Automated route planning for milk-run transport logistics using model checking

Takashi KITAMURA
National Institute of Advanced Industrial Science and Technology (AIST)
Email: t.kitamura@aist.go.jp

Keishi OKAMOTO
Sendai National College of Technology
okamoto@sendai-nct.ac.jp

***Abstract*—We develop a specification framework for milk-run transport logistics, applying model checking. The framework adopts LTL (Linear Temporal Logic) as a specification language for flexibly specifying complex delivery requirements in the setting of milk-run logistics. The framework defines the notion of "optimal truck routes" which satisfy given delivery requirements in a route map, by applying the bounded semantics of LTL. We develop an automated route planner based on the framework using the NuSMV model checker as an early implementation. We evaluate the feasibility of the implementation design by analyzing its computational complexity and showing experimental results.**

***Keywords*-transport logistics, model checking, route planning**

I. Introduction

Milk-run transport logistics, which refers to the means of transportation where a single truck cycles around multiple suppliers to collect or deliver freights, is one of the most efficient and popular approaches to improve logistic operations. Recently, a variety of industries, e.g., food, automobile manufacturing, military, as well as the dairy industry, have adopted the milk-run approach to make their logistics operations more efficient. However, compared with the existing (non-efficient) logistics, using milk-run logistics often tends to be complex, which is one of the main barriers preventing wider prevalence of the approach in industry.

Focusing on the complexity, Satoh [1], [2] proposed a novel and practical framework for such milk-run transport logistics. The framework introduces a formal specification language for the users (customers and suppliers) to specify truck routes in milk-run logistics settings, and *a mechanism for selecting appropriate trucks* according to the specified truck routes. The framework is realized based on process algebra; the specification language is designed based on CCS (Calculus of Communicating System)[3], and the mechanism for selecting appropriate trucks is realized by using the notion of bisimulation relations.

Inspired by [1], [2] we develop a framework for *automated route planning* for such milk-run logistics. The framework given delivery requirements and a route map deals with *optimal truck routes* on the route map that satisfy the given delivery requirements. We apply a model checking approach to realize this framework. We adopt LTL (Linear Temporal Logic) [4], [5] as a specification language for flexibly specifying the delivery requirements in milk-run logistics, which tend to be complex w.r.t. the location order of truck routing. Finally, as an early implementation, we develop an automated route planner for the framework using the NuSMV model checker. And we evaluate the feasibility of the implementation design by analyzing its computational complexity and showing experimental results.

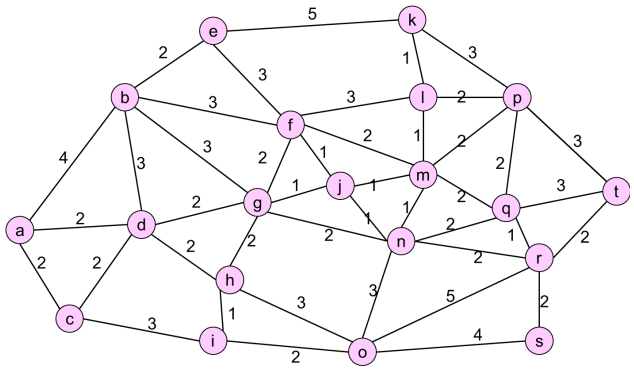


Figure 1. An example of route map

We end this section with an outline of the paper: the next section explains the background of the framework we develop, Section II briefly reviews LTL, Section III explains the details of our framework, Section IV explains our early implementation of the framework for automated route planning using NuSMV, and the conclusion and discussion of future work follow in the last section.

II. Background

A. Example scenario

Consider a route map as weighted, undirected graph, where the nodes express locations, the edges express routes, and the weights on edges express the costs of a truck moving on the route segment. Figure. 1 shows an example of such a route map.

"Delivery requirements" are the conditions for delivery regarding the truck routes; i.e., trucks have to satisfy these conditions on their routes through route map. A main characteristic of delivery requirements in milk-run logistics typically appears in the complex order of locations which trucks need to visit. This is a consequence of an important

978-0-7695-4893-7/12 $26.00 © 2012 IEEE
DOI 10.1109/ICNC.2012.44

feature of milk-run logistics: trucks are shared by multiple users (i.e., suppliers and customers). Trucks collect freights at one or more source points and deliver them to one or more destination points on their way, visiting the source points before the destination points. Typical delivery requirements in milk-run logistics are exemplified as follows:

i) A truck visits location b and then k, and after that visits location a or i and then s.
ii) A truck visits location e and then i, but between them it should not go through f and o.
iii) A truck visits location o and then t and p and then c, while visiting from o to t it should not go through r.

