) = s_i$ if and only if $t \in I_i$.

In the transition system model of a VRP instance, any feasible solution can be associated with a timed state sequence.

Example III.6 *Let $\mathcal{TS}$ be the transition system model of the VRP instance given in Example III.3. Consider the feasible solution in which UAV 1 launches at time $t = 0$, services targets 1 and 2 at times $t = 1.5$ and $t = 2.5$ respectively. Then the corresponding timed state sequence will be $\omega = (\sigma, \kappa)$ where $\sigma = (s_1 s_2 s_3)$ and $\kappa = (I_0 I_1 I_2)$ such that $I_0 = [0, 1.5)$, $I_1 = [1.5, 2.5)$ and $I_2 = [2.5, \infty)$.*

Given a timed state sequence $\omega = (\sigma, \kappa)$ with $\sigma = (s_1, s_2, \dots)$ and $\kappa = (I_1, I_2, \dots)$, which corresponds to a feasible solution of a VRP instance, the atomic propositions take unique values at any given time $t \in \mathbb{R}_{\geq 0}$. The value of the atomic proposition $p \in \Pi$ at time $t \in \mathbb{R}_{\geq 0}$ will be denoted by $s(t)[p_i]$.

Finally, let us introduce the following definition of a suffix of a timed state sequence which will be used in the definition of the semantics of MTL.

Definition III.7 (Suffix) *Let $\omega = (\sigma, \kappa)$ be a timed state sequence. For some $t \in I_i$, the suffix ω^t at time t is the timed state sequence $\omega' = (\sigma', \kappa')$ where $\sigma' = (s_i, s_{i+1}, s_{i+2}, \dots)$ and $\kappa = (I_i - t, I_{i+1} - t, I_{i+2} - t, \dots)$.*

B. Metric Temporal Logic: Syntax

The formulae of MTL are constructed using the atomic propositions along with the operators of MTL, which are to be defined shortly. Every formula of MTL is a proposition by itself, i.e., it is either `True` or `False` at some given time for a given feasible solution of the VRP, or equivalently for a given timed state sequence.

MTL includes the usual operators of propositional logic together with a set of temporal operators, which are bound with an interval. The syntax of MTL is defined as follows:

$$\phi ::= p | \neg\phi | \phi_1 \wedge \phi_2 | \phi_1 \, \mathcal{U}_I \phi_2, \tag{10}$$

where I is an interval, $p \in \Pi$ is an atomic proposition, ϕ, ϕ_1, and ϕ_2 are MTL formulae. $\neg$ is the negation operator,

$\wedge$ is the conjunction operator and $\mathcal{U}_I$ is the until operator. Informally speaking, the formula $\phi_1 \mathcal{U}_I \phi_2$ is true at time t if there is some time $t' \in t + I$ for which ϕ_2 is `True` and ϕ_1 holds to be true within the interval (t, t').

Even though the operators above are adequate to fully represent the MTL language, we define the following operators as well to improve readability of MTL formulae. Given the negation and conjunction operators, the operators *disjunction* ($\vee$), *implication* ($\Rightarrow$), and *equivalency* ($\Leftrightarrow$) can be defined as $\phi_1 \vee \phi_2 = \neg(\neg\phi_1 \wedge \neg\phi_2)$, $\phi_1 \Rightarrow \phi_2 = \neg\phi_1 \vee \phi_2$, and $\phi_1 \Leftrightarrow \phi_2 = (\phi_1 \Rightarrow \phi_2) \wedge (\phi_2 \Rightarrow \phi_1)$ respectively. Together with these operators given the temporal operator *until*, the temporal operators *eventually* ($\Diamond_I$), *always* ($\Box_I$) and *unless* $\mathcal{W}_I$ can be defined as $\Diamond_I \phi = \top \mathcal{U}_I \phi$, $\Box_I \phi = \neg\Diamond_I \neg\phi$, and $\phi_1 \mathcal{W}_I \phi_2 = \neg(\neg\phi_1 \mathcal{U}_I \neg\phi_2)$. Informally speaking, the formula $\Diamond_I \phi$ is `True` at some time t if ϕ holds to be `True` at some point time in the interval $t + I$. Similarly the formula $\Box_I \phi$ is `True` at some time t if ϕ holds to be `True` at all points in the interval $t + I$. Finally, the formula $\phi_1 \mathcal{W}_I \phi_2$ holds at time t if and only if either ϕ_1 is `True` throughout the interval or there exists a $t' > t$ for which ϕ_2 is `True` at t' and ϕ_2 is `True` during the interval $[t', t] \cap I$.

