On the complexity of maximally permissive deadlock avoidance in multi-vehicle traffic systems

Spyros A. Reveliotis and Elzbieta Roszkowska

Abstract— **The establishment of collision-free and live vehicle motion is a prominent problem for many traffic systems. Past work studying this problem in the context of guidepath-based and free-range vehicular systems has implicitly assumed that its resolution through maximally permissive supervision is NP-hard, and therefore, it has typically pursued suboptimal (i.e., more restrictive) solutions. The work presented in this paper offers formal proof to this implicit assumption, closing the apparent gap in the existing literature. In the process, it also derives an alternative proof for the NP-hardness of maximally permissive liveness-enforcing supervision in Linear, Single-Unit Resource Allocation Systems, that is more concise and more lucid than the currently existing proof.**

I. Introduction

The establishment of collision-free and live vehicle motion is a prominent problem for many traffic systems. In this work, we are particularly interested in the manifestation of this problem in the operational context of (a) guidepath-based vehicular systems, like the Automated Guided Vehicle (AGV) and overhead monorail systems that are used in many industrial environments [1], and (b) free-range vehicular systems, where a set of mobile agents moves freely within a confined planar area [4]. Past work has shown that, in both cases, the collision-free and live motion of the underlying traffic system can be established by superimposing a resource allocation structure on it and invoking results from the burgeoning resource allocation system (RAS) theory [7]. More specifically, in the case of guidepath-based vehicular systems, the guidepath network is partitioned into a number of "zones" and it is enforced that any such zone can be traversed by at most one vehicle at any point in time (e.g., [6], [10], [11]). Similarly, in the case of free-range vehicular systems, the entire motion area is partitioned into a number of "cells" by means of a rectangular tesselation, and it is required that every cell contains at most one vehicle at any point in time (e.g.,[9], [8], [3]). Hence, in both cases, the elements of the superimposed partition – i.e., the zones or the cells – are treated as "resources" that must be sequentially acquired and released by the system vehicles in order to execute their designated routes. The vehicle separation resulting from this control scheme ensures the collision-free operation of the entire system, but it also arises the need for further control logic that will ensure that the applied resource allocation is live, i.e., deadlocks will be avoided and every vehicle will eventually advance to its final destination.

As already mentioned, this requirement for live resource allocation in the considered traffic systems has been addressed through the adaptation of a set of results developed for the problem of deadlock avoidance arising in more generic resource allocation systems [7]. The derived resource allocation policies have been based on the implicit assumption that, similar to the more generic cases of sequential resource allocation, enforcing the liveness of the considered traffic systems in a maximally permissive manner is an NP-hard problem, and therefore, these policies tend to sacrifice permissiveness for computational tractability. However, when viewed from a more theoretical standpoint, the computational complexity of maximally permissive deadlock avoidance in the aforementioned traffic systems is an open issue. The topology of the guidepath network structures and the geometry of the tesselations employed in the specification of the resource allocation that takes place in these two traffic environments, imply additional constraints for the structure of the resulting RAS. These constraints are not satisfied by the reductions that have been employed in the past for the establishment of the NP-hardness of maximally permissive deadlock avoidance / liveness-enforcing supervision in various RAS classes, and therefore, the relevant proofs are not directly applicable to the new cases considered herein.

This work addresses the theoretical gap identified in the previous paragraph. The presented results are structured as follows: First, we review the class of Linear, Single-Unit RAS (L-SU-RAS) and the problem of "(RAS) state safety" [7], that is at the core of the maximally permissive deadlock avoidance in any sequential RAS, and we provide an alternative proof for its NP-completeness in the L-SU-RAS context. Subsequently, we use this proof as a "stepping stone" in order to establish the NP-completeness of state safety in free-range vehicular systems. Finally, the NP-completeness of the state safety in guidepath-based vehicular systems is obtained as a straightforward corollary of the second result. Concluding this introductory section, we want also to notice that, beyond its role as a "stepping stone" towards the main results of this paper, the presented proof for the NP-completeness of state safety in L-SU-RAS is more concise than the original proof for the same result that was provided in [5], and more lucid regarding the elements that underlie the increased problem complexity; therefore, it holds its own merit as a technical result.

II. The Linear Single-Unit RAS

The system For the purposes of this work, a *Linear Single-Unit Resource Allocation System (L-SU-RAS)* is formally defined by a 4-tuple $\Phi =< \mathcal{R}, C, \mathcal{P}, D >$, where: (i) $\mathcal{R}$ is the set of the system *resource types*, (ii) $C : \mathcal{R} \rightarrow Z^+$ – the set of strictly positive integers – is the system *capacity* function, characterizing the number of identical units from each resource type available in the system, (iii) $\mathcal{P} = \{\Pi_1, \ldots, \Pi_n\}$ denotes the set of the system *process types*, where each process Π_i, $i = 1, \ldots, n$, consists of $\Xi_{i1}, \Xi_{i2}, \ldots, \Xi_{il_i}$ consecutive *processing stages*, and (iv) $D : \Xi = \{\Xi_{ij} \mid i = 1, \ldots, n; j = 1, \ldots, l_i\} \rightarrow \mathcal{R}$ is the *resource allocation function* associating every processing stage Ξ_{ij} with the resource required for its execution. At any point in time, the system contains a certain number of (possibly zero) instances of each process type that execute one of the corresponding processing stages. A process instance executing a non-terminal stage $\Xi_{ij}, i = 1, \ldots, n; j = 1, \ldots, l_i - 1$, must first be allocated a resource unit of the resource type $D(\Xi_{i,j+1})$ in order to advance to its next stage $\Xi_{i,j+1}$, and only then it will release the currently held resource unit of $D(\Xi_{ij})$. The considered resource allocation protocol further requires that no resource type $R \in \mathcal{R}$ is over-allocated with respect to its capacity $C(R)$ at any point in time. The resulting dynamics of L-SU-RAS can be represented by a *Deterministic Finite State Automaton (DFSA)* [2].

