Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2002), *16*, 31–38. Printed in the USA.

DOI: 10.1017.S0890060401020042

Vehicle route-sequence planning using temporal logic

KIAM TIAN SEOW[1] AND MICHEL PASQUIER[2]

[1]Centre for Decision Support, DSO National Laboratories, Republic of Singapore 118230
[2]Intelligent Systems Laboratory, School of Computer Engineering, Nanyang Technological University, Republic of Singapore 639798

(RECEIVED June 4, 2000; REVISED July 18, 2001; ACCEPTED July 26, 2001)

Abstract

This paper proposes a new logical framework for vehicle route-sequence planning of passenger travel requests. Each request is a *fetch-and-send* service task associated with two request-locations, namely, a source and a destination. The proposed framework is developed using propositional linear time temporal logic of Manna and Pnueli. The novelty lies in the use of the formal language for both the specification and theorem-proving analysis of precedence constraints among the location visits that are inherent in route sequences. In the framework, legal route sequences—each of which visits every request location once and only once in the precedence order of fetch-and-send associated with every such request—is formalized and justified, forming a basis upon which the link between a basic precedence constraint and the corresponding canonical forbidden-state formula is formally established. Over a given base route plan, a simple procedure to generate a feasible subplan based on a specification of the forbidden-state canonical form is also given. An example demonstrates how temporal logic analysis and the proposed procedure can be applied to select a final (feasible) subplan based on additional precedence constraints.

Keywords: Constraint Specification; Fetch-and-Send Problem; Temporal Logic; Vehicle Route Plan

1. INTRODUCTION

In this paper, we study the *pickup-and-delivery* problem (Savelsbergh & Sol, 1995) in the travel-service domain of passenger land transportation. Past research has approached the *pickup-and-delivery* problem (better called the passenger *fetch-and-send* problem in our travel-service terminology) and its variants using mainly graph-theoretic and operations research techniques, including heuristics, optimization, and branch-and-bound methods (see Ruland, 1995), and the references contained therein). One common formulation of the problem, perhaps the most general of which is proposed by Savelsbergh and Sol (1995), is a mathematical program of various constraints to be solved simultaneously with respect to optimizing (i.e., either maximizing or minimizing) a certain objective function. The solution is a set of routes attending to all the travel requests, with each route servicing a disjoint subset of these travel requests assigned to a vehicle. However, though well-formulated, such an approach does not appear flexible enough to admit travel-service directives that translate into additional precedence constraints to be satisfied. These directives are usually not known *a priori* because they are dependent on various *online* situations; for instance, due to the peculiarities of a particular request set assigned to a vehicle, the higher service priority of one travel request over another may be specifically demanded as a directive. In attempting to uncover new insights and provide a more flexible planning basis, we abstract the *fetch-and-send* problem as a problem of route-sequence planning. Route-sequence planning is concerned with generating a (high-level) *logical* route plan for a given travel-request set. Such a plan is a set of route sequences each showing a proper sequential order of request locations to visit. The route plan must be *feasible* in the sense that it must satisfy not only inherent constraints such as the precedence of fetching the traveller(s) at the designated pickup or *source* location before sending them to the designated delivery or *destination* location, but also on additional precedence requirements which are travel-service directed.

To formally specify and analyze precedence constraints for route-plan generation, this paper proposes a planning

Reprint requests to: Kiam Tian Seow, Centre for Decision Support, DSO National Laboratories, 20 Science Park Drive, Republic of Singapore 118230. E-mail: KiamTian@singnet.com.sg

framework using the (propositional linear time) temporal logic of Manna and Pnueli (1983; Ostroff, 1989). Temporal logic is a branch of modal logic (Rescher & Urquhart, 1971). In addition to the usual connectives such as AND $\wedge$, OR $\vee$, and NOT $\neg$, modal logic deals with two propositional operators $\Box$ and $\Diamond$, which are interpreted as "always" and "eventually," respectively, in temporal logic (Manna & Pnueli, 1983; Ostroff, 1989). Temporal logic is chosen for our work because it is perhaps the only formal language that provides an intuitively appealing and useful set of operators, including Precede $\mathcal{P}$, Until $\mathcal{U}$, Always $\Box$ and Eventually $\Diamond$, logically unified under a well-established paradigm. These operators enable us to specify the constraints for route sequences in a natural language fashion. The semantics of linear time temporal logic is, in essence, a *directed* graph of sequences over which temporal formulas are reasoned. By formulating and restricting this graph as a model that encompasses the route sequences, we empower route-sequence planning with an expressive yet precise (temporal) language for describing the various precedence constraints among the location visits as temporal formulas. Temporal logic also provides a proof system to mechanize the analysis of complex precedence relationships which are invariably involved.

In this paper, the focus is on route-sequence planning for the *fetch-and-send* problem via a single vehicle. This is unlike the *general fetch-and-send* problem addressed in Savelsbergh and Sol (1995), in which a fleet of service vehicles is considered. But importantly, it is envisaged that the planning framework proposed can subsequently be generalized or strategically deployed for route-sequence planning via a fleet of vehicles. In strategic deployment, one could consider a (partially decentralized) route-planning system, in which a fleet manager first applies some heuristics to decompose a set of pending travel requests into disjoint subsets, before assigning each subset to a fleet vehicle agent. The vehicle agent then proceeds to generate the individual route plan as travel-service directed. The kind of heuristics the fleet manager requires may be quite complex though, and is in itself an important topic for further research that is beyond the scope of this paper.

