UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Matematica "Tullio Levi – Civita"

Master Degree in Mathematics

Integer Programming Formulations and Valid Inequalities for Fleet Quickest Routing on Grids

Supervisor: Prof.ssa Carla De Francesco
Co-Supervisor: Prof. Luigi De Giovanni

Candidate: Giulia Bortoluz
Badge number: 1203099

Academic Year 2019-2020

To Giulia & Davide

Contents

1 Introduction

This work regards Linear Programming Models and their application to the "Fleet Quickest Routing on Grids" problem.
Linear Programming Modelling is a huge and important field of Applied Mathematics and it is used in several contexts, for example Economics, Biology, Genetics and Transport Networks.
This field developed especially during the second half of the previous century, relatively military planning activities. In 1947, american mathematician George B. Dantzig developed the first model of Linear Programming: it consisted in finding how much some industries had to work in order to satisfy a certain demand, respecting prefixed badgets. Dantzig stated some relations between a first group of objects (which were the parameters of the model, representing the demands and the badgets) and a second group of objects (which were the variables of the model, representing the work of the industries). The result was a problem in which he minimized the cost of the whole process, guaranteeing that every demand was satisfied or, in other words, a problem in which he minimized a linear function subject to linear equations and inequalities. Dantzig understood that the Simplex method was a valid way to resolve such a problem and that this problem's structure could be applied to several situations: in this way, Linear Programming Modelling took hold.
This work contains a section in which we develop some of the most useful generalities about Linear Programming and Integer Linear Programming. We show the structure of a Linear Programming Problem, we define what is a solution for such a problem and we provide some results about the set of all solutions, treating it especially from a geometric point of view.
We apply Integer Linear Programming Modelling to a particular problem: the "Fleet Quickest Routing on Grids" (FQRP). It consists in a grid where some vehicles move in order to reach their destinations as soon as possible avoiding conflicts. This problem is also quite famous and it arises in many contexts: the routing of automated guided vehicles in container terminals, the coordination of ground service vehicles in airport aprons, aircraft taxiing operations, etc. We provide a consistent summary of the main results about this problem, highlighting those which are useful for this work. Afterwards, we give 3 different Integer Linear Programming Models to resolve the FQRP and also the codes for their implementation with software AMPL. We deeply analyze the models: we explain their structural characteristics and the meaning of their constraints.
As we will see, we decided to focus our attention on the third model, trying

to refine it. Indeed, when we solve the Continuous Relaxation of the FQRP with the third formulation, AMPL provides solutions which are not feasible. The goal of this work is finding some valid inequalities that can be added to the model: in this way, we cut away some solutions from the feasible region of the continuous relaxation. We give 7 sets of valid inequalities and for each inequality we show that it actually cuts away some points. We tested the inequalities with a lot of instances of the problem, formed by different amount of vehicles. We used both a common AMPL-license (CPLEX 12.8.0.0) and also the AMPL-license installed in the computers of the computer science laboratory of Torre Archimede (ILOG AMPL), because this last one can work with bigger instances than the one available for generic users.

We introduce the use of software PORTA: it works with the representations of polyhedra constituting the feasible regions of Linear Programming Models. We used it in order to have some hints for the search of valid inequalities.

Finally, we provide a computational study about our new inequalities in order to see how they affect the implementation of the third model.

2 The "Fleet Quickest Routing on Grids" Problem

2.1 Introduction

The "Fleet quickest routing on grids" problem we are concerned with is related to the movement of some vehicles on an undirected grid graph or Manhattan graph $G = (V, E)$. In G we recognize m rows and n columns, V is the set of nodes and E is the set of edges. Assume the columns are numbered from the left to the right and the rows are numbered from the bottom to the top and, assume also the number of vehicles coincides with the number of columns. Each vehicle has a specific starting-point, which is at the bottom of the grid (row 1), and a specific end-point, which is at the top of the grid (row m). Two different vehicles cannot share neither the starting-point nor the end-point. Assume time is discrete and, at each moment in time, every vehicle has to move either horizontally or vertically, until it reaches its final position. Each edge of the graph (both horizontal and vertical) takes one unit of time to be traversed. In particular, each vehicle has to reach its final destination in the shortest possible time. The goal of the FQRP is to minimize the overall time, i.e. the time n vehicles need to perform their movements. We can assign a number to the starting-positions of the vehicles from 1 to n in the first level. The final positions of the vehicles is a permutation:

$$\sigma = \sigma_1, .., \sigma_n$$

of the starting-positions $\{1, .., n\}$. Then, every vehicle has to move from position i at level 1 to position σ_i at level m and, the quickest path of a generic vehicle can be defined as follow.

Definition 1 *(Manhattan Quickest Path) A path of vehicle $i \in \{1, .., n\}$ is a Manhattan Quickest Path (or a minimum-length path) if and only if vehicle i performs only one move (horizontal or vertical) at each moment in time and the path does not contain steps in opposite directions.*

We can immediately notice that we are using equally the concepts that a path is the quickest one in term of time and that a path is the shortest one in term of Manhattan distance. The shortest path for each vehicle is easy to be determined and it is not unique generally. However, if each vehicle tries to reach its destination moving on one of its shortest paths, it would

be possible that two or more vehicles create a conflict, i.e. some vehicles can crash with the others. Conflicts must be avoided and we always search for fleet patterns that are conflict-free. There are two different types of conflicts.

Definition 2 *(Arc-conflict) An arc-conflict arises when at least two different vehicles use the same edge at the same time.*

Definition 3 *(Node-conflict) A node-conflict arises when at least two different vehicles use the same node at the same time.*

The grid $m \times n$ is our workplace and its dimensions are established casualty; we only require that $m \geqslant 2$. Consequently, it is obvious that the units of time T needed to be sure that the n vehicles reach their destination can be defined as follow:

$$T = \max_i |i - \sigma(i)| + m - 1$$

The last expression allows us to explain why the value of m is important: such a number affects both T and the existence of a solution. As already said, we want to minimize the overall time T, therefore the value of m should be minimum but high enough to guarantee the existence of a solution. In order to do so, we introduce a variable, called z, which represents the highest level in which a horizontal move takes place. In this way, we do not change the value of m, which is fixed and established at the beginning, but we work on the value of z which is a variable. In light of this, the overall time T can be rewritten as follow:

$$T = \max_i |i - \sigma(i)| + z - 1$$

Since the quantity $\max_i |i - \sigma(i)| - 1$ depends on the specific instance of the problem and is fixed for every instance, we are interested in finding the value of z that minimizes T but also guarantees the existence of a free-impact solution. In this way we make the vehicles move on their shortest possible path and we force all their horizontal movements to take place in the lowest possible levels. If $m \geqslant z$, we can avoid to consider all the lines above level z, because only vertical moves take place there.