C. Metric Temporal Logic: Semantics

The formulae of MTL are interpreted over the timed state sequences. Let $\omega = (\sigma, \kappa)$ be a timed state sequence, then the semantics of MTL is given by the following recursive definition of the satisfaction relation $\models$.

$$\omega \models p \quad \text{iff} \quad s_0 \models p; \tag{11}$$

$$\omega \models \neg\phi \quad \text{iff} \quad \omega \not\models \phi; \tag{12}$$

$$\omega \models \phi_1 \wedge \phi_2 \quad \text{iff} \quad \omega \models \phi_1 \text{ and } \omega \models \phi_2; \tag{13}$$

$$\omega \models \phi_1 \mathcal{U}_I \phi_2 \quad \text{iff} \quad \exists t \in I, \omega^t \models \phi_2 \text{ and } \forall t' \in (0, t), \omega^{t'} \models \phi_1. \tag{14}$$

A timed state sequence ω is said to satisfy the formula ϕ if and only if $\omega \models \phi$. Then, the semantics of *disjunction*, *always* and *eventually* can be given as follows:

$$\omega \models \phi_1 \vee \phi_2 \quad \text{iff} \quad \omega \models \phi_1 \text{ or } \omega \models \phi_2; \tag{15}$$

$$\omega \models \Diamond_I \phi \quad \text{iff} \quad \exists t \in I, \omega^t \models \phi; \tag{16}$$

$$\omega \models \Box_I \phi \quad \text{iff} \quad \forall t \in I, \omega^t \models \phi. \tag{17}$$

IV. Vehicle Routing Problem with Metric Temporal Logic (VRPMTL)

A formal definition of the VRPMTL can be given as follows:

Problem IV.1 (VRPMTL) *A VRPMTL instance is a tuple $(\mathcal{K}, \mathcal{N}, \mathcal{L}, \mathcal{C}, \mathcal{T}, c, \Pi, \phi)$ where $\mathcal{K}$ is a set of vehicles (UAVs), $\mathcal{N}$ is a set of customers (targets), $\mathcal{L}$ is a set of initial depots (launch sites), $\mathcal{C}$ is a set of final depots (landing sites), $\mathcal{T}$ is a set of parameters indicating cost (or similarly time) to travel from one node to another for all the vehicles, c is a cost function of the parameters in set $\mathcal{T}$, Π is a set of atomic propositions and ϕ is an MTL formula defined on Π. The VRPMTL is to find a feasible solution such that*

- *the timed state sequence ω corresponding to this feasible solution satisfies ϕ at the initial time, i.e., $\omega^0 \models \phi$;*
- *the function c is minimized.*

The rest of the paper is devoted to the introduction of a MILP-based algorithm for solving VRPMTL instances under some assumptions on the structure of the MTL formula ϕ.

V. A MILP-based Formulation of VRPMTL

A. MILP Formulation of Atomic Propositions

Let us associate the following two variables to the evolution of an atomic proposition p: a binary variable ξ_p and a continuous variable τ_p. $\xi_p = 1$ if the atomic proposition p stays `False` forever; $\xi_p = 1$ if p becomes `True` at some point in time. Then, τ_p indicates the earliest time that p switches to `True`, if $\xi_p = 1$; it is meaningless if $\xi_p = 0$.

In the rest of this section, some examples of atomic propositions will be provided in the application domain. Consider, for example, the atomic proposition p, which states that "Target j is serviced by UAV k". In this case, the constraints on ξ_p and τ_p become $\xi_p = \sum_{i \in I} x_{ijk}$; $\tau_p = t_j$.