The ultimate goal of this research is to develop an intelligent route-planning system for passenger travel service. Towards the goal, the contribution of this paper is to motivate and develop an analytical framework using temporal logic for characterizing, specifying, and analyzing vehicle route plans. The purpose is to develop a theoretical basis on which the route-planning system for passenger travel service can be built. The framework is, in our opinion, also equally applicable in many other domains concerned with the distribution of goods and services as surveyed in Psaraftis (1995), in which the problem of pickup and delivery is inherent.

In the rest of the paper, Section 2 presents a planning scenario that defines our research scope. Section 3 presents the formal syntax and semantics of temporal logic. To provide a logical foundation for route-sequence planning, Section 4 provides a temporal characterization of legal route sequences— each of which visits every request location once and only once in the precedence order of fetch-and-send associated with every such travel request. Based on the characterization of legal sequences, Section 5 establishes an important link between a basic precedence constraint and the corresponding canonical forbidden-state formula. Over a given base route plan—a subset of legal route sequences which can be serviced by a vehicle without its (maximum) capacity ever being exceeded—a simple procedure to generate a feasible subplan based on a specification of the forbidden-state canonical form is also given. In Section 6, an example demonstrates how temporal logic analysis and the proposed procedure can be applied to select a final subplan based on additional precedence specifications. Finally, Section 7 summarizes and concludes the paper.

2. PLANNING SCENARIO

In our planning formulation, a set of travel requests initiated is to be serviced by an assigned vehicle. In general, each request is assumed to consist of a pickup or *source* location and a delivery or *destination* location. The service is done by the vehicle following a certain sequence of visits to these request locations corresponding to the given set of requests; when all locations are visited, the service is said to have been completed. Implicit in these visits are the *pickup* task at the source location and the *delivery* task at the destination.

2.1. Request set

Given a set of requests $\mathcal{R}$ associated with location set $\mathcal{T}$ for a vehicle, we assume that there are two subsets of travel requests $\mathcal{SD}$, and $\mathcal{D}$, namely, a request $R_i = (s_i, d_i) \in \mathcal{SD}$ that needs to be fetched from a location represented by $s_i \in \mathcal{T}$ and sent to a location represented by $d_i \in \mathcal{T}$; and a request $D_k = (t_k) \in \mathcal{D}$ that only needs to be sent to a location represented by $t_k \in \mathcal{T}$. We can think of a request in $\mathcal{D}$ as having its source already visited and the corresponding travellers on board the vehicle initially. We also include a set of auxiliary requests $A_s = (a_s) \in \mathcal{A}$, with locations $a_s \in \mathcal{T}$ that need to be visited. Formally, the set relations are as follows: $\mathcal{R} = \mathcal{SD} \cup \mathcal{D} \cup \mathcal{A}$ and $\mathcal{T} = \{s_i | (s_i, -) \in \mathcal{SD}\} \cup \{d_i | (-, d_i) \in \mathcal{SD}\} \cup \{t_k | (t_k) \in \mathcal{D}\} \cup \{a_s | (a_s) \in \mathcal{A}\}$. The symbols s_i, d_i, t_k, and a_s are location (visit-)status indicators $l \in \mathcal{T}$ formally defined as follows.

Definition 1 (Location Variable). It is a Boolean or logic variable $l \in \mathcal{T}$ that indicates the status of whether the request location it represents has been visited or not as follows:

$$l = \begin{cases} 1 & \text{if location } l \text{ is visited,} \\ 0 & \text{otherwise.} \end{cases} \qquad \blacksquare$$

No two elements in $\mathcal{R}$ correspond to the same request. Each location visit is assumed to be associated with the comple-

https://doi.org/10.1017/S0890060401020042 Published online by Cambridge University Press

tion of a different task and therefore, no two location variables from request set $\mathcal{R}$ are symbolically identical. Then, it should be clear that $|\mathcal{T}| = 2|\mathcal{SD}| + |\mathcal{D}| + |\mathcal{A}|$.

To elaborate on the tasks at these locations, a visit to a source location of some request is associated with the task of *fetching* the traveller(s) concerned (i.e., he/they has/have boarded the vehicle); a visit to a destination location is associated with the task of *sending* the traveller(s) concerned (i.e., he/they has/have alighted from the vehicle). Finally, a visit to an auxiliary location of $\mathcal{A}$ is associated with a certain service-support task such as *refuelling* at a kiosk or return to some designated depot for another travel assignment. In a typical route plan, it is possible that $\mathcal{A} = \varnothing$ and/or $\mathcal{D} = \varnothing$.

2.2. Directed (state) graph of vehicle route plan

A vehicle route plan to service the request set $\mathcal{R}$ constituting the location set $\mathcal{T}$ is contained in a directed graph **G**:

$$\mathbf{G} = (\mathcal{Q}, \mathcal{Z}) \tag{1}$$

where:

1. $\mathcal{Q}$ denotes a set of nodes,
2. $\mathcal{Z} = (\mathcal{Q}, \mathcal{T}, \rho(\mathcal{T}, \mathcal{Q}))$ is a set of transitions for which
   - ρ: $\mathcal{T} \times \mathcal{Q} \mapsto \mathcal{Q}$ denotes the partial node transition function, associating the transition with $l \in \mathcal{T}$ as the (next) location to visit. The definition of ρ can be extended to $\mathcal{T}^*$, the set of all possible finite strings of elements in $\mathcal{T} \cup \{\varepsilon\}$, with ε denoting a null string, as follows:

$$\rho(\varepsilon, v) = v,$$

$$(\forall t \in \mathcal{T})(\forall w \in \mathcal{T}^*), \rho(wt, v) = \rho(t, \rho(w, v)).$$

   - for $v \in \mathcal{Q}$, v: $\mathcal{T} \mapsto \{0,1\}$. ($v(l)$ is read as: 'value of variable l in node v').