2.2 Common Notation

Let's consider N the set of all vehicles (indexed by k). For each vehicle $k \in N$, let's consider two values: $\alpha(k)$ and $\omega(k)$ which are respectively the

column-indexes of the starting-point and the end-point of k. Set N can be divided into three subsets:

$$S = \{k \in N : \alpha(k) = \omega(k)\}$$

$$R = \{k \in N : \alpha(k) < \omega(k)\}$$

$$L = \{k \in N : \alpha(k) > \omega(k)\}$$

It will be useful to recall the last two sets in this way:

$$P = R \cup L$$

We can immediately notice that the vehicles of S can proceed straight to their final destination, so we do not have to find their quickest path because they have only one shortest path (the one composed of only vertical movements). Vehicles of R have to perform a certain number of moves to the right (for each $k \in R$, they are exactly $\omega(k) - \alpha(k)$ moves) and vehicles of L have to perform a certain number of moves to the left (for each $k \in L$, they are exactly $\alpha(k) - \omega(k)$ moves). Vehicles of S never come in conflict with the others, by contrast vehicles of R can come in conflict only with vehicles of L and vice versa.
We use the set of row-indexes:

$$J = \{1, .., m-1\}$$

and the sets of column-indexes related to every vehicle $k \in P$:

$$I(k) = \{\min\{\alpha(k), \omega(k)\}, .., \max\{\alpha(k), \omega(k)\}\} \quad \forall k \in P$$

We define the current position of a vehicle through a couple of numbers (i, j): they are respectively the column and the row of the vehicle.
Given two vehicles $p \in R$ and $q \in L$, let's define $c(p, q)$ the column:

$$c(p,q) = \begin{cases} [\alpha(p) + \alpha(q)]/2 & \textit{if } \alpha(p) + \alpha(q) \textit{ even} \\ \lfloor [\alpha(p) + \alpha(q)]/2 \rfloor & \textit{if } \alpha(p) + \alpha(q) \textit{ odd} \end{cases}$$

We can denote with C the set of pairs of vehicles that could create a conflict:

$$C = \{(p,q) \in R \times L : \alpha(p) < \alpha(q),\ \omega(q) \leqslant c(p,q),\ \omega(p) \geqslant \begin{cases} c(p,q) & \textit{if } \alpha(p) + \alpha(q) \textit{ even} \\ c(p,q) + 1 & \textit{if } \alpha(p) + \alpha(q) \textit{ odd} \end{cases} \}$$

Remark 4 *Two vehicles may compete for the same node only if one of them belongs to R and the other to L. Furthermore, a conflict could occur only in a node of a specific column. In fact, suppose $p \in R$ and $q \in L$ then a node-conflict can occur only in a node of the column $c(p,q)$ provided that:*

$$\begin{cases} \alpha(p) < \alpha(q) \\ \alpha(p) + \alpha(q) \quad even \\ \omega(p) \geq c(p,q) \\ \omega(q) \leq c(p,q) \end{cases} \tag{1}$$

Vice versa, let $p \in R$ and $q \in L$, let $c(p,q)$ be the column defined like above and let the conditions in (1) hold, then vehicles p and q could compete for the same node in column $c(p,q)$ (See Appendix 2, Figure 1 and 2 for a node-conflict and an intersection of two routes without node-conflict respectively).

This condition for a node-conflict is both sufficient and necessary. It is obvious that a couple of vehicles for which such conditions in (1) hold could create a node-conflict. The vice versa is not so obvious so we provide a proof.

Proof. Assume p and q create a node-conflict. Then $p \in R$ and $q \in L$ because they must move in opposite directions in order to create the condition for a conflict.
It holds $\alpha(p) < \alpha(q)$ otherwise vehicles do not move towards each other and they could never create a conflict.
The conflict column must be equidistant from $\alpha(p)$ and $\alpha(q)$, therefore it must be $c(p,q) = [\alpha(p)+\alpha(q)]/2$ and the quantity $\alpha(p)+\alpha(q)$ must be even.
Last two conditions hold because the vehicles must reach a common point at the same time. If one of these conditions do not hold, the vehicles cannot meet in column $c(p,q)$. ■

Let's define the set of all possible node-conflict pairs, called C_{even}:

$$\begin{aligned} C_{even} = \{(p,q) \in R \times L : \alpha(p) + \alpha(q) \ even,\ \alpha(p) < \alpha(q), \\ \omega(p) \geqslant c(p,q),\ \omega(q) \leqslant c(p,q)\} \end{aligned}$$

Remark 5 *Two vehicles may compete for the same edge only if one of them belongs to R and the other to L. Furthermore, an edge conflict could occur only in an edge joining two specific columns. In fact, suppose $p \in R$ and $q \in L$ then an edge conflict can occur only in an edge joining a node of*

column $c(p,q)$ with a node of column $c(p,q)+1$, provided that:

$$\begin{cases} \alpha(p) < \alpha(q) \\ \alpha(p) + \alpha(q) \quad odd \\ \omega(p) \geq c(p,q) + 1 \\ \omega(q) \leq c(p,q) \end{cases} \tag{2}$$

Vice versa, let $p \in R$ and $q \in L$, let $c(p,q)$ be the column defined like above and let the conditions in (2) hold, then vehicles p and q could compete for the same arc joining columns $c(p,q)$ and $c(p,q)+1$ (See Appendix 2, Figure 3 and 4 for an arc-conflict and an intersection of two routes without arc-conflict respectively).