Given such a route map and delivery requirements, we discuss the truck routes on the route map that satisfy and violate the given requirements. Further, we can discuss the truck routes with the best efficiency, i.e., the truck routes satisfying the requirements with the smallest cost, which we call "*optimal truck routes*".

Assume that a truck has sufficient carrying capacity, and it starts at location a. Table. I shows three trucks routes on the route map that satisfy and violate the delivery requirements on different routes. Each truck route is represented as a sequence of locations, where locations are connected by $\rightarrow$. The sequence of the locations is the order in which the trucks visit the locations. The (natural) number attached to each arrow expresses the cost for a truck moving from previous to next location, and the number on each location expresses the accumulated cost to reach the location on the truck route, given the route map and sequence.

Observe that truck route (1) and (2) satisfy the above delivery requirements (i) - (iii), and truck route (3) does not. Also, we consider that truck route (1) is more efficient than (2); (1) requires cost 39 on its route to satisfy the given requirements while (2) requires 42. In fact, (1) is an *optimal truck route*, which is a truck route on the route map that satisfies the delivery requirements with the minimum cost. Note that the cost of a truck route is the sum of the weights on edges in its route, instead of the number of locations which it goes through. Accordingly, though truck route (2) visits less locations on its route than (1), (1) is considered to be more efficient.

B. Requirements for the framework

With this example scenario, we clarify the main requirements of the framework for automated route planning for milk-run logistics. We distinguish *general* requirements for milk-run logistics and *specific* ones for our framework of automated route planning. We share *general* requirements for milk-run logistics in common from [1], [2] as follows:

- *Trucks may be shared by multiple suppliers and customers, so that they collect products at one or more source points and deliver the products at one or more destination points on their way. The trucks need to visit the source points before they visit the destination points. The framework therefore needs to specify the order in which trucks call at various points.*
- *The routes taken by trucks may also affect product quality. For example, foods should be transported by the shortest route possible to keep their freshness, and perishable foodstuffs should be picked up later than preservable foodstuffs and taken to a food processor or consumer.*
- *Pallets or boxes that contain multiple products are considered as transport units in many current logistics systems, rather than as individual products. These types of containers may have multiple destinations and the receivers may take only some of the products in the container when it arrives at their point.*

Besides the *general* requirements for milk-run logistics, we assume the following *specific* ones for our framework:

- The framework, given a route map and delivery requirements, provides a mechanism to find optimal truck routes, which are truck routes satisfying the given delivery requirements with the minimum cost. Also, the framework provides a mechanism, given a cost, to find a truck route with that cost, and additionally to check if there exists a truck route which satisfies the given requirements with that cost.
- To keep generality, route maps are regarded as weighted undirected graphs, following the precedent of general routing problems such as [6]. Weights are imposed on the edges, expressing a notion of cost, interpreted as, e.g., time, money, and emission of CO_2, for trucks' moving on the edges.
- Delivery requirements in milk-run logistics are complex w.r.t. the order of locations which trucks visit. Also, the requirements vary temporally, since users have different requirements each week, each day, and or hour, depending on their use. To address these aspects, the framework should take a language-based approach for specifying delivery requirements; i.e, it should provide a language with which users can flexibly specify delivery requirements.

C. Basic approach

Our aim is to develop a framework for milk-run logistics to satisfy the above requirements. We realize the framework applying model-checking techniques. Our basic approach is to specify delivery requirements with LTL and route maps with Kripke models, and apply model checking algorithms to find truck routes that best satisfy the requirements. That is, we interpret automated route planning as the following model checking problem:

$$\textit{route map} \models \textit{Delivery requirements}.$$

(1) $a^0 \rightarrow^4 b^4 \rightarrow^2 e^6 \rightarrow^2 b^8 \rightarrow^3 d^{11} \rightarrow^2 h^{13} \rightarrow^1 i^{14} \rightarrow^2 o^{16} \rightarrow^3 n^{19} \rightarrow^2 q^{21} \rightarrow^2 t^{23} \rightarrow^3 p^{26} \rightarrow^2 l^{28} \rightarrow^1 k^{29} \rightarrow^1 l^{30} \rightarrow^1 m^{31} \rightarrow^1 j^{32} \rightarrow^1 g^{33} \rightarrow^2 d^{35} \rightarrow^2 a^{37} \rightarrow^2 c^{39}$