REMARK 1. The null string $\varepsilon \notin \mathcal{T}$ is a generic symbol to denote the vehicle's current location at node $v \in \mathcal{Q}$. ■

The following definitions formalize some intuitively clear concepts.

DEFINITION 2 (ROUTE STATE). Every route state is a discrete node $v \in \mathcal{Q}$ that assigns to every location variable $l \in \mathcal{T}$ a value $v(l)$ of true (logic 1) or false (logic 0) to indicate whether the location $l \in \mathcal{T}$ represents is visited or not, respectively. No two distinct states assign the same set of values to the variables of $\mathcal{T}$. ■

DEFINITION 3 (ROUTE-SEQUENCE). σ is a finite sequence of distinct route states that is a path in **G** of the following general form:

$$v_0 - v_1 - v_2 - \ldots - v_n$$

and for all $0 < i \leq n$, there is a $l \in \mathcal{T}$ such that $v_i = \rho(l, v_{i-1})$. ■

3. TEMPORAL LOGIC BACKGROUND

Propositional linear time temporal logic (Ostroff, 1989) is a language of propositional (or Boolean) logic augmented with temporal operators to facilitate reasoning over the space of sequences of states. The logical "AND," "OR," "Negation," "Implication," and "Equivalence" connectives used in propositional logic (Ross & Wright, 1988) are denoted by the symbols $\wedge$, $\vee$, $\neg$, $\rightarrow$, and $=$, respectively. The sum ($\sum$) and product ($\prod$) symbols denote a string of logical "OR" and "AND" operations, respectively. Temporal operators include Always $\Box$, Eventually $\Diamond$, Until $\mathcal{U}$, and Precede $\mathcal{P}$. A logic formula W expressed in propositional temporal logic is called a (propositional) temporal formula; a temporal formula that does not contain any temporal operator is also called a Boolean formula.

3.1. Syntax

A temporal formula is inductively defined (over the Boolean variables in set $\mathcal{T}$) via the following syntax rules for Boolean formulae and temporal formulas.

- For Boolean formula:
  1. Any Boolean variable $l \in \mathcal{T}$ is a Boolean formula.
  2. If F is a Boolean formula, so is $\neg F$.
  3. If F_1 and F_2 are Boolean formulas, so are $F_1 \vee F_2$ and $F_1 \wedge F_2$.
- For temporal formula:
  1. Any Boolean formula is a temporal formula.
  2. If W is a temporal formula, so are $\neg W$, $\Box W$, and $\Diamond W$.
  3. If W_1 and W_2 are temporal formulas, so are $W_1 \vee W_2$, $W_1 \wedge W_2$, $W_1 \mathcal{U} W_2$, and $W_1 \mathcal{P} W_2$.

Logically, for two temporal formulae, W_1 and W_2, $W_1 \rightarrow W_2$ is the same as $\neg W_1 \vee W_2$; $W_1 = W_2$ is the same as $(W_1 \rightarrow W_2) \wedge (W_2 \rightarrow W_1)$.

3.2. Semantics

In the context of route-sequence planning, the truth of a temporal formula is interpreted (or evaluated) over a *finite* route-sequence $\sigma = v_0 - v_1 - v_2 - \ldots - v_n \in \mathbf{G}$, with node $v_n \in \mathcal{Q}$ defined as its *terminal* node. The structure **G** can therefore be regarded as a restricted class of possible semantics of temporal logic (Ostroff, 1989).

The suffix of such a sequence σ refers to its subsequence that begins from a given node.

https://doi.org/10.1017/S0890060401020042 Published online by Cambridge University Press

DEFINITION 4 (SUFFIX). For $0 \le k \le n$, the k-truncated suffix of a temporal sequence $\sigma = v_0 - v_i - v_2 - \ldots - v_n$ (written: $\sigma^{(k)}$) is the sequence $v_k - v_{k+1} - v_{k+2} - \ldots - v_n$. ■

Now, for all sequences $\sigma \in \mathbf{G}$, with terminal node $v_n \in \mathcal{Q}$, $n \ge 0$, we say that $\sigma^{(i)}$ satisfies W (written: $\models^{\sigma^{(i)}} W$) where the satisfaction relation is defined inductively on formulas as follows:

$\models^{\sigma^{(i)}} W_1 \vee W_2$ iff $\models^{\sigma^{(i)}} W_1 \vee \models^{\sigma^{(i)}} W_2$.

$\models^{\sigma^{(i)}} W_1 \wedge W_2$ iff $\models^{\sigma^{(i)}} W_1 \wedge \models^{\sigma^{(i)}} W_2$.

$\models^{\sigma^{(i)}} \neg W$ iff $\neg\models^{\sigma^{(i)}} W$.

$\models^{\sigma^{(i)}} \Box W$ iff for all $k \ge i$, $\models^{\sigma^{(k)}} W$.

$\models^{\sigma^{(i)}} \Diamond W$ iff there exists a $k \ge i$ such that $\models^{\sigma^{(k)}} W$.

$\models^{\sigma^{(i)}} W_1 \mathcal{U} W_2$ iff there is a $k \ge i$ such that $\models^{\sigma^{(k)}} W_2$ and for all $m, i \le m < k$, $\models^{\sigma^{(m)}} W_1$.

$\models^{\sigma} W_1 \mathcal{P} W_2$ iff $\neg\models^{\sigma} \neg W_1 \mathcal{U} W_2$.

REMARK 2. The following two temporal logic theorems (Ostroff, 1989), and a derived rule (Seow et al., 1999), expressed in terms of arbitrary temporal formulas P, P_1, and P_2, are applied in the formal analysis made in this paper.