Next, consider the atomic proposition p which states that "Target j is serviced". Notice that the parameters $\xi_p \in \{0, 1\}$ and $\tau_p \in \mathbb{R}_{\geq 0}$ can be defined using the following linear constraints over the variables of the VRP formulation presented in Section II: $\xi_p = \sum_{i \in I} \sum_{k \in \mathcal{K}} x_{ijk}$; $\tau_p = t_j$.

Another example is the atomic proposition p, which states that "UAV k landed on landing site j". For this atomic proposition the linear constraints turn out to be $\xi_p = \sum_{i \in I} \sum_{j \in C} x_{ijk}$; $\tau_p = t_{jk}$.

Let us note the following final example. This time, the atomic proposition p states that "UAV k is launched from launch site i", for which the constraints on ξ_p and τ_p can be written as $\xi_p = \sum_{j \in \mathcal{J}} x_{ijk}$; $\tau_p = s_{ik}$. Notice that this proposition uses the loitering time at the launch site which corresponds to the launch time of the UAV.

B. MILP Formulation of MTL Formulae

This section is devoted to the introduction of a systematic procedure which constructs a set of linear inequalities over the variables ξ_{p_i} and τ_{p_i} for any given MTL formula ϕ in a specific structure to be presented shortly. Given some $\omega = (\sigma, \kappa)$ where $\sigma = (s_1, s_2, \dots)$ and $\kappa = (I_1, I_2, \dots)$, let $\xi_p = 1$ and $\tau_{p_i} \leq t$ if $s(t)[p_i] = \texttt{True}$, and $\xi_p = 0$ if $s(t)[p_i] = \texttt{False}$, for $\forall p_i \in \Pi$. Then, for any given timed state sequence ω, the aforementioned set of linear constraints are satisfied if and only if $\omega \models \phi$, where ϕ satisfies the following assumption.

Assumption V.1 *The temporal operators of MTL are applied only to atomic propositions and their negations.*

Even though this assumption seems constraining, many interesting complex temporal specifications already satisfy this assumption. Let us postpone this discussion until the examples and discuss the variety of specifications that can be represented in this way then.

The algorithm, which generates the set of MILP constraints from the MTL specification, runs in two phases.

In the first phase, it eliminates all the temporal operators from the formula, whereas in the second phase it considers the remaining non-temporal formula to construct the MILP constraints.

More precisely, in the first phase of the algorithm, for each temporal operator and the propositions it binds, a set of constraints is generated. Noting that, by Assumption V.1, each temporal operator is either applied to an atomic proposition p or its negation $\neg p$, this set of constraints can be defined considering the four cases of the binary operator *until*, and two cases of the unary operators *eventually* and *always*. Let us first consider the *until* operator. Let $p_1, p_2 \in \Pi$ be the set of atomic propositions with their corresponding variables ξ_{p_1}, τ_{p_1} and ξ_{p_1}, τ_{p_1}, respectively. Then, $p_1 \mathcal{U}_I p_2$ with $I = [a,b]$ is satisfied at time $t = 0$ if and only if $(\xi_{p_2} = 1, \tau_{p_2} \leq b) \wedge ((\tau_{p_2} \leq a) \vee (\xi_{p_1} = 1, \tau_{p_1} < \tau_{p_2}))$. For $\neg p_1 \mathcal{U}_I p_2$, the constraints are $(\xi_{p_2} = 1, \tau_{p_2} \leq b) \wedge ((\tau_{p_2} \leq a) \vee (\tau_{p_2} < \tau_{p_1}))$. For the cases $p_1 \mathcal{U}_I \neg p_2$ and $\neg p_1 \mathcal{U}_I \neg p_2$ the only constraint is $\tau_{p_2} \geq a$, which is same for both since both force p_2 to be `False` throughout the interval.

Given the interval $I = [a,b]$, the constraints corresponding to the formula $\Diamond_I p_1$ are $\xi_{p_1} = 1, \tau_{p_1} \leq b$, whereas the ones corresponding to the formula $\Diamond_I \neg p_1$ are $\tau_{p_1} > a$.