Also in this case, the condition for an edge-conflict is both sufficient and necessary. If all these facts in (2) are true, it follows that such couple of vehicles creates a conflict. The vice versa is not trivial, so we provide a proof.

Proof. Assume p and q create an arc-conflict. Then $p \in R$ and $q \in L$ because they must move in opposite directions in order to create the condition for a conflict.
It holds $\alpha(p) < \alpha(q)$ otherwise vehicles do not move towards each other and they could never create a conflict.
The two edges of the arc in which they create a conflict must be respectively equidistant from $\alpha(p)$ and $\alpha(q)$, consequently such an arc must join columns $c(p,q) = \lfloor [\alpha(p) + \alpha(q)]/2 \rfloor$ and $c(p,q)+1$ and the quantity $\alpha(p)+\alpha(q)$ must be odd to guarantee this property.
Last two conditions hold because both the vehicles must reach a side of any arc between $c(p,q)$ and $c(p,q)+1$ in order to create an arc-conflict there, in particular p must reach the rightmost side of such an arc and q must reach the leftmost side of such an arc. ■

Let's define the set of all possible arc-conflict pairs, called C_{odd} :

$$C_{odd} = \{(p,q) \in R \times L : \alpha(p) + \alpha(q) \ odd, \ \alpha(p) < \alpha(q), \\ \omega(p) \geqslant c(p,q) + 1, \ \omega(q) \leqslant c(p,q)\}$$

It is obvious that:

$$C = C_{even} \sqcup C_{odd}$$

where symbol $\sqcup$ means disjoint union; moreover we can use these two sets to rewrite an equivalent definition of $c(p,q)$:

$$c(p,q) = \begin{cases} [\alpha(p) + \alpha(q)]/2 & \text{if } (p,q) \in C_{even} \\ \lfloor [\alpha(p) + \alpha(q)]/2 \rfloor & \text{if } (p,q) \in C_{odd} \end{cases}$$

Finally, we call z the integer variable representing the last level where horizontal moves take place.

2.3 Main Results on the FQRP

In this part we provide a brief summary of the main results on the FQRP. In particular, we focus our attention on those results that can be considered a completion of this work. The goal of this part is showing that, given any instance of the FQRP with n vehicles, it is always possible to state which is the number of rows of the workplace grid that are necessary and sufficient for finding a feasible solution of the considered instance when vehicles perform all their horizontal steps on one and only one level and each level allows horizontal movements in one direction only. All our tests have been done guaranteeing a sufficient number of rows provided by this section.

2.3.1 More types of Node-Conflicts

First of all, it is common to distinguish different types of node-conflicts.

Definition 6 *(Node-conflicts: type A) Two vehicles p and q create an A-type node conflict if they satisfy one of the following relations:*

$$\alpha(p) < \alpha(q) \leqslant \omega(q) < \omega(p) \quad or \quad \omega(q) < \omega(p) \leq \alpha(p) < \alpha(q)$$

Remark 7 *When the first relation is valid, the only node where an A-type conflict may arise is $(\omega(q), m)$; if we have the second relation, the only potential conflict node is $(\omega(p), m)$. In both cases, there is no node below level m in which the two vehicles can arrive at the same time. As a consequence, to avoid the occurrence of an A-type conflict, it is necessary that the destination node of q (resp. p) does not belong to the path of vehicle p (resp. q).*

Remark 8 *For vehicles in set S, the only potential type of conflict is A-type.*

Definition 9 *(Node conflicts: type B) B-type conflicts occur when two vehicles p and q, with $\alpha(p) < \alpha(q)$, must perform their horizontal steps in opposite directions, under the following conditions:*

$$\begin{cases} \alpha(p) < \omega(p) \\ \alpha(q) > \omega(q) \\ \omega(q) < [\alpha(p) + \alpha(q)]/2 < \omega(p) \\ \alpha(q) - \alpha(p) = 2l \quad l > 0 \quad \textit{integer} \end{cases}$$

i.e. the horizontal distance between p and q starting nodes is an even number and the destinations of p and q are located, respectively, to right and left of the median column $[\alpha(p) + \alpha(q)]/2$.

Remark 10 *B-type conflict relations are symmetrical: if p is involved in a B-type conflict with q, then the opposite also holds. Moreover, the existence of a B-type conflict for vehicle p is not necessarily univocal, since p could also enter into B-type conflict with other vehicles.*

Remark 11 *Notice that any routing rule that allows movements in only one direction per level makes B-type conflicts and arc-conflicts impossible.*

Definition 12 *(Node conflicts: type C) C-type conflicts occur when two vehicles p and q, with $\alpha(p) < \alpha(q)$, must perform their horizontal steps in opposite directions and either one of the following relations holds:*

$$\begin{cases} \omega(p) = [\alpha(p) + \alpha(q)]/2 \\ \omega(q) < [\alpha(p) + \alpha(q)]/2 \end{cases}$$

or

$$\begin{cases} \omega(q) = [\alpha(p) + \alpha(q)]/2 \\ \omega(p) > [\alpha(p) + \alpha(q)]/2 \end{cases}$$

This means that we describe the vehicle whose destination is a node of column $[\alpha(p) + \alpha(q)]/2$ as subject to C-type conflict.

Notation 13 *To express this conflict relation between vehicles p and q we use the notation $p(q)$ if $\omega(p) = [\alpha(p) + \alpha(q)]/2$, or, alternatively, $q(p)$ if $\omega(q) = [\alpha(p) + \alpha(q)]/2$.*

Proposition 14 *In order to avoid the collision which could arise from the occurrence of $p(q)$, when $\alpha(p) < \alpha(q)$, the first node of column $\omega(p)$ belonging to the path of vehicle p must be $(\omega(p), l)$ and the last node of the same column*

belonging to the path of vehicle q must be $(\omega(p), l')$, satisfy the relation $l > l'$ (In other words, the paths of the two vehicles p and q are such that in column $[\alpha(p)+\alpha(q)]/2$, vehicle p always remains at a higher level than that of vehicle q).