(2) $a^0 \rightarrow^4 b^4 \rightarrow^3 d^7 \rightarrow^2 h^9 \rightarrow^2 o^{11} \rightarrow^3 n^{14} \rightarrow^3 m^{17} \rightarrow^1 n^{18} \rightarrow^1 q^{19} \rightarrow^2 t^{21} \rightarrow^2 p^{23} \rightarrow^3 l^{25} \rightarrow^2 k^{27} \rightarrow^1 e^{28} \rightarrow^5 b^{33} \rightarrow^4 a^{37} \rightarrow^4 c^{39} \rightarrow^3 i^{42}$

(3) $a_0 \rightarrow^2 d_2 \rightarrow^2 h_4 \rightarrow^3 o_7 \rightarrow^3 h_{10} \rightarrow^2 d_{12} \rightarrow^2 b_{14} \rightarrow^2 e_{16} \rightarrow^3 f_{19} \rightarrow^1 m_{20} \rightarrow^1 l_{21} \rightarrow^1 k_{22} \rightarrow^1 l_{23} \rightarrow^2 p_{25} \rightarrow^2 t_{27} \rightarrow^3 q_{30} \rightarrow^2 n_{32} \rightarrow^2 g_{34} \rightarrow^2 d_{36} \rightarrow^2 c_{38}$

Table I
DIFFERENT TRUCK ROUTES (1) (2) AND (3) TO SATISFY/VIOLATE THE DELIVERY REQUIREMENTS

As the framework takes a language-based approach to specify delivery requirements, we adopt LTL for the purpose. LTL is a logical language that can specify complex temporal properties by means of ordered events to happen. Hence, it makes an appropriate basis for specifying complex and various delivery requirements in milk-run logistics. For example, delivery requirement (i) (ii) and (iii) can be specified with LTL as;

i) $\mathbf{F}(b \wedge \mathbf{F}(k \wedge \mathbf{F}(a \vee (i \wedge \mathbf{F}s))$
ii) $\mathbf{F}(e \wedge \neg(f \vee o)\mathbf{U}i)$
iii) $\mathbf{F}(o \wedge (\neg r\mathbf{U}t) \wedge \mathbf{F}(p \wedge \mathbf{F}c))$

By applying the semantics of LTL, the framework preserves formal accounts of the satisfaction or violation of truck routes given complex delivery requirements. To realize the framework for automated route planning, we apply *bounded semantics* and *bounded model checking (BMC)* proposed by [4], [5]. Due to this approach, the framework can impose a cost bound into the analysis of a truck route satisfying or violating delivery requirements. Consequently, the framework can formally define "optimal truck routes", which are truck routes that satisfy the delivery requirements with the minimum cost. Further, we implement the framework using NuSMV, an off-the-shelf BMC model checker.

III. BOUNDED LTL MODEL CHECKING[4], [5], [7], [8]

Bounded semantics of LTL: Let AP be a set of atomic propositions., ranged over by $p, q, \cdots$. Then LTL formulas over AP are defined recursively as follows: atomic propositions are LTL formulas; and if ϕ_1 and ϕ_2 are LTL formulas so are $X\phi$ (neXt), $\phi_1 U \phi_2$ (ϕ_1 Until ϕ_2), $\phi_1 \vee \phi_2$ and $\neg\phi_1$. A Kripke model M over AP is a quadruple $M = (S, I, T, L)$ where S is a finite set of states, $I \subseteq S$ is a finite set of initial states, $T \subseteq S \times S$ is the transition relation and $L : S \rightarrow 2^{AP}$ is the labeling function.