Spyros A. Reveliotis is with the School of Industrial & Systems Engineering, Georgia Institute of Technology, spyros@isye.gatech.edu. During this work, Dr. Reveliotis was partially supported by NSF grants CMMI-0619978 and CMMI-0928231.

Elzbieta Roszkowska is with the Institute of Computer Engineering, Control and Robotics, Wroclaw University of Technology, ekr@pwr.wroc.pl

Definition 1: The DFSA $G(\Phi) = (S, E, f, s_0, S_M)$ abstracting the feasible dynamics of an L-SU-RAS $\Phi =< \mathcal{R}, C, \mathcal{P}, D >$ is defined as follows:

1. The *state set* S consists of all vectors $s = [s_{11}, s_{12}, \ldots, s_{1,l_1}, s_{21}, s_{22}, \ldots, s_{n,l_n}] \in (Z_0^+)^{|\Xi|}$ such that for each $k \in \{1, \ldots, m\}$, $\sum_{i \in \{1,\ldots,n\} \wedge j \in \{1,\ldots,l_i\} \wedge D(\Xi_{ij}) = R_k} s_{ij} \leq C(R_k)$. Each component s_{ij} of s gives the number of instances of process type Π_i that execute stage Ξ_{ij} in state s.

2. The *event set* $E = \{e_{ij} \mid i = 1, \ldots, n; j = 0, \ldots, l_i\}$, where for each $i = 1, \ldots, n$, event e_{i0} represents the *loading* of a new instance of process type Π_i, event e_{in_i} represents the *unloading* of a finished instance of process type Π_i, and event e_{ij}, $j \in 1, \ldots, l_i - 1$, represents the *advancement* of a process instance from stage Ξ_{ij} to stage $\Xi_{i,j+1}$.

3. The *state transition function* $f : S \times E \to S$ is defined by $s' = f(s, e_{qr})$, where the components s'_{ij} of the resulting state s' are given by:

$$s'_{ij} = \begin{cases} s_{ij} - 1 & \text{if } \; i = q \; \text{ and } j = r \\ s_{ij} + 1 & \text{if } \; i = q \; \text{ and } j = r + 1 \\ s_{ij} & \text{otherwise} \end{cases}$$

Furthermore, $f(s, e_{qr})$ is a *partial* function defined only if the resulting state $s' \in S$.

4. The *initial state* $s_0 = \mathbf{0}$, which corresponds to the situation when the system is empty of any process instances.

5. The *set of marked states* S_M is the singleton $\{s_0\}$, and it expresses the requirement for complete process runs. □

L-SU-RAS state safety, the corresponding decision problem and its complexity The notions of state safety and the corresponding decision problem for the above introduced system, can be formally stated as follows:

Definition 2: Consider an L-SU-RAS specified by the 4-tuple $\Phi =< \mathcal{R}, C, \mathcal{P}, D >$, and a state $s \in S$ of the corresponding DFSA $G(\Phi)$.

1. State s is characterized as *safe*, if and only if (*iff*) there exists a feasible event sequence σ that drives the DFSA $G(\Phi)$ from state s to its marked state s_0.

2. The corresponding *L-SU-RAS state safety problem* is the decision problem that, upon input Φ and s, addresses the question of whether state s is safe. □

As remarked in the introductory section, the RAS state safety problem is at the core of maximally permissive deadlock avoidance in any sequential RAS, since the corresponding supervisory control policy should admit a state *s iff* it is safe [7]. The next theorem establishes the NP-completeness of the L-SU-RAS state safety problem, by considering its more restricted version where every resource type has unit capacity. This theorem also implies that, in general, maximally permissive deadlock avoidance for L-SU-RAS is an NP-hard task.[1]

Theorem 1: The restriction of the L-SU-RAS state safety problem to the case where every resource type has unit capacity is NP-complete in the strong sense.

Proof. In order to prove the theorem, (I) first we show that the considered L-SU-RAS state safety problem belongs to the problem class $\mathcal{NP}$, and subsequently (II) we establish its NP-completeness by reducing to it the well-known NP-complete problem of 3-SAT [2].