Considering the always operator, the constraints corresponding to the formula $\Box_I p_1$ are $\xi_{p_1} = 1, \tau_{p_1} \leq a$, and the constraints corresponding to the formula $\Box_I \neg p_1$ are $\tau_{p_1} \geq b$.

Every constraint above is of the form $a_j z \leq b_j$. In the next step, each such constraint is bound with a binary variable $y_j \in \{0,1\}$ which satisfies $y_j \leq 1 - \frac{1}{M}(a_j z - b_j)$; $y_j \geq -\frac{1}{M}(a_j z - b_j)$, where M is a big enough number. Notice that $y_j = 1$ if $a_j z \leq b_j$ holds and $y_j = 0$ otherwise.

In the second phase, all the temporal subformulae are replaced with the corresponding binary variables y_j in the formula ϕ, and we apply the following rules recursively to the remaining formula ϕ, composed of negations, conjunctions, and disjunctions of y_i. Initially, $l = m+1$ and then,

- for any negation of a single variable y_j a slack variable $y_l \in \mathbb{R}$, $0 \leq y_l \leq 1$ is defined, which satisfies $y_l = 1 - y_j$;
- for any conjunction of m single variables $y_1, \ldots, y_J$, a slack variable $y_l \in \mathbb{R}$, $0 \leq y \leq 1$ is defined which satisfies $y_l \leq y_j$, $j \in \{1, \ldots, J\}$; $y_l \geq \sum_{j=1}^{J} y_j - (J-1)$;
- for any disjunction of m single variables $y_1, \ldots, y_J$, a slack variable $y_l \in \mathbb{R}$, $0 \leq y \leq 1$ is defined which satisfies $y_l \leq \sum_{J=1}^{J} y_j$; $y_l \geq y_j$, $j \in \{1, \ldots, J\}$.

Then, at the end of each step, the variable y_l is substituted in the formula instead of variables and the operator it represents; and l is incremented by one. This recursive procedure is continued until ϕ is becomes a single variable, as in $\phi = y_L$. Finally, the following constraint is added to the formulation: $y_L = 1$, which states that the formula is `True` at the initial time, i.e., for any feasible solution of the VRP instance the corresponding timed state sequence satisfies $\omega^0 \models \phi$.

VI. Multi-UAV Mission Planning Applications

In this section we present a simple example in order to illustrate the framework presented in the paper. Let us consider a scenario with three UAVs, one launch site, two landing sites, and five targets. The spatial distribution of the targets and the sites are shown in Figure 1. The three vehicles K_1, K_2, and K_3 can travel at 15, 18, and 20 mph respectively. The servicing time of each of the targets is 0.05 hours. When calculating the time required to from a node to another we consider rectilinear distances which is assumed to be a good model for urban environments [24].

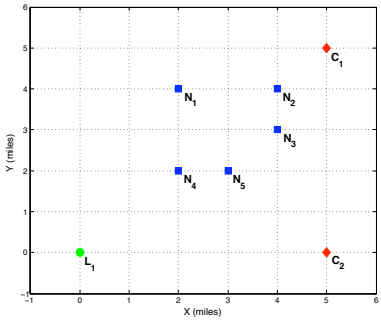

Fig. 1. Map of the example mission planning scenario.

The objective in this mission is to either service targets N_1, N_4, N_5 within the first 1.5 hours, or to service targets N_2, N_3 within 0.7 hours. If the former option is taken, then eventually K_1 must land on the landing site C_2, after target N_4 is serviced, even if it does not service any of the targets. Furthermore, target N_4 can only be serviced after 0.4 from the beginning of the mission only by the UAV K_2. If the latter option is taken, servicing of the second target must always be avoided in the first 0.6 hours. Moreover, target N_3 must be serviced before target N_2. To model this problem in MTL, let us define the following atomic propositions,