The LTL semantics is defined by way of paths of a Kripke model M. Besides, in bounded semantics, paths with loop-back and without loop-back are considered separately. Here, we only review the case for paths without loop-back since it is of the only situation necessary in our framework. A path π in M is a sequence $\pi = (s_0, s_1, ...)$ of states, given in an order that respects the transition relation of M. For $i < |\pi|$, $\pi(i)$ denotes the i-th state s_i in the sequence, and $\pi_i = (s_i, s_{i+1}, \cdots)$ denotes the suffix of π starting with state s_i. For $k \geq 0$, let π be a path without loop-back, and ϕ be a LTL formula. Then π satisfies ϕ with bound k (written as $\pi \models_k \phi$) iff $\pi \models_k^0 \phi$ where $\pi \models_k^i \phi$ is defined in Table. II.

$\pi \models_k^i p$	**if** $p \in L(\pi(i))$
$\pi \models_k^i \neg p$	**if** $p \notin L(\pi(i))$
$\pi \models_k^i \phi_1 \wedge \phi_2$	**if** $\pi \models_k^i \phi_1$ *and* $\pi \models_k^i \phi_2$
$\pi \models_k^i \phi_1 \vee \phi_2$	**if** $\pi \models_k^i \phi_1$ *or* $\pi \models_k^i \phi_2$
$\pi \models_k^i \mathbf{X}\phi$	**if** $i < k$ and $\pi_1 \models_k^{i+1} \phi$
$\pi \models_k^i \mathbf{F}\phi$	**if** $\exists j, i \leq j \leq k.\pi \models_k^j \phi$
$\pi \models_k^i \mathbf{G}\phi$	*is always false.*
$\pi \models_k^i \phi_1 \mathbf{U} \phi_2$	**if** $\exists j.i \leq j \leq k.\pi \models_k^j \phi_2$ and $\forall h, i \leq h < j.\pi_i \models_k^h \phi_1$
$\pi \models_k^i \phi_1 \mathbf{R} \phi_2$	**if** $\exists j.i \leq j \leq k.\pi \models_k^j \phi_1$ and $\forall h, i \leq h < j.\pi_i \models_k^h \phi_2$

Table II
BOUNDED LTL SEMANTICS WITHOUT A LOOP

It is defined that a Kripke model M satisfies a LTL formula ϕ, written $M \models \phi$, if $\pi \models \phi$ for all paths π of M; i.e., $M \models \phi$ *if* $\forall \pi \in M.\pi \models \phi$.

Bounded model checking using SAT: One advantage of BMC is that we can discuss the semantics taking the length of paths into account. As an application, we can discuss paths that satisfy/violate a LTL formula with the shortest length. BMC problems are typically solved by reducing SAT problems, therefore the complexity of BMC of this approach is determined using the number of propositional variables to appear in formula by SAT encoding [7], [8]. The translation results in $O(k * |log(S)| + (k+1)^2 * |\phi|)$ variables, where S is the number of states in model M, k is the bound, and $|\phi|$ is the length of ϕ.

IV. AN AUTOMATED ROUTE PLANNING FRAMEWORK FOR MILK-RUN LOGISTICS

In this section we explain our framework for an automated route planning framework for milk-run logistics.

A. Semantics of delivery requirements w.r.t. route maps

First, we formally define what it means for a truck route in a route map to satisfy given delivery requirements, by providing a formal semantics of delivery requirements (specified by LTL) w.r.t. route maps. Though the semantics

is designed based on bounded LTL semantics w.r.t. Kripke models, its significance is emphasized to accurately explain the framework. Also, our early implementation for the framework explained in later sections is developed based on the formal foundation. We start by defining route maps:

Definition 1 (route map): A *route map* M is a pair $M = (L, R)$, where L is a finite set of locations and $R(\subseteq (L \times L) \times \mathcal{N}^+)$ is a finite set of *routes* where $\mathcal{N}^+$ is the set of positive integers, satisfying the conditions: (1) If $((a, b), n) \in R$ and $((a, b), n') \in R$ then $n = n'$, and (2) If $((a, b), n) \in R$ then $((b, a), n) \in R$. □

Like LTL semantics w.r.t. Kripke models, LTL semantics w.r.t. route maps are given in two steps; i.e., first semantics w.r.t. a path of a route map is given, then based on that, semantics w.r.t. route maps is given. To this end, the notion of truck routes, which we also call *paths*, on route maps is defined.