Proof of (I): We remind the reader that a decision problem is in the class $\mathcal{NP}$ *iff* it can be solved in polynomial time by a Nondeterministic Turing Machine (NDTM) [2]. For the L-SU-RAS state safety problem considered in this theorem, let $n = |\mathcal{P}|$, $L = max(l_1, l_2, \ldots, l_n)$, and notice that the specification of the underlying resource allocation function D requires no more than nL elements. Furthermore, it is clear that state s is safe *iff* there exists at least one event sequence σ such that (i) $s_0 = \delta(s, \sigma)$ and (ii) σ does not include any loading event. The length of each such sequence σ is less than or equal to mL, where $m = |\mathcal{R}|$ (since the total capacity of the system bounds the number of process instances that can be simultaneously active at any point in time). For well-defined problem instances, $m \leq nL$ (since otherwise there would be some completely unutilized resource types). Since the time required by NDTM to generate and verify any of the aforementioned sequences σ is proportional to the length of the sequence, NDTM can solve the considered L-SU-RAS state safety problem in time $O(nL^2)$, and the problem belongs to $\mathcal{NP}$.

Proof of (II): As mentioned above, in order to prove the NP-completeness of the L-SU-RAS state safety problem considered in this theorem, we will provide a reduction from the 3-SAT problem. We remind the reader that the 3-SAT problem can be stated as follows:

3-SAT [2]: Given a set of literals $\mathcal{X} = \{X_1, \bar{X}_1, X_2, \bar{X}_2, \ldots, X_\mu, \bar{X}_\mu\}$ and a set of clauses $\Lambda = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_\nu\}$, each clause being a disjunction $\Lambda_q = y_q^1 \vee y_q^2 \vee y_q^3$, $y_q^1, y_q^2, y_q^3 \in \mathcal{X}$, does there exist $K \subseteq \mathcal{X}$ such that the conjunction of the clauses in Λ is satisfiable, i.e., i) $\forall i = 1, \ldots, \mu$, K does not contain both X_i and $\bar{X}_i$, and ii) $\forall q = 1, \ldots, \nu$, $K \cap \Lambda_q \neq \emptyset$?

For reasons that will become clear in the following, we further assume, without loss of generality, that the enumeration of the literal set, $\mathcal{X} = \{X_1, \bar{X}_1, X_2, \bar{X}_2, \ldots, X_\mu, \bar{X}_\mu\}$, defines a linear order upon the elements of this set, and this order is also respected during the definition of the problem clauses, Λ_q, $q = 1, \ldots, \nu$; in particular, in the following we shall assume that for any given $q = 1, \ldots, \nu$, the literals of the clause Λ_q, y_q^1, y_q^2, y_q^3, are quoted in increasing order with respect to the aforementioned ordering of the literal set $\mathcal{X}$.

Then, given a 3-SAT problem istance $(\mathcal{X}, \Lambda)$, an instance of the considered L-SU-RAS state safety problem, with $\Phi =< \mathcal{R}, C, \mathcal{P}, D >$ and state $s \in S$, can be obtained as follows:

i. The set of resources is given by $\mathcal{R} = \Lambda \cup \{x_{ij} : i = 1, \ldots \mu, \; j = 1, \ldots, \nu\} \cup \{z_j : j = 1, \ldots \nu\}$.

ii. For each resource $R \in \mathcal{R}$, $C(R) = 1$.

iii. The set of process types is given by $\mathcal{P} = \{\Pi_1, \Pi_2, \ldots, \Pi_\nu\}$, where each process Π_q is uniquely associated with the clause $\Lambda_q = y_q^1 \vee y_q^2 \vee y_q^3$.

iv. For each process Π_q, the resource allocation function $D(\Xi_{qj})$, $j = 1, 2, \ldots$, that specifies the resources required at the consecutive stages of Π_q, is given by the *"working procedure"* $W_q = \Lambda_q, Y_q^1, Y_q^2, Y_q^3, \overrightarrow{\Lambda}, z_q, \overleftarrow{\Lambda}$, where for each $k = 1, 2, 3$,

$$Y_q^k = \begin{cases} \overrightarrow{x_i} = x_{i1}, x_{i2}, \ldots, x_{i\nu} & \text{if} \quad y_q^k = X_i \\ \overleftarrow{x_i} = x_{i\nu}, x_{i(\nu-1)}, \ldots, x_{i1} & \text{if} \quad y_q^k = \bar{X}_i, \end{cases}$$

[1] As discussed in the Introduction, the result of Theorem 1 has already been established in the literature [5]. Here we provide an alternative proof that is more concise and (in our opinion) more lucid than the proof of [5], and it will be useful in the developments of the following sections.

Consider the 3-SAT problem instance defined by $\mathcal{X} = \{X_1, \bar{X}_1, X_2, \bar{X}_2, X_3, \bar{X}_3\}$, $\Lambda_1 = X_1 \vee X_2 \vee \bar{X}_3$, and $\Lambda_2 = \bar{X}_1 \vee X_2 \vee X_3$. Then:
$\mathcal{R} = \{\Lambda_1, \Lambda_2\} \cup \{x_{11}, x_{12}, x_{21}, x_{22}, x_{31}, x_{32}\} \cup \{z_1, z_2\}$,
$C(R) = 1, \forall R \in \mathcal{R}$,
$W_1 = \Lambda_1, x_{11}, x_{12}, x_{21}, x_{22}, x_{32}, x_{31}, \Lambda_1, \Lambda_2, z_1, \Lambda_2, \Lambda_1$,
$W_2 = \Lambda_2, x_{12}, x_{11}, x_{21}, x_{22}, x_{31}, x_{32}, \Lambda_1, \Lambda_2, z_2, \Lambda_2, \Lambda_1$,
$s = [100000000100|100000000100]$.