- p_1: Target N_1 is serviced;
- p_2: Target N_2 is serviced by UAV K_2;
- p_3: Target N_3 is serviced by UAV K_3;
- p_4: Target N_4 is serviced by UAV K_2;
- p_5: Target N_5 is serviced;
- p_6: UAV K_1 is on landing site C_2,

using which the specification can be written as

$$\begin{aligned}\phi = \quad & (\Diamond_{[0,1.5]} p_1 \wedge \Diamond_{[0,1.5]} p_4 \wedge \Diamond_{[0,1.5]} p_5 \\ & \wedge \Diamond_{[0,1.5]} p_6 \wedge \Box_{[0,0.4]} \neg p_4 \wedge (\neg p_6) \mathcal{U} p_4) \\ & \vee (\Diamond_{[0,0.7]} p_2 \wedge (\neg p_2) \mathcal{U}_{[0,0.7]} p_3 \wedge \Box_{[0,0.6]} \neg p_2)\end{aligned}$$

The optimal solution for the mission is shown in Figure 2. Notice that K_2 services the targets N_1, N_2, and N_3 and K_1 directly flies to landing site C_2, even though it does not service any of the targets. The landing times of K_1 and K_2 are 0.4334 and 0.8667 hours, and the servicing times of targets N_5, N_4, N_1 are 0.3278, 0.4334, and 0.5945, respectively. K_1 loiters around the launch site L_1 before starting its route for 0.0501 hours. Notice that the solution also satisfies the constraints that the target N_4 is not serviced within the first 0.4 hours of the mission. Notice also that the optimal solution makes UAV K_1 loiter for a while at the launch site in order to delay its arrival to C_2 and satisfy the constraint that UAV V_1 is not landed on C_2 before target N_4 is serviced.

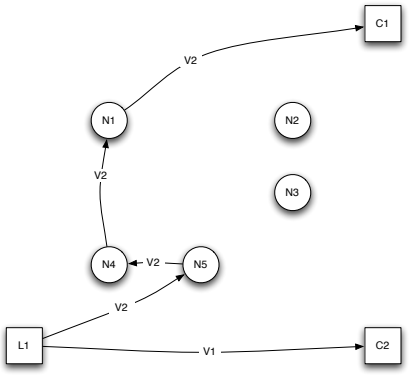

Fig. 2. Optimal scheduling for MTL specification ϕ

As a slightly different example let us consider a scenario in which VRPMTL instance is the same except that the MTL specification changed to ϕ' which differs from ϕ by requiring that N_4 can not serviced within the first 0.5 hours of the mission instead of 0.4 hours. The corresponding optimal solution of the mission is shown in Figure 3. Landing times of UAVs K_2 and K_3 are 0.7611 and 0.6, the servicing times of the targets N_2 and N_3 are 0.6 and 0.4, respectively. In this solution UAV K_2 loiters at the launch location L_1 for 0.1056 hours before starting the mission.

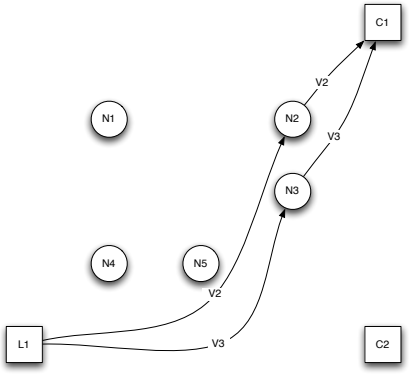

Fig. 3. Optimal scheduling for MTL specification ϕ'

VII. Conclusions

This paper proposed a novel variant of Vehicle Routing Problem which we have called Vehicle Routing Problem with Metric Temporal Logic Specifications (VRPMTL) and provided a novel MILP-based algorithm that solves the problem to optimality. Applications to multi-UAV mission planning have been considered in which high level complex mission tasks were specified naturally via Metric Temporal Logic. Our future work will cover consideration of similar formalisms for multi-agent agent systems in a VRP setting. We will also consider effective heuristics to solve VRPMTL for large-scale problems.

Acknowledgments

This work was supported in part by the Michigan/AFRL Collaborative Center on Control Sciences. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the supporting organizations.