Definition 2 (Paths of a route map): A path π of a route map $\mathcal{M}$ is a sequence of pairs of a location and a natural number; i.e., $\pi = ((a_1, k_1), (a_2, k_2), (a_3, k_3), \cdots)$, where the natural number expresses the accumulated cost to reach the associated location, given in an order that aligns with the route map M. $\pi(k)$ denotes the location name paired with cost k in the path. Note that $\pi(k)$ may not be defined for all costs k. Thus, we consider $\pi(k)$ as a partial function, which, given a cost, returns a location name. $\pi_{\mathrm{l}}(i)$ and $\pi_{\mathrm{k}}(i)$ denote the location and the accumulated cost at the i-th pair in path π, respectively. For a brief notation, we may also write a path as "$\pi_{\mathrm{l}}(1)^{\pi_{\mathrm{k}}(1)} \rightarrow^{n_1} \pi_{\mathrm{l}}(2)^{\pi_{\mathrm{k}}(2)} \rightarrow^{n_2} \pi_{\mathrm{l}}(3)^{\pi_{\mathrm{k}}(3)} \ldots$", where $n_i = \pi_{\mathrm{k}}(i+1) - \pi_{\mathrm{k}}(i)$. □

The path representations in Table. I follow this definition using the brief notion.

Next the semantics of delivery requirements w.r.t. paths of a route map is defined. It is designed as a bounded semantics: i.e., it introduces a natural number to express a cost bound, and defines a path that satisfies delivery requirements taking the cost bound into account.

Definition 3 (Bounded semantics w.r.t. paths of route maps): Assume M is a route map, π a path of M, and ϕ a LTL formula. For $k \geq 0$, we define that π satisfies ϕ with bound k, denoted by $\pi \models_k \phi$, if $\pi \models^0_k \phi$, where $\pi \models^i_k \phi$ is defined as Table. II, except for the following:

- for $\pi \models^i_k l$ and $\pi \models^i_k \neg l$, $\pi \models^i_k$ if $\pi(i) = l$ and $\pi \models^i_k \neg l$ if $\pi(i) \neq l$, respectively, and
- for $\pi \models^i_k$, $\pi \models^i_k \mathbf{G}\phi$ if $\forall j, i \leq j \leq k . \pi \models^j_k \phi$. □

For the **G** operator, due to this leap from the original bounded semantics of LTL w.r.t. Kripke models, we acquire duality between **G** and **F**, i.e., $\neg\mathbf{F}\phi \equiv \mathbf{G}\neg\phi$, in the context of the bounded semantics. We will use this property of the semantics in our implementation explained in section IV.

Thus, the semantics of LTL w.r.t. route map M is defined. Since our interest is to find some truck-routes to satisfy the given delivery requirements, we define that if any path of the route map satisfies the delivery requirements, then the satisfaction relation holds.

Definition 4 (Semantics w.r.t. a route map): Let M be a route map and ϕ a LTL formula. We write $M \models^\exists_k \phi$, if $\pi \models^\exists_k \phi$ for *some* paths π of M, □

The LTL semantics, in general, can be classified from the existential and universal viewpoint; i.e., determining whether an LTL formula ϕ is existentially/universally valid in a given model is called existential/universal model checking (EMC/UMC), respectively. For clarity, we use the notational convention of quantifier symbols $M \models^\exists \phi$ and $M \models^\forall \phi$ to denote them.

B. Examples of specification framework

In section II, we have shown several examples of delivery requirement specification by LTL. Here we demonstrate the satisfaction relation $\pi \models^\exists_k \phi$. As mentioned, $\pi_{(1)}$ satisfies delivery requirements (i), (ii) and (iii). This exactly means that the initial segment of the path, whose cost cumulatively sums to 39, satisfies the delivery requirements. Here we demonstrate that $\pi_{(1)}$ satisfies delivery requirement (iii), showing that such a path satisfies (iii) with a cost bound of 39, as follows:

$\pi_{1)} \models_{39} \mathbf{F}(o \wedge (\neg r \mathbf{U} t) \wedge \mathbf{F}(p \wedge \mathbf{F}c))$
if $\pi_{1)} \models^{16}_{39} o_{\mathbf{True}} \wedge \pi_{1)} \models^{16}_{39} \neg r \mathbf{U} t \wedge \mathbf{F}(p \wedge \mathbf{F}c)$
if $\pi_{1)} \models^{16}_{39} \neg r \mathbf{U} t \wedge \pi_{1)} \models^{16}_{39} \mathbf{F}(p \wedge \mathbf{F}c)$
if $(\forall n . 17 \leq n \leq 22 \wedge \pi_{1)} \models^n_{39} \neg r)_{\mathbf{True}} \wedge \pi_{1)} \models^{23}_{39} t_{\mathbf{True}} \wedge \pi_{1)} \models^{16}_{39} \mathbf{F}(p \wedge \mathbf{F}c)$
if $\pi_{1)} \models^{26}_{39} p \wedge \mathbf{F}c$
if $\pi_{1)} \models^{26}_{39} p_{\mathbf{True}} \wedge \pi_{1)} \models^{26}_{39} \mathbf{F}c$
if $\pi_{1)} \models^{39}_{39} c_{\mathbf{True}}$
if **True**

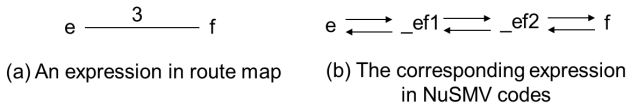
On the other hand, paths that have truck route (2) in its initial segment do not satisfy delivery requirement (i) - (iii) at the same time within cost bound 39; i.e., given such a path $\pi_{2)}$ then $\pi_{2)} \not\models_{39}$ (i) $\wedge$ (ii) $\wedge$ (iii). This is because the minimum cost of truck movement to satisfy the delivery requirements is 42, and this exceeds the allowed cost, given here as 39. That means $\pi_{2)} \models_{42}$ (i) $\wedge$ (ii) $\wedge$ (iii).

C. Optimal truck routes

Optimal truck routes, which are paths satisfying given delivery requirements with the minimum cost, are defined as:

Definition 5 (Optimal truck routes): A path π of a route map M is optimal w.r.t. a given delivery requirement ϕ if it satisfies the following conditions: (1) $\pi \models_i \phi$ for some i, and (2) if $\pi' \models_j \phi$ for some π' in M and for some j then $i \leq j$. □

Example 1 (Optimal truck routes): Take the example in Section II. As already mentioned, there are several truck routes of the route map that satisfy the given delivery requirements such as (1) and (2). And among them truck routes (1) is an optimal path in terms of Def. 5. □



(a) An expression in route map (b) The corresponding expression in NuSMV codes

Figure 2. How to encode weights of route map in NuSMV codes

V. Implementing an automated route planner for the milk-run logistics framework using the NuSMV model checker

This section explains our early implementation of an automated route planning system for the milk-run logistics framework developed in the previous sections. The system, given a route map and delivery requirements (specified by LTL), automatically finds an optimal truck route that satisfies the requirements based on the framework. We use the NuSMV model checker [9] [10] to realize such a system. Several gaps to realize the system by NuSMV, and techniques to bridge them are explained. Also we show experimental results of the implementation as its evaluation.

A. Encoding route maps into NuSMV codes

In applying NuSMV to realize a system for the framework, the route maps are encoded into NuSMV codes. In encoding route maps to NuSMV codes, one gap we need to bridge is how to encode the weights of edges in route maps in NuSMV codes so that the model checking algorithm of NuSMV can find optimal paths.

We realize this by introducing extra nodes, which we call *dummy nodes*, to express the weights. Consider, for example, a route (i.e., edge) connecting two locations with weight "3". Figure. 2 shows how the weight of each route in the route map (on the left) is expressed in the NuSMV codes using two dummy nodes of _ef1 and _ef2 (on the right). That is, the weight of n of a route in the route map is expressed by preparing $n-1$ dummy nodes between the two locations.

Also, it is important to notice that, due to this implementation design, the *neXt* operator ($X\phi$) is excluded from the LTL language set, which is for specifying delivery requirements, in this implementation.

B. Bridging gaps between UMC and EMC

The second gap lies between EMC, which our framework assumes, and UMC, which NuSMV is based on [9], [10]. In EMC, only one path of a model that satisfies a formula is required to prove the satisfaction relation between the model and the formula, which we may call the "witness". In UMC which NuSMV assumes, it is required that all the paths of a model satisfy a given formula, for their satisfaction relation to hold. That is, the existence of a path that fails to satisfy the formula violates the satisfaction relation, which is a so-called *counter-example*.