Fig. 1. An example instance of the reduction employed in Theorem 1.

and $\overrightarrow{\Lambda} = \Lambda_1, \Lambda_2, \ldots, \Lambda_\nu$, $\overleftarrow{\Lambda} = \Lambda_\nu, \Lambda_{n-1}, \ldots, \Lambda_1$.
v. s is the state where each process type Π_q has two active instances, respectively executing its first stage (that requires resource Λ_q) and the stage that requires resource z_q.

Figure 1 provides a more concrete example of the above reduction of the 3-SAT problem to the L-SU-RAS state safety problem considered in this theorem. It is evident that the reduction presented above can be performed in polynomial time with respect to the size of the 3-SAT problem. Thus, to prove that the considered L-SU-RAS state safety problem is NP-complete, we only need to establish that an instance of the 3-SAT problem is satisfiable *iff* state s of the above constructed RAS is safe. To do this, let us call the process instances of the constructed state s that are located on the resources Λ_q as *second-instances*, and the process instances located on the resources z_q as *first-instances*, $q = 1, 2, \ldots, n$. Then, we make the following observations:

a. If any second-instance enters section $\overrightarrow{\Lambda}$ before at least one first-instance has left the system, the resulting state is not safe. Moreover, no first-instance can leave the system until all the second-intstances have left their initial locations Λ_q.
b. From (a) it further follows that each sequence of states leading from s to s_0 must contain a state s' such that the second-instance of each process Π_q is located in some resource belonging to a resource subsequence Y_q^k, $k \in \{1, 2, 3\}$.

Next we show that the L-SU-RAS state s is safe *iff* there exists a state s' that (i) is reachable from s, (ii) it is of the type identified in Observation (b), and (iii) it possesses the additional property that no two processes Π_q and Π_r have their second instances in the complementary sections $Y_q^k = \overrightarrow{x_i}$ and $Y_r^l = \overleftarrow{x_i}$. The necessity of this condition for the safety of s is obvious from Observations (a) and (b), and the further remark that any state s' that does not satisfy the last part of the condition will unavoidably lead to a deadlock of the processes Π_q and Π_r. The sufficiency of the aforestated condition for the safety of s can be established by noticing that state s_0 can be reached from state s' according to the following sequence:
1. First, all processes corresponding to first-instances can proceed to completion, one at a time (as long as these process instances are completed one at a time, they can be processed in any order). The feasibility of such a completion sequence is guaranteed by the availability of all the resource units Λ_q, $q = 1, \ldots, \nu$, in state s' (c.f. Observation (b)).
2. Once all first-instances have been completed, the remaining set of processes, corresponding to second-instances, can be advanced to completion one at a time, according to a sequence that is defined by the resources allocated to them in state s'. In order to define this sequence, we first remind the reader that the definition of state s' implies that each of the second-instance processes will be located at a resource x_{ij}, $i = 1, \ldots, \mu$, $j = 1, \ldots, \nu$, that corresponds to a literal X_i or $\bar{X}_i$ in the relevant clause Λ_q (c.f. item (iv) in the construction of state s). Hence, in state s', each second-instance can be mapped to an element of the literal set $\mathcal{X}$. This mapping combined with the ordering of the elements of $\mathcal{X}$ that was introduced in the definition of the 3-SAT problem, induces an ordering on the process set of second-instances. This ordering is partial, since it is possible that state s' contains more than one jobs in the resource sequences $\overrightarrow{x_i}$ or $\overleftarrow{x_i}$, $i = 1, \ldots, \mu$. However, we can obtain a total ordering of the set of second-instances, by ordering any unordered process subsets according to the natural order defined by the second index of the allocated resources x_{ij}. Finally, it is easy to verify that, under the working assumptions, the advancement and completion of the second-instances one instance at a time, and in a decreasing sequence with respect to the process total order constructed above, constitutes a feasible processing sequence and establishes the safety of the considered state s'.