Proof. If we have vehicles p and q with $\alpha(p) < \alpha(q)$ and $p(q)$, it follows that:

$$\begin{cases} \omega(p) = [\alpha(p) + \alpha(q)]/2 \\ \omega(q) < \omega(p) \end{cases}$$

Therefore column $\omega(p)$ is equidistant from column $\alpha(p)$ and $\alpha(q)$. We distinguish two cases:

1. If $l \leqslant l'$ then both vehicles reach the node $(\omega(p), l')$ at time $t = \frac{\alpha(q)-\alpha(p)}{2} + l' - 1$. In fact, remembering that they are equidistant from column $\omega(p)$, if $l = l'$ then both vehicles are on the node $(\omega(p), l')$ at the same time. On the other side, if $l < l'$, then vehicle p reaches column $\omega(p)$ at time $t = \frac{\alpha(q)-\alpha(p)}{2} + l - 1$ and it has only vertical steps left. Moving vertically in the following time units, it occupies the nodes from $(\omega(p), l+1)$ to $(\omega(p), l')$ and so q cannot avoid to use a node in which vehicle p is still.

2. If $l > l'$, then vehicle q leaves the column $\omega(p)$ before than the arrival of vehicle p, i.e. at time $t = \frac{\alpha(q)-\alpha(p)}{2} + l - 1$. In this way, the occurrence of a conflict is avoided.

■

Remark 15 *Conflict relation $p(q)$ is asymmetric because, when vehicle p is subject to C-type conflict with vehicle q, q cannot be subject to the same conflict with p. Moreover, it is impossible to have two simultaneous relations $p(q)$ and $p(q')$, with $q \neq q'$.*

Remark 16 *Consider a C-type conflict $p(q)$. Then, in the column $[\alpha(p) + \alpha(q)]/2$, a B-type conflict involving vehicles p or q cannot occur.*

2.3.2 C-types Conflicts Paths

We introduce a particular structure of conflicts, called C-type conflict paths.

Definition 17 *(C-type Conflict Paths) We call C-type conflict path (for brevity "conflict path") any sequence of vehicles having a succession of C-type conflicts $z_1(z_2), z_2(z_3), ..., z_k(z_{k+1})$. Notice that in such a conflict path $z_1(z_2), z_2(z_3), ..., z_k(z_{k+1})$, all vehicles with odd (resp. even) index move in the same direction. As a particular case, a conflict path can consist of a single C-type conflict: $z_1(z_2)$.*

Definition 18 *(Length of a C-type Conflict Path) We define Length of a conflict path as the number of vehicles subject to C-type conflict in the given path. The length of the generic conflict path $z_1(z_2), z_2(z_3), ..., z_k(z_{k+1})$ is k.*

Notation 19 *Considering a generic vehicle z_i belonging to a conflict path like above, we denote h_{z_i} the level of the first node in column $\omega(z_i)$ that belongs to the path of z_i , and h'_{z_i} the level of the last node in column $\omega(z_{i-1})$ that belongs to the path of z_i.*

Remark 20 *We provide two properties that must be true for a feasible solution of the FQRP:*

$$\begin{cases} h_{z_i} > h'_{z_{i+1}} \\ h_{z_{i+1}} > h'_{z_{i+1}} \end{cases}$$

Assume $z_i(z_{i+1})$, the first level of column $\omega(z_i)$ that belongs to z_i must be greater than the last level of column $\omega(z_i)$ that belong to z_{i+1} in order to avoid conflicts. The second rule is a bit more difficult: column $\omega(z_i)$ must be located between columns $\omega(z_{i+1})$ and $\alpha(z_{i+1})$. As a consequence, when z_{i+1} reaches node $(\omega(z_i), h'_{z_{i+1}})$, two are cases: if z_{i+1} does not perform any vertical steps in the section of its paths from column $\omega(z_i)$ to column $\omega(z_{i+1})$, then it must be $h_{z_{i+1}} = h'_{z_{i+1}}$. Otherwise, i.e. if z_{i+1} performs one or more vertical steps in such a section, it must be $h_{z_{i+1}} > h'_{z_{i+1}}$.

Remark 21 *$k+1$ levels are sufficient to route the vehicles belonging to a conflict path whose length is k. Indeed, these $k+1$ vehicles could be routed using a very simple rule. Each vehicle performs all its horizontal steps on one and only one level and each level allows performing its horizontal steps to one and only one of these vehicles. The result is an 1-1 assignment of vehicles to levels, so that $k+1$ levels are required and sufficient.*

Proposition 22 *Given a path $z_1(z_2), z_2(z_3), ..., z_k(z_{k+1})$, the distance $|\alpha(z_{k+1}) - \alpha(z_k)|$ is higher than the distance between any other couple belonging to the conflict-path.*

Proof. WLG assume $\alpha(z_i) - \omega(z_i) < 0$ if i is odd and $\alpha(z_i) - \omega(z_i) > 0$ otherwise. In this case we have:

$$\begin{array}{rcl} \omega(z_1) & > & \omega(z_2) \\ \omega(z_2) & < & \omega(z_3) \\ \omega(z_3) & > & \omega(z_4) \\ & & ... \end{array}$$

where the last inequality is $\omega(z_k) < \omega(z_{k-1})$ if k is even, $\omega(z_k) > \omega(z_{k-1})$ otherwise. Recalling that $\omega(z_i) = \frac{\alpha(z_i)+\alpha(z_{i+1})}{2}$ and substituting it in the previous inequalities, we obtain:

$$\begin{cases} \alpha(z_k) < .. < \alpha(z_5) < \alpha(z_3) < \alpha(z_1) \\ \alpha(z_2) < \alpha(z_4) < \alpha(z_6) < .. < \alpha(z_{k+1}) \end{cases} \quad \textit{if } k \textit{ even}$$

and:

$$\begin{cases} \alpha(z_{k+1}) < .. < \alpha(z_5) < \alpha(z_3) < \alpha(z_1) \\ \alpha(z_2) < \alpha(z_4) < \alpha(z_6) < .. < \alpha(z_k) \end{cases} \quad \textit{if } k \textit{ odd}$$