Here, an important observation for bridging the gap is that due to the semantics design, which keeps duality in the bounded semantics setting in Definition 4, finding a witness in EMC corresponds to finding a counter-example to the negation form of the formula in UMC. Accordingly, to build an automated route planner for the framework using NuSMV, it suffices to apply the negation form of the LTL formula to NuSMV, and in this way a generated counter-example is a witness and hence is a truck route that satisfies the delivery requirements.

C. Finding an optimal path using NuSMV

A straightforward way to find an optimal truck route using NuSMV, as defined in Def. 5 using NuSMV is to apply BMC in the above-explained setting. This is because optimal paths are defined as truck routes with the minimum weights in Def. 5 and BMC can find a counter-example with the shortest length using the breadth-first nature of SAT search procedures. In doing so, we use the "check_ltlspec_bmc_onepb" command, instead of the standard BMC process in NuSMV. In addition, we give natural numbers k to the command to specify the length of counter-examples (hence the length of truck route), and "X" for the option "l" to obtain paths with "no loop-back". These are required in order to find a truck route without loop-back, as otherwise NuSMV would find such truck routes that do not comply with Def. 5 due to its several notions of paths with the shortest length.

D. Computational complexity of the implementation design

We analyze computational complexity of this implementation design for the framework. As mentioned in Section III, the complexity of BMC using SAT is analyzed via the number of propositional variables to appear by SAT encoding, represented as $O(k * |log(S)| + (k+1)^2 * |\phi|)$, where S is the number of states in model M, k is the bound, and $|\phi|$ is the length of ϕ. Since the implementation encodes route maps into Kripke models, here we need to analyze the number of states accompanied by the encoding. In encoding, dummy locations to express the weights of a route map as well as locations are encoded to the states of Kripke model. Hence the number of states of a Kripke model encoded from route map M, which we consider as the size of route map M and denote as $size(M)$, is given as the sum of the number of locations (S) and dummy locations; i.e., $size(M) = |Loc| + \sum\{n-1 \mid ((a,b),n) \in R\}$. That is, the number of states in the Kripke models due to the encoding increases in a linear manner, and hence the number of propositional variables appearing in the formula encoded is $O(k * |log(size(M))| + (k+1)^2 * |\phi|)$.

E. Experimental results

As an experiment to demonstrate the feasibility of the implementation design, we show benchmark results based on the analysis of computational complexity above. From the complexity analysis, it is known that cost bounds (i.e., the length of paths) and the length of the LTL formula

cost bound	The length of formula ϕ $length(\phi)$ 20	30	40	50	60	70
k=20	4s	3s	4s	3s	4s	4s
k=30			68s	66s	2m40s	2m39s
k=40						277m58s
k=50						
k=60						

The experiments were conducted on a machine with an AMD Dual-Core Opteron 2220 CPU @2.8GHz, 33GB of RAM and Debian Linux 5.0.10.

Table III
EXPERIMENTAL RESULTS FOR FINDING OPTIMAL TRUCK ROUTING.

are the main contributors to the computational cost. Hence the experiment is designed based on interactions of these two factors where the size of route map is fixed at 200. Also, due to the advantageous property of the breadth-first search in BMC, we can apply model checking for each different cost bound on a different computing node in parallel. Thus, for cost bounds, we show results obtained by applying model checking with the exact cost bound k instead of all the cost bounds from 1 to k. Table. III shows the experimental results, where each result shows the average execution times over five trials. We distinguish the executions that result in finding a path with a gray-colored cell. We can observe that the experimental results almost conform to the complexity analysis; i.e., the computing cost w.r.t. the length of the formula and cost bound k increases in an exponential manner. We can observe that the cost w.r.t. the bound k decreases after finding a counter-example; this reflects the property of "phase transition", which is a feature of SAT problem [11], [12].

VI. RELATED WORK

As mentioned, this work is inspired by [1], [2] to provide a formal framework for milk-run transport logistics which reduces its complexity. But our work differs from those in its purpose; i.e., the purpose of our work is to develop a formal framework for automated route planning, while that of [1], [2] is to develop a framework for a mechanism for truck selection. This makes differences in the technologies used in the frameworks. Our framework uses model checking, while [1], [2] uses bisimulation checking. Also, this work is the first to apply model checking technologies, originally for automated verification, to automated route planning for milk-run transport logistics.