References

[1] M. Alighanbari, Y. Kuwata, and J.P. How. Coordination and control of multiple UAVs with timing contraints and loitering. In *American Control Conference*. IEEE, 2003.

[2] R. Alur, C. Courcoubetis, and D. Dill. Model-checking for real-time systems. In *Logic in Computer Science*, pages 414–425, 1990.

[3] R. Alur, T. Feder, and T.A. Henzinger. The benefits of relaxing punctuality. *Journal of the Association for Computing Machinery*, 43(1):116–146, 1996.

[4] R. Alur and T.A. Henzinger. A really temporal logic. *Journal of the Association for Computing Machinery*, 41(1):181–204, 1994.

[5] E. Angelelli and M.G. Speranza. The applications of a vehicle routing model to a waste-collection problem: two case studies. *Journal of the Operational Research Society*, 53:944–952, 2002.

[6] J. Bellingham, M. Tillerson, A. Richards, and J.P. How. Multi-task allocation and path planning for cooperating UAVs. In S. Butenko, R. Murphey, and P.M. Pardalos, editors, *Cooperative Control: Models, Applications and Algorithms*. Kluwer Academic Publishers, 2001.

[7] O. Braysy and M. Gendreau. Vehicle routing problem with time windows, part i: Route construction and local search algorithmsh time windows, part i: Route construction and local search algorithms. *Transportation Science*, 39(1):104–118, February 2005.

[8] O. Braysy and M. Gendreau. Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation Science*, 39(1):119–139, February 2005.

[9] E.M. Clarke, O. Grumberg, and D.A. Peled. *Model Checking*. MIT Press, Cambridge, MA, 1999.

[10] T.A. Henzinger, X. Nicollin, J. Safakis, and S. Yovine. Symbolic model checking for real-time systems. In *Logic in Computer Science*, pages 394–406, 1992.

[11] T.A. Henzinger, J.F. Raskin, and P.Y. Schobbens. The regular real-time languages. *Lecture Notes in Computer Science, LNCS 1443*, pages 580–591, 1998.

[12] S. Karaman and E. Frazzoli. Complex mission optimization using linear temporal logic. In *American Control Conference*, 2008.

[13] S. Karaman and E. Frazzoli. Vehicle routing using linear temporal logic: Applications to multi-UAV mission planning. In *AIAA Conference on Guidance Navigation and Control*, 2008.

[14] R. Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2:255–299, 1990.

[15] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems*. Springer-Verlag, 1992.

[16] H. Onal, B.M. Jaramillo, and M.A. Mazzoco. Two formulations of the vehicle routing problem: An emprical application and computational experience. *Logistics and Transportation Review*, 32:177–190, 1996.

[17] J. Ouaknine and J. Worrell. On the decidability of metric temporal logic. In *Logic in Computer Science*, pages 188–197, 2005.

[18] A. Pnueli. The temporal logic of programs. In *18th annual IEEE-CS Symposium on Foundations of Computer Science*, pages 46–57, 1977.

[19] A. Richards, Y. Kuwata, and J.P. How. Experimental demonstrations of real-time MILP control. In *Guidance, Navigation, and Control Conference*. AIAA, 2003.

[20] R. Ruiz, C. Maroto, and J. Alcaraz. A decision support system for a real vehicle routing problem. *European Journal of Operational Research*, 153:593–606, 2004.

[21] C. Schumacher, P. Chandler, M. Pachter, and L. Pachter. UAV task assignment with timing constraints via mixed-integer linear programming. In *AIAA 3rd Unmanned Unlimited Technical Conference, Workshop and Exhibit*. AIAA, 2004.

[22] C. Schumacher, P.R. Chandler, M. Pachter, and L.S. Patcher. Optimization of air vehicles operations using mixed-integer linear programming. *Journal of the Operational Research Society*, 58:516–527, 2007.

[23] P. Toth and D. Vigo. *The Vehicle Routing Problem*. SIAM, 2002.

[24] A.L. Weinstein and C. Schumacher. UAV scheduling via the vehicle routing problem with time windows. Technical Report AFRL-VA-WP-TP-2007-306, Air Force Research Laboratory, January 2007.