Since $z_1(z_2)$, it follows $\alpha(z_1) < \alpha(z_2)$ and consequently:

$$\begin{cases} \alpha(z_k) < .. < \alpha(z_3) < \alpha(z_1) < \alpha(z_2) < \alpha(z_4) < .. < \alpha(z_{k+1}) & \textit{if } k \textit{ even} \\ \alpha(z_{k+1}) < .. < \alpha(z_3) < \alpha(z_1) < \alpha(z_2) < \alpha(z_4) < .. < \alpha(z_k) & \textit{if } k \textit{ odd} \end{cases}$$

which is the thesis since $\alpha(z_k)$ and $\alpha(z_{k+1})$ are the rightmost and leftmost elements respectively of the lines above. ■

Corollary 23 *A conflict path whose length is k needs at least $|\alpha(z_{k+1}) - \alpha(z_k)| + 1$ columns to exist.*

Proof. The proof is easy and follows directly from the previous proposition. Vehicles z_{k+1} and z_k are the most distant from each other and all the others are settled in the columns between $\alpha(z_k)$ and $\alpha(z_{k+1})$. ■

2.3.3 C-types Conflict Paths and Number of Vehicles in a Graph

In this section, we establish a relation between the length k of a conflict path and the minimum number of columns (and, consequently, vehicles) needed

to guarantee the existence of such a path $z_1(z_2), z_2(z_3), ..., z_k(z_{k+1})$. Let's call $n_{\min}$ the minimum number of needed columns, it can be obtain as follow:

$$n_{\min} = \min_{S^k} |\alpha(z_{k+1}) - \alpha(z_k)| + 1$$

where $|\alpha(z_{k+1}) - \alpha(z_k)| + 1$ is the number of columns needed to a k-length conflict path to exist and S^k is the set of all permutations which contain a k length conflict path.

Proposition 24 *Given a C-conflict path $z_1(z_2), z_2(z_3), ..., z_k(z_{k+1})$, it holds that:*

$$|\alpha(z_{k+1}) - \alpha(z_k)| = 2[|\alpha(z_1) - \omega(z_1)| + \sum_{i=1}^{k-1} |\omega(z_i) - \omega(z_{i+1})|] \qquad (3)$$

Proof. WLG assume $\alpha(z_i) - \omega(z_i) < 0$ if i is odd and $\alpha(z_i) - \omega(z_i) > 0$ otherwise. Assume k odd, for the other case the proof is similar. In this case we have:

$$|\alpha(z_k) - \alpha(z_{k+1})| = \alpha(z_{k+1}) - \alpha(z_k)$$

Remembering that:

$$\alpha(z_{i+1}) - \alpha(z_i) = 2[\omega(z_i) - \alpha(z_i)]$$

we can write:

$$\begin{aligned} \alpha(z_{k+1}) - \alpha(z_k) &= 2[\omega(z_k) - \alpha(z_k)] \\ &= 2[\omega(z_k) - \alpha(z_k) + \alpha(z_{k-1}) - \alpha(z_{k-1}) + .. + \alpha(z_1) - \alpha(z_1)] \\ &= 2[(\omega(z_k) - \omega(z_{k-1})) + (\omega(z_{k-2}) - \omega(z_{k-1})) + .. + (\omega(z_1) - \alpha(z_1))] \end{aligned}$$

and the terms inside the brackets are positive since i is odd, $z_i \in R$ if i odd, $z_i \in L$ otherwise. ■

Remark 25 *We can immediately notice that the RHS of (3) is an increasing function in the sum of the distances of the final positions of consecutive vehicles and also in the number of horizontal steps of vehicle z_1. Let's consider the following observations:*

1. *Instead of minimizing* $|\alpha(z_{k+1}) - \alpha(z_k)|$*, we can proceed to minimize the RHS of (3).*

2. *The quantity* $|\alpha(z_1) - \omega(z_1)|$ *has a minimum and trivially it takes value 1. This means that* $|\alpha(z_1) - \alpha(z_2)| = 2$ *and so we can immediately say that* $n_{\min}$ *takes value 3 if the length of the C-path Conflicts is* $k = 2$*. In this case we have* $\alpha(z_2) = \alpha(z_1) + 2$ *or* $\alpha(z_2) = \alpha(z_1) - 2$*.*

Keeping in mind that the goal is to find the value of $n_{\min}$, in the following part we provide an equivalent problem to this one. We show that an optimal solution of this last problem must have certain characteristics (next two propositions) and then we show that such an optimal solution always exists. We can rewrite the problem of finding the value of $n_{\min}$ as follow. In order to get $n_{\min}$, we solve the problem consisting in the determination of k different destinations of vehicles and minimizing the quantity:

$$|\alpha(z_{k+1}) - \alpha(z_k)| = 2[|\alpha(z_1) - \omega(z_1)| + \sum_{i=1}^{z-1} |\omega(z_i) - \omega(z_{i+1})|]$$

In this way, the problem of finding the minimum number of columns that allows a k-length conflict path is equivalent to find an 1-1 assignment of k vehicles to k destinations for which is a minimum the quantity:

$$\sum_{i=1}^{z-1} |\omega(z_i) - \omega(z_{i+1})| \tag{4}$$

Notation 26 *We call* $\Delta_i = |\omega(z_i) - \omega(z_{i+1})|$*.*

Proposition 27 *In the solution that minimizes (4), the destinations* $\omega(z_i)$ *for* $i = 1, .., k-1$*, of the vehicles involved in a conflict path of length* k*, must be located in a subset of adjacent columns of the graph.*

Proof. For the proof, see reference [2]. ■

Notation 28 *Let's rename* $w_k = \sum_{i=1}^{k-1} |\omega(z_i) - \omega(z_{i+1})|$ *and* $w_k^* = \min_{S^k}(\sum_{i=1}^{k-1} |\omega(z_i) - \omega(z_{i+1})|)$.

Proposition 29 *For any values of k, it holds:* $w_k^* - w_{k-1}^* \geqslant 2$.