Various shortest-path problems and their algorithms in graph theory, such as "single-source shortest path problems" (and Dijkstra's algorithm), "all-pairs shortest path problems", "travelling salesman problems", "widest path problems", etc, share a common link with our framework in finding optimal paths provided weighted graphs. Note that these graph algorithms are designed to solve their own specific and fixed problems; e.g., Dijkstra's algorithm is designed specifically for a "single-source shortest path problem". But this work proposes a general framework; i.e., it provides a way to flexibly specify various problems as "delivery requirements" by LTL, and a mechanism to solve these problems by automatically finding shortest paths for the given problems in a unified way.

VII. CONCLUSION AND FUTURE WORK

Conclusion: This paper has developed an automated route planning framework for milk-run transport logistics, which given a route map and delivery requirements finds optimal truck routes that satisfy requirements with the lowest cost. The framework is realized by applying model-checking techniques. It uses LTL as a specification language for specifying delivery requirements in milk-run logistics, which are complex w.r.t. the order of locations which trucks should visit. We have discussed the framework formally, including the notion of "optimal truck routes", by applying bounded semantics. Further based on the formal foundation we implemented the framework by applying the NuSMV model checker. As the milk-run logistics is used in various ways in various industries, we have shown the computational complexity and experimental results as the feasibility of the implementation.

Future work: This paper focuses on developing the framework and on demonstrating its feasibility, by implementing the system for the framework in a simple manner as an early trial; i.e., we leave the aspect of efficiency and correctness of the system design and implementation to our future papers. Another important research direction is to develop a Domain Specific Language (DSL) for specifying delivery requirements in milk-run logistics. In this paper, LTL is used for that purpose in order to show the applicability of model checking to a route planning problem in milk-run logistics. However more suitable languages can be designed to more effectively express delivery requirements in milk-run logistics in practice.

ACKNOWLEDGEMENT

The authors would like to thank Dr. Makoto Takeyama in the National Institute of Advanced Industrial Science and Technology (AIST) for his time and insightful suggestions throughout this research.

REFERENCES

[1] I. Satoh, "A specification framework for earth-friendly logistics," in Proc. of FORTE, vol. E91-A, no. 11. Oxford University Press, 2008, pp. 251–266.

[2] I.Satoh, "A formal approach for milk-run transport logistics," IEICE Trans. Fundam. Electron. Commun. Comput. Sci., vol. E91-A, no. 11, pp. 3261–3268, 2008.

[3] R. Milner, Communication and concurrency, ser. PHI Series in computer science. Prentice Hall, 1989.

[4] A. Biere, A. Cimatti, E. M. Clarke, and Y. Zhu, "Symbolic model checking without BDDs," in Proc. of TACAS. Springer-Verlag, 1999, pp. 193–207.

[5] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, and Y. Zhu, "Bounded model checking," Advances in Computers, vol. 58, pp. 118–149, 2003.

[6] E. W. Dijkstra, "A note on two problems in connexion with graphs," Numerische Mathematik, vol. 1, pp. 269–271, 1959.

[7] E. M. Clarke, D. Kroening, J. Ouaknine, and O. Strichman, "Completeness and complexity of bounded model checking," in Proc. of VMCAI, 2004, pp. 85–96.

[8] E. Clarke, D. Kroening, J. Ouaknine, and O. Strichman, "Computational challenges in bounded model checking," STTT, vol. 7, no. 2, pp. 174–183, 2005.

[9] "Nusmv: a new symbolic model checker," available from http://nusmv.fbk.eu/.

[10] A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella, "NuSMV Version 2: An OpenSource Tool for Symbolic Model Checking," in Proc. of CAV, ser. LNCS, vol. 2404. Springer, 2002.

[11] P. Cheeseman, B. Kanefsky, and W. M. Taylor, "Where the really hard problems are," in Proc. of IJCAI. Morgan Kaufmann Publishers Inc., 1991, pp. 331–337.

[12] D. Mitchell, B. Selman, and H. Levesque, "Hard and easy distributions of SAT problems," in Proc. of AAAI. AAAI Press, 1992, pp. 459–465.