The proof concludes by showing that the existence of the aforementioned state s', that is necessary and sufficient for the safety of the original state s, is equivalent to the existence of a satisficing literal subset K for the considered 3-SAT problem instance $(\mathcal{X}, \Lambda)$. Indeed, if the aforementioned state s' exists, then, set K consists of all the literals X_i (resp., $\bar{X}_i$) which correspond to resource sequences $\overrightarrow{x_i}$ (resp., $\overleftarrow{x_i}$) that are non-empty of process instances in s'. Set K satisfies property (i) posed by the 3-SAT problem, since, by the definition of the considered state s', no two processes Π_q and Π_r have their second instances in the complementary sections $Y_q^k = \overrightarrow{x_i}$ and $Y_r^l = \overleftarrow{x_i}$. It also satisfies property (ii) posed by the 3-SAT problem, due to the fact that state s' satisfies the requirement of Observation (b), i.e., the second-instance of each process Π_q is located in some resource belonging to a resource subsequence Y_q^k, $k \in \{1, 2, 3\}$. In order to show that the reverse is also true, i.e., that the existence of a satisficing set K for the 3-SAT problem enables the construction of a state s' that (i) is reachable from state s and (ii) satisfies the safety requirements postulated in the earlier parts of this proof, we work as follows: State s' is chosen as any state such that the second-instance corresponding to clause Λ_q, $q = 1, \ldots, \nu$, is staged in the resource sequence Y_q^k for some of its literals that belongs to the literal set K. Such a literal will exist for every clause Λ_q since K is a satisficing set of literals. Also, the satisficing nature of K further guarantees that, in the resulting state s', no two processes Π_q and Π_r have their second instances in the complementary sections $Y_q^k = \overrightarrow{x_i}$ and $Y_r^l = \overleftarrow{x_i}$. To establish the reachability of such a state s' from the original state s, we consider the ordering of the set of second-instances corresponding to the clauses Λ_q, $q = 1, \ldots, \nu$, that is induced from the ordering of the literals Y_q^k that were used in the construction of the state s' as explained above. Then, it is easy to see that the second-instances can be advanced from their processing stages in state s to their processing stages in state s', one at a time, and with the sequence of these advancements corresponding to a decreasing sequence according to the aforementioned ordering of these processes. In the particular case that two or more processes correspond to the same literal Y_q^k in state s', their advancement can be performed in any sequence without a problem, since the corresponding resource sequence $\overrightarrow{x_i}$ or $\overleftarrow{x_i}$ has a combined capacity

of ν units (c.f. item (iv) in the construction of the considered L-SU-RAS and of state s). Hence, it can be concluded from all the above discussion that state s of the constructed RAS is safe *iff* the considered instance of 3-SAT has a solution. □

III. Free-Range Vehicular Systems

The system In this section we consider a set of autonomous mobile agents that move in a finite planar area $\mathcal{A} \subset \mathbf{R^2}$, where $\mathbf{R}$ denotes the set of reals. Each agent is represented by a disk of radius $\rho > 0$, and its center follows a pre-specified path that is given in the parametric form: $x^c = x^c(t)$, $y^c = y^c(t)$, $t \in [0,T]$. We assume that the agents stay off the system before they start their travel, and that they are retired from the system upon reaching their destination. However, during their concurrent motion in the system, the agents share the available space, and in order to avoid collisions, they may need to modify their velocity profiles.

Such a coordination can be achieved through a supervisory control scheme that is based on the *tesselation* of the motion plane in a number of areas, called *"cells"* [8]. An agent is said to occupy a certain cell if its disk overlaps with the area corresponding to that cell. In order to avoid collisions, it is required that at any point in time, a cell can be occupied by only one agent. Hence, the cells defined by the proposed tesselation constitute fictitious resources of unit capacity. Furthermore, under the proposed control scheme, the paths designated to the different agents are naturally segmented to a number of stages, with each stage corresponding to a maximal path segment with constant cell (i.e., resource) occupation. The resulting stage sequences define the corresponding *resource (i.e., cell) allocation processes* that must be observed by each agent. Consequently, the system of free ranging agents, can be considered as a Linear-Conjunctive-RAS (L-CON-RAS) [7].[2] The behavior of this RAS can be further formalized by a DFSA in a spirit very similar to that introduced in Section II for the formal representation of the L-SU-RAS behavior; two additional elements that characterize this new DFSA are: (a) the unit capacity of the system resources, which implies that the components of all states $s \in S$ will be of binary nature, and (b) the fact that the RAS process types Π_i are in one-to-one correspondence with the mobile agents of the vehicular system, which implies that each process can have only one active process instance (i.e., $\forall i = 1,\ldots,n$, $\sum_{j=1}^{l_i} s_{ij} \leq 1$).

Yet, the main attributes that define the RAS class considered in this section and differentiate it from any other element of the broader L-CON-RAS class, stem from the fact that the resource allocation and/or de-allocation that takes place during the transition between two consecutive processing stages, must observe a "resource proximity" relation that is defined by the adopted tesselation. More specifically, in the considered RAS systems, the allocation corresponding to a particular processing stage must be interpretable as the occupation of a number of neighboring cells by the corresponding mobile agent, while the variation of the allocations between two consecutive processing stages must be interpretable as the occupation of some new neighboring cells and/or the release of some previously held ones, during the agent motion. The sub-class of L-CON-RAS that possesses the aforementioned additional features will be characterized as FREE-RANGE-RAS. Following the notation introduced in Section II, an element of this new RAS class will be denoted by $\Phi = (\mathcal{R}, \mathcal{P}, \mathcal{D})$ and it will be further presumed that the allocation function $\mathcal{D}$ presents the aforementioned attributes with respect to the underlying tesselation.[3]