Proof. For the proof, see reference [2]. ■

Remark 30 *Consider* w_k *the value of (4) in correspondence to a feasible solution, not necessary the optimal one, when the length of the conflict path is k. As consequence of the previous proposition, if* $w_k - w_{k-1}^* = 2$ *we have that* $w_k = w_k^*$. *This means that if* $w_k - w_{k-1}^* = 2$, w_k *is an optimal solution of:*

$$\min_{S^k}(\sum_{i=1}^{k-1} |\Delta_i|)$$

Since $w_2^* = 1$, *for values of* $k \geqslant 2$, *we must have* $w_k^* = 2k - 3$. *Substituting it in (3) we obtain:*

$$|z_{k+1} - z_k| = 2(1 + 2k - 3) = 4(k - 1)$$

that gives the (minimum) highest distance between vehicles belonging to the conflict path as a function of its length.

Proposition 31 *A feasible solution such that* $w_k = w_{k-1}^* + 2 = 2k - 3$ *exists* $\forall k \geqslant 2$.

Proof. Consider a conflict path whose length is k, as usual let's call it $z_1(z_2), z_2(z_3), ..., z_k(z_{k+1})$. Assume $\alpha(z_i) - \omega(z_i) < 0$ if i is odd and $\alpha(z_i) - \omega(z_i) > 0$ otherwise. We distinguish the case in which k is even and the case in which k is odd. Note that in both cases, the exact destination of the vehicle z_{k+1}, which do not belong to the conflict path, is irrelevant, provided that either $\omega(z_{k+1}) > \omega(z_k)$ if k is even or $\omega(z_{k+1}) < \omega(z_k)$ if k is odd. When k is even, then, as a consequence of the existence of a conflict path, it must be:

$$\begin{aligned} \omega(z_1) &> \omega(z_2) \\ \omega(z_2) &< \omega(z_3) \\ &\dots \\ \omega(z_{k-1}) &> \omega(z_k) \end{aligned}$$

Consider the following permutation of destinations:

$$\omega(z_2), \omega(z_1), \omega(z_4), \omega(z_3), .., \omega(z_k), \omega(z_{k-1})$$

This way we have that $\Delta_i = 1$ if $i = 1, 3, 5, .., k-1$ and $\Delta_i = 3$ if $i = 2, 4, .., k$. Their sum is equal to $2k - 3$. When k is odd, then, as a consequence of the existence of a conflict path, it must be:

$$\begin{array}{rcl} \omega(z_1) & > & \omega(z_2) \\ \omega(z_2) & < & \omega(z_3) \\ & & ... \\ \omega(z_{k-1}) & < & \omega(z_k) \end{array}$$

Consider the following permutation of destinations:

$$\omega(z_2), \omega(z_1), \omega(z_4), \omega(z_3), .., \omega(z_{k-2}), \omega(z_{k-1}), \omega(z_k)$$

This way we have that $\Delta_i = 1$ if $i = 1, 3, 5, .., k-2$ and $\Delta_i = 3$ if $i = 2, 4, .., k-3$ and $\Delta_{k-1} = 2$. Their sum is equal to $2k - 3$. ■

Remark 32 *So, it is possible now to establish the relation between the minimum number of columns needed to the existence of a conflict and its length. Recalling that:*

$$n_{\min} = \min_{S^k} |\alpha(z_{k+1}) - \alpha(z_k)| + 1$$

with $|\alpha(z_1) - \alpha(z_2)| = 2$ *and* $\min_{S^k} |\alpha(z_{k+1}) - \alpha(z_k)| = 4(k-1)$ *for* $k \geqslant 2$, *we have:*

$$n_{\min} = \begin{cases} 4k - 3 & \text{if } k \geqslant 2 \\ 3 & \text{if } k = 1 \end{cases}$$

2.3.4 Number of Levels Ensuring Feasibility of any FQRP Instance with n Vehicles

We now determine the number of levels that ensure the existence of a solution to any possible permutation of vehicles destinations when the value of n is fixed. In order to do so, we firstly prove that, given n, the length of the longest C-type conflict path can be computed in advance. The following proposition provides the relation between n and such a length.

Proposition 33 *The length of the longest C-type conflict path in a graph $m \times n$, which we shall call $k_{\max}$, is:*

$$k_{\max} = \begin{cases} \lfloor (\frac{n-1}{4}) \rfloor + 1 & \text{if } n \geqslant 3 \\ 0 & \text{otherwise} \end{cases}$$

Proof. The proof follows immediately, considering the value of $n_{\min}$. If $n \leqslant 2$, no conflict path can exist because the smallest conflict path, having length $k = 1$ needs at least 3 columns to exist. If $n \geqslant 3$, we can obtain the value of $k_{\max}$ this way:

$$k = \frac{n+3}{4} = \frac{n-1+4}{4} = \frac{n-1}{4} + 1$$

since $k_{\max}$ must be integer, it becomes necessary to put the floor function. ■

If the maximum possible length of a conflict path having n vehicles is known, we can also determine the number of levels that ensure the existence of a solution to any instance of FQRP involving n vehicles. We call such a number m^*, as the following proposition states.

Proposition 34 *A solution to FQRP for any possible permutation of n destinations, can be found in a graph with m^* levels, where:*

$$m^* = k_{\max} + 2$$

Proof. For this proof, we proceed distinguishing some cases.

1. If $n = 1$, then $k_{\max} = 0$. There is only one vehicle that belongs to set S and obviously 2 levels are sufficient to route it to its destinations.

2. If $n = 2$, then $k_{\max} = 0$. There are two vehicles and we have 2 alternatives: or they both belong to set S (and in this case 2 lines are sufficient) either we have that the leftmost belongs to R and the rightmost belongs to L. In this case 2 lines are sufficient too because one vehicle performs its horizontal step in one level and the other vehicle performs its horizontal step in the other level.