T5: $\vdash \Box P = \neg\Diamond\neg P$ T7: $\vdash \Box(P_1 \wedge P_2) = (\Box P_1) \wedge (\Box P_2)$

D40
$$\frac{\vdash \Box(P_1 \rightarrow \Box P_1) \qquad \vdash \Diamond P_1}{\vdash P_2 \mathcal{U} P_1 = \Box(P_1 \vee P_2)}$$

The labels T5, T7, and D40 are as given in the source (Ostroff, 1989; Seow et al., 1999). ■

4. THE LEGAL ROUTE SEQUENCES

Before presenting a characterization of legal route sequences, some important terminologies are first defined.

DEFINITION 5 (REQUEST-LOCATION FORMULA L). L is an arbitrary Boolean formula that contains at least one location variable $l \in \mathcal{T}$ and does not have any "$\neg$" (logical "NOT") connective. ■

Therefore, a (request-)location formula L is a simple logic expression, in positive form, of $\wedge$'s (conjunctions) and $\vee$'s (disjunctions) only, relating the location variables. Clearly, Definition 5 is a generalization of Definition 1.

Following is a route state formula characterizing a route node $v \in \mathcal{V}$ of Definition 2 in terms of variables $l \in \mathcal{T}$.

DEFINITION 6 (ROUTE STATE FORMULA F^r). For location set $\mathcal{T} = \{l_1, l_2, \cdots, l_i, \cdots, l_n\}$, $n = |\mathcal{T}|$,

$$F^r = l_a \wedge l_b \ldots \wedge l_h \wedge \neg l_p \wedge \neg l_q \ldots \wedge \neg l_z$$

with

$$\{a, b, \ldots, h\} \cap \{p, q, \ldots, z\} = \{\} \text{ and } \{a, b, \ldots, h\} \cup \{p, q, \ldots, z\} = \{1, 2, \ldots, n\}.$$ ■

It is clear that a route state formula F^r characterizes one and only one particular node $v \in \mathcal{Q}$ of Definition 2, and is a simple Boolean formula involving only conjunctions $\wedge$'s of all the location variables $l \in \mathcal{T}$ once, and only once.

4.1. Route-sequence properties

Let L be an arbitrary location formula.

THE PERSISTENCE AXIOM (RSP 1) $\vdash \Box(L \rightarrow \Box L)$. ■

THE LIVENESS AXIOM (RSP 2) $\vdash \Diamond L$. ■

THE PRECEDENCE AXIOM (RSP 3) $\vdash s_i \mathcal{P} d_i$ for an arbitrary $R_i \in \mathcal{SD}$. ■

RSP 1–3 are necessity properties of legal route sequences. RSP 1 is necessary for efficiency of route-sequence planning since it specifies that every location, once visited, should not at all be unvisited or revisited with respect to servicing the associated request throughout the rest of the route. RSP 2 specifies that every location should eventually be visited. RSP 3 specifies the proper order of servicing each travel request, namely, for each request $R_i \in \mathcal{SD}$, its source location $s_i \in \mathcal{T}$ must be visited before its destination location $d_i \in \mathcal{T}$ is.

These three properties, RSP 1–3, characterize the generic constraints on route sequences. In temporal logic terminology, these properties are *axioms* that characterize the (domain) sequences of interest. With these fundamental properties, we may formally define a legal route sequence as follows:

DEFINITION 7 (LEGAL ROUTE-SEQUENCE σ). σ is a path $v_0 - v_1 - v_2 - \ldots - v_n$, $v_k \in \mathcal{Q}$, that obeys all the following conditions:

1. $v_i \ne v_j$ iff $i \ne j$,
2. $n = |\mathcal{T}|$,
3. for all $v_i \in \mathcal{Q}$, $0 \le i < n$, and $l \in \mathcal{T}$, $\rho(l, v_i) = v_{i+1}$ iff $v_{i+1}(l) = 1$ and $v_i(l) = 0$,
4. $v_0 \in \mathcal{Q}$ is the initial node where all locations are not visited (i.e., all location variables $l \in \mathcal{T}$ are false) and $v_n \in \mathcal{Q}$ is the final node where all the locations are visited (i.e., all location variables $l \in \mathcal{T}$ become true),
5. RSP 1–3. ■

REMARK 3. Some comments on legal route sequences and their implications are as follows:

1. Conditions (1) to (4) of Definition 7, RSP 1, and RSP 2 together mean that every location must be visited

https://doi.org/10.1017/S0890060401020042 Published online by Cambridge University Press

once and only once; along a legal sequence, there is one and only one location transition $l \in \mathcal{T}$ that leads node v_i, $0 \le i < n$, to node v_{i+1}, $v_{i+1} = \rho(l, v_i)$, such that only the logic value of variable $l \in \mathcal{T}$ is changed in node v_{i+1}.

2. RSP 2 and RSP 3 together ensure completion of servicing each request.
3. In route-sequence planning for a given set of request locations, any route sequence that is not legal need not be considered further because it is either not plan efficient or it will not result in proper completion of servicing all the request tasks. ■

5. FEASIBLE ROUTE-SEQUENCE PLANNING

For a nonempty request set $\mathcal{R}$ as defined in Section 2.1, this section presents the route-plan generation based on a forbidden-state formula. The notion of a route plan is first defined.