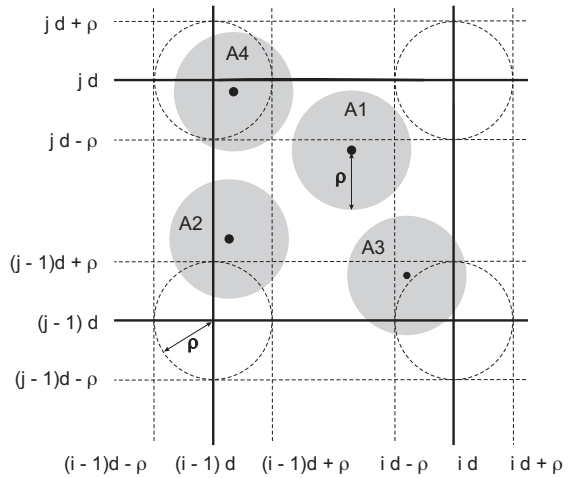

Fig. 2. Depending on the position of its central point, an agent occupies one cell (as $A1$), two cells (as $A2$), three cells (as $A3$), or four cells (as $A4$).

Concluding the presentation of the considered vehicular systems and their abstracting RAS, we notice that, for most practical purposes, the adopted tesselation is obtained by applying a rectangular grid on the motion plane with step sizes (considerably) greater than 2ρ. This will also be the type of tesselation that we shall consider in the rest of our discussion. It should be clear (see Figure 2) that in the case of such a rectangular tesselation, the number of cells, k, occupied at a time by a single agent is $1 \leq k \leq 4$. Furthermore, for reasons that will become clear in the sequel, we distinguish a particular type of the agent paths that consist only of horizontal and vertical segments joining the centers of the consecutive cells that they lie on; we shall refer to these paths as *central vertical-horizontal paths*. Such a path can be specified by the sequence of the traversed cells, $p = R_1, R_2, \ldots, R_w$, as indicated in the example depicted in Figure 3. Also, the motion process of an agent that follows a central vertical-horizontal path specified by $p = R_1, R_2, \ldots, R_{w-1}, R_w$ consists of $2w-1$ stages that, respectively, require the following resource sets: $\{R_1\}$, $\{R_1, R_2\}$, $\{R_2\}$, …, $\{R_{w-1}\}$, $\{R_{w-1}, R_w\}$, $\{R_w\}$. For example, the motion of an agent along the path depicted in Figure 3 consists of nine stages, which require the respective cell subsets: $\{R1\}$, $\{R1, R4\}$, $\{R4\}$, $\{R4, R5\}$, $\{R5\}$, $\{R5, R6\}$, $\{R6\}$, $\{R6, R3\}$ and $\{R3\}$. The reader should particularly notice that the specification of a central vertical-horizontal path p determines uniquely the underlying resource allocation process. All the above observations will become useful in the next section, where we consider the state safety problem for the FREE-RANGE-RAS and its computational complexity.

The state safety problem and its complexity The state

[2]The characterization "conjunctive" denotes the fact that certain processing stages might require the simultaneous – or conjunctive – allocation of more than one resource units.

[3]Since, in the case of FREE-RANGE-RAS, all resources have unit capacity, function C does not provide any additional significant information, and therefore, it was dropped from the tuples defining the corresponding problem instances.

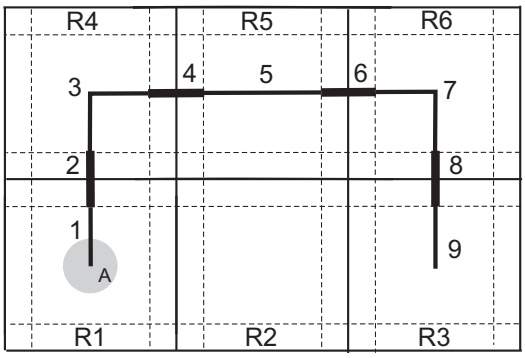

Fig. 3. An example of a central vertical-horizontal path. The path is uniquely specified by the sequence of the traversed cells, $R = R1, R4, R5, R6, R3$, and it induces nine stages for the agent's motion process.

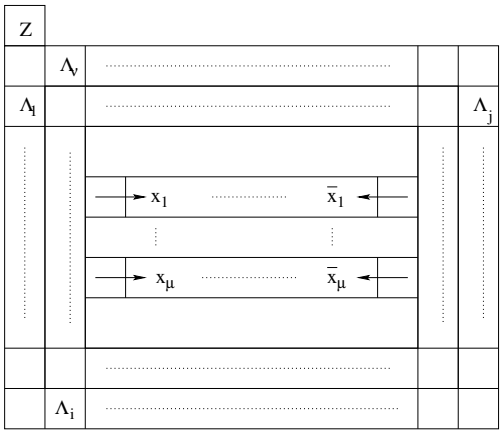

Fig. 4. Illustration for the proof of Theorem 2: Highlighting the basic topology of the agent paths that define the FREE-RANGE-RAS employed in the relevant reduction from 3-SAT.

safety problem for the vehicular system introduced in the previous section can be stated as follows:

Definition 3: Given a FREE-RANGE-RAS specified by the triplet $\Phi =< \mathcal{R}, \mathcal{P}, \mathcal{D} >$, and a state $s \in S$, is the marked state s_0, where all vehicles have successfully completed their trips and retired from the system, reachable from s?