3. If $n \geqslant 3$, it could happen that there is no one C-type conflict or there is one C-type conflict or there are more C-type conflicts, whose length is at most $k_{\max}$. In this case we have that $k_{\max} \geqslant 1$, and consequently

$m^* \geqslant 3$. In the absence of C-type conflicts, we can force vehicles belonging to R to move on the first level and vehicles belonging to L to move on the other level (or vice versa). Using this routing rule ensures to avoid any type of conflicts. Indeed, arc and B-type conflict occurrences are avoided due to the assignment to different levels of vehicles that must move in opposite directions. A-type conflict is avoided, because $m^* \geqslant 3$ and there is not a vehicle which must perform its horizontal steps on level m^*. Alternatively, it could be possible that there is only one C-type conflict, assume it has length $k_{\max}$. In order to take advantage of Remark, in the following part of the proof we will assume that each vehicle performs all its horizontal steps on one and only one level. This implies that $h_{z_i} > h_{z_{i+1}}$ and so $k_{\max}+1$ levels are sufficient to route all the vehicles. Relaxing this hypothesis, it follows that it could be used a number of levels lower than $k_{\max}+1$. In light of this, $k_{\max}+2$ levels are sufficient for sure. Now, consider an instance with 2 C-type conflicts $z'_1(z'_2), .., z'_{k_{\max}}(z'_{k_{\max}+1})$ and $z''_1(z''_2), .., z''_{k_{\max}}(z''_{k_{\max}+1})$, both with length $k_{\max}$. We have to distinguish 2 cases.

The first one occurs when z'_1 and z''_1 have to move horizontally on the same direction. The vehicles can be routed in this way: $z'_{k_{\max}+1}$ and $z''_{k_{\max}+1}$ perform all their horizontal steps on the first level, $z'_{k_{\max}}$ and $z''_{k_{\max}}$ perform their horizontal steps on the second level. The last group is composed by vehicles z'_1 and z''_1 that are forced to make all their horizontal steps on level $k_{\max}+1$, while all the other vehicles make all their horizontal moves on a level lower than $k_{\max}+1$. It follows that, since $m^* = k_{\max}+1$, vehicles z'_1 and z''_1 are the only vehicles forced to move horizontally on the last level and an A-type conflict may occur. On the contrary this cannot occur because z'_1 and z''_1 belong to a C-type conflict of length $k_{\max}$ and so:

$$\left|\alpha(z'_1) - \omega(z'_1)\right| = \left|\alpha(z''_1) - \omega(z''_1)\right| = 1 \tag{5}$$

It follows that we cannot have a vehicle j such that:

$$\alpha(z'_1) < \alpha(j) < \omega(j) < \omega(z'_1) \text{ or } \alpha(z''_1) < \alpha(j) < \omega(j) < \omega(z''_1) \tag{6}$$

because the relations above cannot hold at the same time.

The second case occurs when z'_1 and z''_1 have to move horizontally on opposite directions. In this case, vehicles can be routed in this way:

$z'_{k_{\max}+1}$ can move horizontally on the first level, $z''_{k_{\max}+1}$ and $z'_{k_{\max}}$ can move horizontally on the second level. As a consequence either z'_1 or z''_1 has to perform its horizontal steps on level $k_{\max}+2$. As said, they cannot be involved in A-type conflicts, since (5) and (6) should hold at the same time and this is not possible.

So, in general, in order to guarantee the existence of a solution to FQRP in any possible permutation of n destinations, at least $k_{\max}+2$ levels are needed. ■

Corollary 35 *If the following hypothesis hold:*

1. *Levels are classified in two sets: levels in which vehicles can move only toward right and vice versa, levels in which the only allowed steps are toward left;*
2. *All horizontal steps of a vehicle, if any, must be performed consecutively on the same level;*

then the minimum number of levels necessary and sufficient to solve any FQRP instance involving n vehicles, which we shall denote as m^*, is given by:

$$m^* = \begin{cases} 2 & \text{if } n = 2 \\ \lfloor (\frac{n-1}{4}) \rfloor + 3 & \text{if } n \geqslant 3 \end{cases}$$

Proof. The proof follows from the last 2 propositions. ■

Example 36 *Let's consider a generic instance of the FQRP, we give the value of* m^* *for* $n \in \{4, 5, 6, .., 29, 30\}$.

	Number of vehicles		*Needed lines*
•	$n = 4$	$\rightarrow$	$m^* = 3$
•	$n = 5$	$\rightarrow$	$m^* = 4$
•	$n = 6$	$\rightarrow$	$m^* = 4$
•	$n = 7$	$\rightarrow$	$m^* = 4$
•	$n = 8$	$\rightarrow$	$m^* = 4$
•	$n = 9$	$\rightarrow$	$m^* = 5$
•	$n = 10$	$\rightarrow$	$m^* = 5$
•	$n = 11$	$\rightarrow$	$m^* = 5$
•	$n = 12$	$\rightarrow$	$m^* = 5$
•	$n = 13$	$\rightarrow$	$m^* = 6$
•	$n = 14$	$\rightarrow$	$m^* = 6$
•	$n = 15$	$\rightarrow$	$m^* = 6$
•	$n = 16$	$\rightarrow$	$m^* = 6$
•	$n = 17$	$\rightarrow$	$m^* = 7$
•	$n = 18$	$\rightarrow$	$m^* = 7$
•	$n = 19$	$\rightarrow$	$m^* = 7$
•	$n = 20$	$\rightarrow$	$m^* = 7$
•	$n = 21$	$\rightarrow$	$m^* = 8$
•	$n = 22$	$\rightarrow$	$m^* = 8$
•	$n = 23$	$\rightarrow$	$m^* = 8$
•	$n = 24$	$\rightarrow$	$m^* = 8$
•	$n = 25$	$\rightarrow$	$m^* = 9$
•	$n = 26$	$\rightarrow$	$m^* = 9$
•	$n = 27$	$\rightarrow$	$m^* = 9$
•	$n = 28$	$\rightarrow$	$m^* = 9$
•	$n = 29$	$\rightarrow$	$m^* = 10$
•	$n = 30$	$\rightarrow$	$m^* = 10$

3 Theory's General Tools

3.1 Linear Programming Problems

It is useful to recall some generalities about linear programming problems (LP problems). A LP problem is a constrained optimization problem in which the function to be maximized (or minimized) is linear and the constraints are described by a fixed system of linear (in)equalities. If we write a system of linear (in)equalities in the compact form $Ax \sim b$ (where $\sim$ stands for $\leqslant$ or $\geqslant$ or $=$, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$), a LP problem has the form:

$$\min \text{ or } \max \quad c^T x \tag{7}$$

$$s.t. \quad Ax \sim b \tag{8}$$

$$x \in \mathbb{R}^n \tag{9}$$

where $c \in \mathbb{R}^n$. In model (7)-(9), $x_1, .., x_n$ are the variables, (7) is the objective function and (8)-(9) are the constraints. All the parameters $a_{i,j}, b_i, c_j$ with $i = 1, .., m$ and $j = 1, .., n$ are real numbers; the variables x can assume any value respecting the constraints of the model. We denote with a_i^T $i \in \{1, .., m\}$ the lines of matrix A.