Definition 8 (Route Plan). A route plan for a request set $\mathcal{R}$ is a set of only legal route sequences. ■

Remark 4 (Base Route Plan). For a vehicle, a base route plan is a subset of legal route sequences, any of which the vehicle can logically traverse to service these travel requests in set $\mathcal{R}$ without ever exceeding its (maximum) capacity. The generation of a base route plan is beyond the scope of this paper, but is presented elsewhere (Seow et al., 1999). ■

Very often, the final (feasible) route plan to be derived depends not only on the base route plan, but also on certain travel-service directives that translate into additional precedence constraints. The nature of these directives is such that their corresponding constraints can be used in decision making to narrow down the choice to a few *desired* route sequences from among the possible sequences suggested by the base plan; for instance, due to the peculiarities of a given request set, the higher service priority of one travel request over another may be specifically demanded as a directive. In the following, we provide a simple procedure to derive a final subplan that additionally satisfies a class of precedence constraints which are, in essence, forbidden-state formulas.

The canonical form of a forbidden-state specification is given in Definition 9.

Definition 9 (Canonical Forbidden-State Formula). The canonical form of a forbidden-state specification is given by $\Box\neg(\sum_{all\ i} S_i)$, where S_i is a Boolean formula. ($\sum_{all\ i} S_i$ is said to assert the forbidden state-properties.)

The following result establishes the link between a basic precedence constraint of the form $\neg L_2 \mathcal{U} L_1$, where L_1, L_2 are location formulas and the corresponding canonical forbidden-state formula.

Theorem 1. *For arbitrary location formulae L_1, L_2 and legal route-sequence σ,*

$$\models^{\sigma} \neg L_2\, \mathcal{U} L_1 = \Box\neg[(\neg L_1) \wedge L_2].$$

Boolean formula $(\neg L_1) \wedge L_2$ is the forbidden state-property.

Proof:

1. $\vdash \Box(L_1 \rightarrow \Box L_1)$ (By RSP 1)
2. $\vdash \Diamond L_1$ (By RSP 2)
3. $\vdash \neg L_2 \mathcal{U} L_1 = \Box(L_1 \vee \neg L_2)$ (by 1, 2 and D40)
4. $\vdash \neg L_2 \mathcal{U} L_1 = \Box\neg[(\neg L_1) \wedge L_2]$ (by 3 and PR[1]).

By Definition 7, a legal route-sequence σ satisfies RSP 1 and 2 for $L = L_1$, hence the result. ■

A formal definition of a forbidden-state specification follows.

Definition 10 (Forbidden-State Formula). Let C_F denote an arbitrary canonical forbidden-state formula of Definition 9. Then a formula F is a forbidden-state specification (over arbitrary legal route sequences σ) iff there exists a C_F such that $\models^{\sigma} F = C_F$. ■

Corollary 1. *A conjunction $\prod_{all\ i} F_i$ is a forbidden-state formula of Definition 10 if each component formula F_i is of the basic form $\neg L_2\, \mathcal{U} L_1$, where L_1, L_2 are location formulas.*

Proof: By T7 and PR, the class of canonical forbidden-state formulas of Definition 9 is closed under conjunctions. By Theorem 1, each (conjunctive) component formula F_i is a forbidden-state formula of Definition 10. Hence the corollary. ■

The procedure for subplan generation that follows is motivated by the fact that many useful precedence constraints are conjunctions of constraints of the basic form, and thus are forbidden-state formulas, as established by Corollary 1 above.

Using the base route plan $\mathbf{G}_{base} = (\mathcal{V}_{base}, \mathcal{E}_{base})$, the final subplan $\mathbf{G}_{sel} = (\mathcal{V}_{sel}, \mathcal{E}_{sel})$, based on a (nontrivial) canonical forbidden-state specification of Definition 9, is defined by

1. $\mathcal{V}_{sel} = \mathcal{V}_{base} - \mathcal{V}_{pf}$,

where

$\mathcal{V}_{pf} \subseteq \mathcal{V}_{\text{base}}$ is a set of nodes forbidden due to $\Box\neg(\sum_{all\ i} S_i)$.

$$\mathcal{V}_{pf} = \left\{ v \,|\, v \in \mathcal{V}_{base} \text{ and } v\left(\sum_{all\ i} S_i\right) = 1 \right\}.$$

[1]"PR" (Ostroff, 1989) stands for Propositional Reasoning and any logic rule or theorem quoted under this generic label PR includes those found in Ross & Wright (1988).

https://doi.org/10.1017/S0890060401020042 Published online by Cambridge University Press

2. $\mathcal{E}_{sel} = (\mathcal{V}_{sel}, \mathcal{T}, \delta_{sel}(\mathcal{T}, \mathcal{V}_{sel}))$ such that the (partial) transition function is defined by

$$\delta_{sel}: \mathcal{T} \times \mathcal{V}_{sel} \mapsto \mathcal{V}_{sel}$$

according to

$$\delta_{sel}(l,v) = \begin{cases} \delta_{base}(l,v) & \text{if } \delta_{base}(l,v) \in \mathcal{V}_{sel}, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

To generate $\mathbf{G}_{\text{sel}}$, compute the following:

1. $\mathcal{V}_{sel} = \mathcal{V}_{base} - \mathcal{V}_{pf}$.
2. $\mathcal{E}_{pf} = \{(v, l, v') | v \in \mathcal{V}_{pf} \text{ and } (v, l, v') \in \mathcal{E}_{base}\} \cup \{(v', l, v) | v \in \mathcal{V}_{pf} \text{ and } (v', l, v) \in \mathcal{E}_{base}\}$ ($\mathcal{E}_{pf}$ refers to the set of forbidden transitions).
3. $\mathcal{E}_{sel} = \mathcal{E}_{base} - \mathcal{E}_{pf}$.