Next we prove the following:

Theorem 2: For rectangular tesselations with step sizes greater than or equal to 2ρ, the problem of FREE-RANGE-RAS state safety is NP-complete in the strong sense.

Proof. Similar to the case of Theorem 1, we establish the result of Theorem 2 in two major steps: (I) First we show that FREE-RANGE-RAS state safety is in class $\mathcal{NP}$, and (II) next we show that the NP-complete problem 3-SAT is polynomially reducible to FREE-RANGE-RAS state safety.

Proof of (I): Notice that the size of an instance of the considered problem is essentially determined by the size of the data required to specify function $\mathcal{D}$, which is proportional to the total number of processing stages, $|\Xi|$. Since this number bounds also the length of any event sequence σ that can constitute a feasible solution to the considered state safety problem, it follows that the problem can be solved by an NDTM in polynomial time, and consequently the FREE-RANGE-RAS state safety problem belongs to the class $\mathcal{NP}$.

Proof of (II): To prove the NP-completeness of the FREE-RANGE-RAS state safety problem, we shall provide a reduction from the 3-SAT problem that can be performed in polynomial time with respect to the size of this last problem. More specifically, an instance of the FREE-RANGE-RAS state safety problem, specified by the RAS $\Phi =< \mathcal{R}, \mathcal{P}, \mathcal{D} >$ and state $s \in S$, can be obtained from a 3-SAT instance $(\mathcal{X}, \Lambda)$ as follows:

a. The set of resources $\mathcal{R}$ consists of the set of (square) cells depicted in Figure 4.

b. The set of processes is given by $\mathcal{P} = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_\nu\} \cup \{Z\}$.

c. The resource allocation function $\mathcal{D}$ is implied by the topology of the paths followed by the agents, which involves (c.f. Figure 4) :

- two nested rings,
- μ "bridges", each corresponding to one of the variables $X_1, \ldots, X_\mu$ of the 3-SAT problem and consisting of ν cells,
- another cell, marked by Z in the figure.

The agents executing the processes Λ_q, $q = 1, \ldots, \nu$, are distributed at various (arbitrary) locations at the outer ring and each of them has to pursue the following central horizontal-vertical path:

(i) First it enters the inner ring through its cell next to it.

(ii) Subsequently it moves clockwise on that ring until it meets the entry point of the bridge corresponding to the first variable in clause Λ_q. If this variable is not negated in the clause, the bridge must be crossed from left to right; otherwise, it must be crossed from right to left.

(iii) Upon exiting the first bridge, the agent continues moving clockwise on the inner ring until it enters the bridge corresponding to its second variable, and then it continues in the same way with the bridge corresponding to its third variable.

(iv) Upon exiting the third bridge, the agent must (i) perform a complete loop of the entire inner ring, moving in the clockwise direction, (ii) pass to the outer ring through the cell held by Λ_ν in the figure, (iii) traverse clockwise the entire outer ring and eventually terminate at the cell held by Z.

The agent executing the process annotated as Z must move in the counter-clockwise sense, initially traversing the outer ring, then entering the inner ring from the cell next to Λ_ν, and finally traversing this entire ring before terminating in the cell where the ring was entered.

d. In state s, all the agents are at the first stages of the above described routes.

Clearly, the above construction can be polynomial with respect to the number of literals and clauses of the underlying 3-SAT problem. Next we establish that the considered 3-SAT problem instance has a solution *iff* the FREE-RANGE-RAS state s, that was defined through the above construction, is safe. For this, the reader should notice the following:

i. Since, in state s, process Z occupies a cell required by each process Λ_q for its completion, no Λ_q can complete until Z advances to another stage. Furthermore, since process Z moves on the two rings in a direction opposite to that of the motion of processes Λ_q, no state such that process Z is in one of the two rings while any other process Λ_q executes the last part of its route, as specified by item (c-iv) above, is safe.

ii. A little more reflection will reveal that the target state, where all processes have run to completion, is reachable from the considered state s, *iff* it is possible to reach a state s' such that all processes Λ_q are accomodated on the bridges and each bridge is

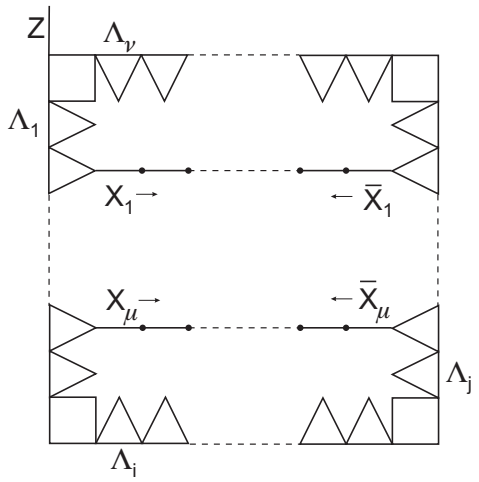

Fig. 5. Illustration for the proof of Theorem 3: Highlighting the basic topology of the agent paths that define the AGV-RAS employed in the relevant reduction from 3-SAT.

occupied by processes that traverse it in the same direction.
iii. But then, it is easy to see that in the state s' identified in Observation (ii) above, the bridges and their two directions of traversal play the same role with that played by the resource sequences $\overrightarrow{x_i}$ and $\overleftarrow{x_i}$ in the proof of Theorem 1, and therefore, the rest of the proof proceeds in exactly the same manner.