In an algorithmic implementation of the above procedure, every node in the given set $\mathcal{V}_{base}$ is tested, possibly against every forbidden state property S_i of the canonical forbidden-state formula of Definition 9. In testing against each S_i, all the $|\mathcal{T}|$ location variable values in a node $v \in \mathcal{V}_{base}$ are also possibly checked. A node $v \in \mathcal{V}_{base}$ that "satisfies" any of the S_i's (i.e., $v \in \mathcal{V}_{pf}$) is deleted and this can be done simultaneously with deleting all those transitions in set $\mathcal{E}_{base}$ that enter or leave the node (i.e., those transitions in $\mathcal{E}_{pf}$ due to node $v \in \mathcal{V}_{pf}$). Hence, the worst case complexity of the procedure is $\mathbf{O}(|\mathcal{V}_{base}| \times |\mathcal{T}| \times m)$, where m is the number of S_i's of the canonical forbidden-state formula.

6. AN EXAMPLE

The following example demonstrates how, given a base route plan as defined in Remark 4, further precedence constraints, as travel-service directed, may be imposed to produce a final (feasible) subplan. The constituents of $\mathcal{R}$ to be serviced by a vehicle are given as follows:

$$\mathcal{SD} = \{(s_1, d_1), (s_2, d_2)\}, \qquad \mathcal{D} = \{(t_1)\}, \qquad \mathcal{A} = \{(a)\}$$

Thus, $\mathcal{T} = \{a, s_1, s_2, d_1, d_2, t_1\}$. The base route plan for $\mathcal{R}$ is represented as graph $\mathbf{G}_{base}$ in Figure 1.

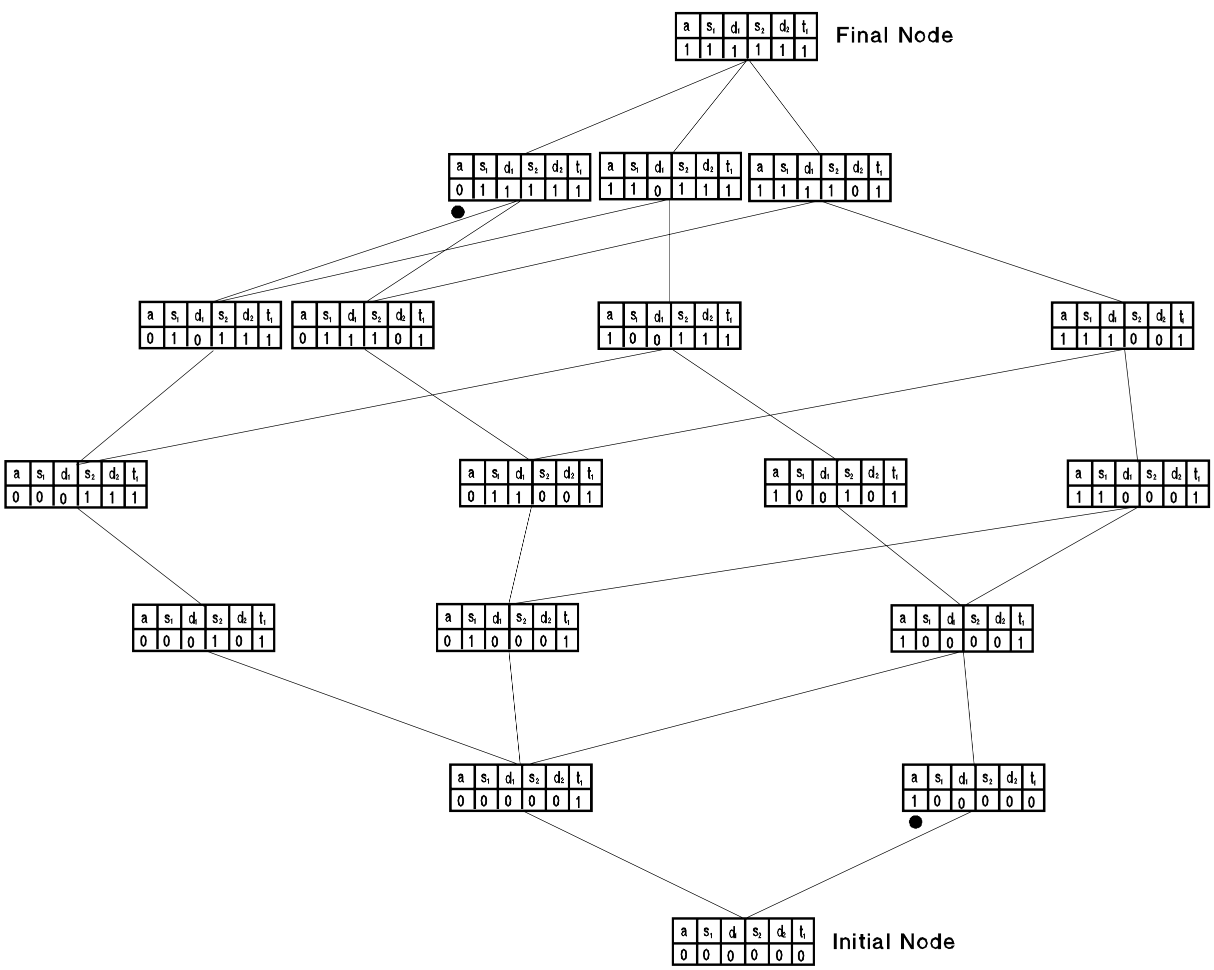

Fig. 1. $\mathbf{G}_{base}$: Graph of base route plan with nodes in $\mathcal{V}_{pf}$ marked with •.

https://doi.org/10.1017/S0890060401020042 Published online by Cambridge University Press

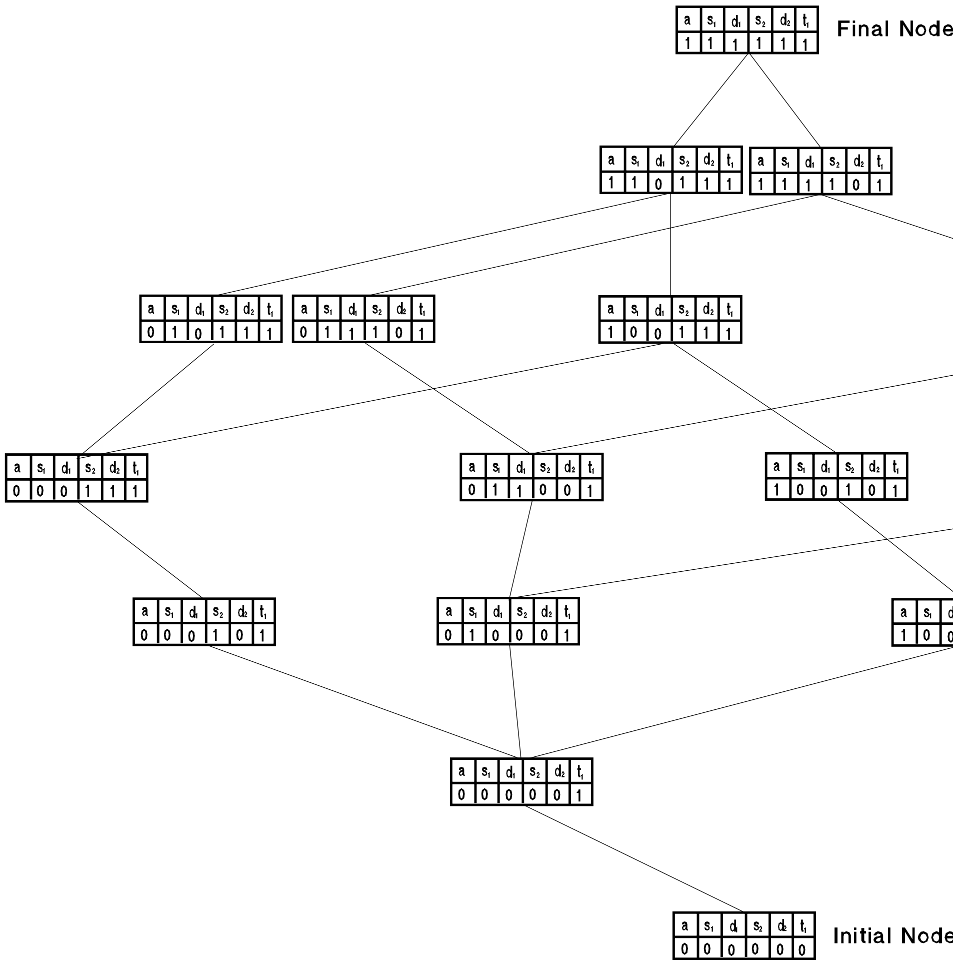

Fig. 2. $\mathbf{G}_{sel}$: Graph of final route plan.

Suppose $(a) \in \mathcal{A}$ denotes a request to visit (the location of) a petrol kiosk (for refuelling). Then, suppose we have the following travel-service directives:

1. The travellers of request $(t_1) \in \mathcal{D}$ do not want to stop by the kiosk; this directive translates into the following specification:
   - $[\neg a\, \mathcal{U} t_1]$
     The temporal formula is paraphrased as "do not visit location $a \in \mathcal{T}$ until request $(t_1) \in \mathcal{D}$ has been serviced (i.e., the travellers on board the vehicle initially have reached the destination (and alighted))."
2. The vehicle must be refuelled after servicing at most two requests in $\mathcal{SD} \cup \mathcal{D}$; this translates into the following specification:
   - $[\neg(d_1 \wedge d_2 \wedge t_1)\mathcal{U} a]$
     The temporal formula is paraphrased as "not all requests in $\mathcal{SD} \cup \mathcal{D}$ will be serviced until request $(a) \in \mathcal{A}$ has been serviced."

Anding the two specifications over legal route sequences, we have $[\neg a\, \mathcal{U} t_1] \wedge [\neg(d_1 \wedge d_2 \wedge t_1)\, \mathcal{U} a]$, which by Corollary 1, is a forbidden-state formula.

1. Temporal Logic Analysis of Specifications

$$[\neg a\, \mathcal{U} t_1] \wedge [\neg(d_1 \wedge d_2 \wedge t_1)\, \mathcal{U} a]$$

$$= \Box\neg[\neg t_1 \wedge a] \wedge \Box\neg[\neg a \wedge (d_1 \wedge d_2 \wedge t_1)] \quad \text{(By Theorem 1)}$$

$$= \Box[\neg\{\neg t_1 \wedge a\} \wedge \neg\{\neg a \wedge d_1 \wedge d_2 \wedge t_1\}] \text{ (by T7)}$$

$$= \Box\neg[\{\neg t_1 \wedge a\} \vee \{\neg a \wedge d_1 \wedge d_2 \wedge t_1\}] \text{ (by PR).}$$

2. Specify $\mathcal{V}_{pf}$. Following, we tabulate the nodes $v \in \mathcal{V}_{pf} \subseteq \mathcal{V}_{base}$ in Table 1; the tabular presentation is such that the row of logic values characterizes a $v \in \mathcal{V}_{pf}$ if the entries marked x are correctly instantiated with either 1s or 0s so that the whole instantiated row characterizes a $v \in \mathcal{V}_{base}$.

https://doi.org/10.1017/S0890060401020042 Published online by Cambridge University Press

Table 1. *The node set* $\mathcal{V}_{pf} \subseteq \mathcal{V}_{base}$

a	s_1	d_1	s_2	d_2	t_1	Boolean Formula
1	x	x	x	x	0	$\neg t_1 \wedge a$
0	x	1	x	1	1	$\neg a \wedge d_1 \wedge d_2 \wedge t_1$

In graph $\mathbf{G}_{base}$ shown in Figure 1, these nodes $v \in \mathcal{V}_{pf}$ are marked with •.