IV. The State Safety Problem in AGV Systems

Unlike what is happening with the case of free-range systems, in an AGV system vehicles move in a guidepath network, whose topology is invariant and can be abstracted with a graph. In order to avoid physical collisions of the various vehicles, the entire network is partitioned into zones, represented by the graph edges, and only one vehicle is allowed in any zone at any point in time. In this section we consider the class of *open* and *statically routed* AGV systems, characterized by the following two attributes: (i) A vehicle enters the guidepath network when assigned a task and it leaves this network, retiring to a docking station, when the task is completed. (ii) The vehicle route is completely determined *a priori*, upon the task initiation, and it is defined by a specific sequence of zones to be visited by the vehicle until its retirement to the docking station.

Notice that such a system can be considered as a RAS where the system resources are defined by the set of zones and the system processes are defined by the set of vehicles following their assigned paths. In fact, the resulting RAS constitutes a sub-class of L-SU-RAS, distinguished by the following two additional attributes: (a) each resource $R \in \mathcal{R}$, possesses unit capacity; (b) the resource allocation function $\mathcal{D}$ must observe the network topology, i.e., the resources required for the consecutive stages of a process must form a path in the graph representing the guidepath network. We shall characterize such a RAS system as an AGV-RAS, and the corresponding problem to decide whether or not a particular state in it is safe, as the AGV-RAS state safety problem. With the results of the previous sections, it is not hard to prove the following:

Theorem 3: The AGV-RAS state safety problem is NP-complete in the strong sense.

Proof: The proof of this theorem is analogous to that for Theorem 2. The topology of the agent paths that define the AGV-RAS employed in the relevant reduction from 3-SAT, is presented in Figure 5. The structure of the system and of the vehicle processes implements the same ideas as those considered in the corresponding result for the FREE-RANGE-RAS. The guidepath network consists of two rings and μ bridges, and the routes of the vehicles are the same as assumed in the proof of Theorem 2. Thus, by the same argumentation, Theorem 3 is also true.

V. Conclusions

In this paper, we provided some new results regarding the computational complexity of the "state safety" problem, as it arises in the context of some multi-vehicle traffic systems that are encountered in modern technological applications. In the process of deriving these results, we also developed a new proof for the NP-completeness of state safety in the class of L-SU-RAS, which is more concise and more lucid than the currently existing one. Two additional, similar problems that remain open, concern the computational complexity of state safety in (a) *dynamically routed* AGV and free-range vehicular systems, where a vehicle can be routed dynamically to any zone / cells neighboring its current location, and (b) in *closed* AGV and free-range vehicular systems where agents never retire from the domain of their motion, but they remain in it between the assignment of two consecutive trips. Furthermore, it is also interesting to identify any special structure under which the FREE-RANGE-RAS and AGV-RAS state safety problems, considered in this work, acquire polynomial complexity. Such a line of research can leverage and expand results of similar type for the L-SU-RAS state safety problem reported in [7] (c.f., Chapter 3).

References

[1] M. P. Groover. *Fundamentals of Modern Manufacturing: Materials, Processes and Systems*. Prentice Hall, Englewood Cliffs, N.J., 1996.

[2] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, MA, 1979.

[3] A. Kobetski. *Optimal Coordination of Flexible Manufacturing Systems with Automatic Generation of Collision and Deadlock-Fee Working Schedules*. PhD thesis, Chalmers University of Technology, Goteborg, Sweden, 2008.

[4] S. M. La Valle and S. A. Hutchinson. Optimal motion planning for multiple robots having independent goals. *IEEE Trans. on Robotics & Automation*, 14:912–925, 1998.

[5] M. A. Lawley and S. A. Reveliotis. Deadlock avoidance for sequential resource allocation systems: hard and easy cases. *Intl. Jrnl of FMS*, 13:385–404, 2001.

[6] S. A. Reveliotis. Conflict resolution in AGV systems. *IIE Trans.*, 32(7):647–659, 2000.

[7] S. A. Reveliotis. *Real-time Management of Resource Allocation Systems: A Discrete Event Systems Approach*. Springer, NY, NY, 2005.

[8] S. A. Reveliotis and E. Roszkowska. Conflict resolution in multi-vehicle systems: A resource allocation paradigm. In *Proceedings of IEEE CASE 2008*, pages –. IEEE, 2008.

[9] E. Roszkowska. High-level motion control for workspace sharing mobile robots. In *Robot Motion and Control*, pages 427 – 435. Springer, LNCIS vol. 360, 2007.

[10] E. Roszkowska and S. Reveliotis. On the liveness of guidepath-based, zoned-controlled, dynamically routed, closed traffic systems. *IEEE Trans. on Automatic Control*, 53:1689–1695, 2008.

[11] N. Wu and M. Zhou. Resource-oriented Petri nets in deadlock avoidance of AGV systems. In *Proceedings of the ICRA'01*, pages 64–69. IEEE, 2001.