3. Generate $\mathbf{G}_{sel}$. Finally, we have $\mathbf{G}_{sel}$ shown in Figure 2.

REMARK 5. In the example, $\mathbf{G}_{sel}$ does not contain any *stray* node, that is, a node that either cannot be reached from the initial node or does not lead to the final node via the transition function δ_{sel}. In general, the proposed simple procedure for subplan generation does not guarantee the absence of stray nodes. A standard operation to *trim* (i.e., remove all the stray nodes of) $\mathbf{G}_{sel}$ may be required, and it can be easily implemented by an algorithm of linear computational complexity $\mathbf{O}(|\mathcal{V}_{sel}|)$ (Cassandras & Lafortune, 1999). ■

7. SUMMARY AND CONCLUSION

In this paper, vehicle route-sequence planning, a high-level process of generating a route plan for a given set of requests, is introduced and studied for the single-vehicle version of the passenger *fetch-and-send* problem. Using temporal logic, a new logical framework for specifying and analyzing precedence constraints underlying route-sequence planning is proposed. Under the framework, a basis in terms of three fundamental properties, namely, RSP 1–3, that describe the legal route-sequence behavior is introduced and justified. On this basis, the link between a basic precedence constraint and the corresponding canonical forbidden-state formula has been formally established. Over a given base route plan, a simple procedure to generate a final (feasible) subplan based on a specification of the forbidden-state canonical form is given. An example demonstrates how temporal logic analysis and the proposed procedure can be applied to select a final subplan based on additional precedence specifications.

In conclusion, although previous research (see Ruland, 1995, and the references contained therein) has built predominantly upon the solid foundations of operations research, it does not appear to have isolated and dealt with the logical issue of flexible route-sequence planning through the use of travel-service directives, as has been attempted in this paper using temporal logic.

ACKNOWLEDGMENTS

The research work reported in this paper was done under an NTU-INRIA Joint Initiative. The first author acknowledges the support of a research fellowship award (AcRF RG52/97) under the *Strategic Research Programme: Vision and Control* directed by Associate Professor Ming Xie of the School of Mechanical and Production Engineering, Nanyang Technological University (NTU), Singapore. Research done while author was with the Division of System and Engineering Management, School of Mechanical and Production Engineering, Nanyang Technological University, Nanyang Avenue, Republic of Singapore 639798.

REFERENCES

Cassandras, C.G., & Lafortune, S. (1999). Languages and Automata. In *Introduction to Discrete Event Systems*, pp. 61–133. Kluwer Academic Publishers, Boston, MA.

Manna, Z., & Pnueli, A. (1983). A temporal proof system. In *Foundations of Computer Science IV, Distributed Systems: Part 2, Semantics and Logic* (de Bakker, J.W., & Leeuwen, J.V., Eds.), pp. 163–255. Mathematical Centre Tracts 159, Amsterdam. Centre for Math and Computer Sciences.

Ostroff, J.S. (1989). Appendix A: Formal overview of temporal logic and Appendix B: Temporal logic theorems and rules. In *Temporal Logic for Real Time Systems*, pp. 155–171 and 172–183. New York: Research Studies Press Ltd., John Wiley & Sons Inc.

Psaraftis, H.N. (1995). Dynamic vehicle routing: Status and prospects. *Annals of Operations Research 61*, 143–164.

Rescher, N. & Urquhart, A. (1971). *Temporal Logic*. Springer Verlag see p. 32, New York.

Ross, K.A., & Wright, C.R.B. (1988). *Discrete Mathematics*, 2nd Ed. Englewood Cliffs, NJ: Prentice Hall, Inc.

Ruland, K.S. (1995). *Polyhedral solution to the pickup and delivery problem*. Doctor of Science Thesis, Seattle, WA: Washington University.

Savelsbergh, M.W.P., & Sol, M. (1995). The general pickup and delivery problem. *Transportation Science 29(1)*, 17–29.

Seow, K.T., Pasquier, M., & Hong, M.L. (1999). Vehicle route-sequence planning for passenger travel service. Collaboration Project: TECH REP #02-99, Schools of Applied Science/Mechanical and Production Engg., Nanyang Technological University, Singapore.

Kiam-Tian Seow received the degree of B.Eng. (1990) from The National University of Singapore, and the degrees of M.Eng. (1993) and Ph.D. (1998) from Nanyang Technological University (NTU), Singapore, all in Electrical Engineering and Computer Science. Since April 2000, he has been working in DSO National Laboratories, Singapore. Prior to that, he held an NTU research fellowship for two years. In 1997, he was a (short-term) visiting fellow with the Systems Control Group, University of Toronto, Canada. Dr. Seow's current research interests are in supervisory control of discrete-event systems, temporal logic, and multi-agent systems, with emphasis on their applications.

Michel Pasquier received a diploma of Electrical Engineering and a Ph.D. degree in Computer Science in 1985 and 1988, respectively, from the Institut National Polytechnique de Grenoble (INPG), France. From 1989 to 1993, he worked in Tsukuba, Japan, first as a visiting researcher at the ElectroTechnical Laboratory (ETL), then as a research engineer for Sanyo Electric. In 1994, Dr. Pasquier joined the Nanyang Technological University, Singapore, where he has since been teaching courses in Artificial Intelligence and Software Engineering. His research interests include fundamental AI and especially search and planning techniques, with applications to intelligent transportation, robotics and automation.