Definition 37 *(Feasible Solution) A vector $x \in \mathbb{R}^n$ which respects all the constraints in (8)-(9) is called feasible solution of the problem. The set of all feasible solutions of a linear programming problem is called feasible region.*

Definition 38 *(Optimal Solution) A vector $\bar{x} \in \mathbb{R}^n$ is an optimal solution of the maximization problem (7)-(9) if it is a feasible solution and it holds that $c^T \bar{x} \geqslant c^T x$ for every vector x belonging to the feasible region (if the problem consists in minimizing the objective function, the condition becomes $c^T \bar{x} \leqslant c^T x$). The quantity $c^T \bar{x}$ is called optimal value.*

It is not always possible that a LP problem has got an optimal solution. LP problems can be divided into 3 sets:

1. Unfeasible problems: the feasible region is the empty set;
2. Unlimited problems: the objective function can be randomly maximized or minimized;

3. Problems with optimal solutions: there exists at least one optimal solution.

Solving a LP problem means establishing to which set the problem belongs and, in the third case, giving an optimal solution.
It could be useful to analyze the feasible region of a problem from a geometric point of view. Every equation and every inequality in (8) determines a specific region in $\mathbb{R}^n$: every equation identifies a hyperplane and every inequality identifies a close half-space. The feasible region is the intersection of those hyperplanes and close half-spaces.

Definition 39 *(Polyhedron) A set $P \subseteq \mathbb{R}^n$ is a polyhedron if it can be obtained from a finite intersection of hyperplanes and close half-spaces of $\mathbb{R}^n$.*

At this point, it is natural to show where an optimal solution lives, if it exists, with respect to a polyhedron. Thinking about the graphic method for the resolution of such problems, it is easy to convince us that an optimal solution, if it exists, is settled in one of the vertexes of the polyhedron. Let's introduce some concepts that help us to explain this point.

Definition 40 *(Convex Combination of two points) Given 2 points $x, y \in \mathbb{R}^n$, point $z \in \mathbb{R}^n$is a convex combination of x and y if there exists a real number $\lambda \in [0, 1]$ such that: $z = \lambda x + (1 - \lambda)y$.*

Definition 41 *(Strict convex combination of two points) Given 2 points $x, y \in \mathbb{R}^n$, point $z \in \mathbb{R}^n$is a strict convex combination of x and y if there exists a real number $\lambda \in (0, 1)$ such that: $z = \lambda x + (1 - \lambda)y$.*

Notice that, the strict convex combination does not include points x and y.

Definition 42 *(Vertex of a Polyhedron) Given a polyhedron $P \subseteq \mathbb{R}^n$ and a point $v \in P$, v is a vertex of P if it cannot be written as a strict convex combination of other 2 distinct points of P: $\nexists x, y \in P, \lambda \in (0, 1) : x \neq y, v = \lambda x + (1 - \lambda)y$.*

We can generalize the concept of convex combination as follow.

Definition 43 *(Generalized Convex Combination) Given k points $x_1, .., x_k$, point z is the convex combination of the previous ones if there exist k real numbers $\lambda_1, .., \lambda_k \geqslant 0$, such that: $\sum_{i=1}^{k} \lambda_i = 1$ and $z = \sum_{i=1}^{k} \lambda_i x_i$.*

Recall lastly an important result, known as Minkowsky-Weyl Representation/bounded case: "Given a bounded polyhedron $P \in \mathbb{R}^n$ and the set of its vertexes $V = \{v_1, ..v_k\}$, it follows that every point $x \in \mathbb{P}$ can be written as a convex combination of its vertexes". Now, we are ready to give a formal result.

Theorem 44 *Given a LP problem as in (7)-(9), if the polyhedron P representing the feasible region of this problem is non-empty and bounded, the problem has at least one optimal solution and an optimal solution is settled in a vertex of the polyhedron.*

Proof. Assume we are working with a minimization problem. The existence of an optimal solution follows from the fact that we are excluding the cases in which P is empty or unlimited. Let's call $V = \{v_1, ..v_k\}$ the set of the vertexes of P. Let's consider the value that the objective function assumes in the vertexes and let's call v^* the vertex in which the objective function is minimum:

$$c^T v^* \leqslant c^T v_i \quad \forall v_i \in V$$

For a generic $x \in P$ (using Minkowsky-Weyl), we can write:

$$c^T x = c^T \sum_{i=1}^{k} \lambda_i v_i = \sum_{i=1}^{k} \lambda_i c^T v_i \geqslant \sum_{i=1}^{k} \lambda_i c^T v^* = c^T v^* \sum_{i=1}^{k} \lambda_i = c^T v^*$$

Since x is generic, this means that v^* is an optimal solution and it is settled in a vertex. ■

This result is really important because it allows us to search for an optimal solution of any problem only in the vertexes of its polyhedron and not in any point of its feasible region. This is a very useful result because every polyhedron has got finitely many vertexes. In order to prove this, we need some more information about the constraints $Ax \sim b$.

Definition 45 *(Active Constraint) Given the problem in (7)-(9), if a vector $\bar{x} \in \mathbb{R}^n$ is such that $a_i^T \bar{x} = b_i$ for some $i \in \{1, .., m\}$, we say that the corresponding constraint is active in $\bar{x}$. Moreover we denote with $I(\bar{x})$ the set of the indexes of all active constraints:*

$$I(\bar{x}) = \left\{ i \in \{1, .., m\} : a_i^T \bar{x} = b_i \right\}$$

From now on, we say that two constraints are linearly independent if correspondent vectors a_i^T are linearly independent.

Theorem 46 *Given a polyhedron P and a point $\bar{x} \in P$, point $\bar{x}$ is a vertex of P if and only if there exist n rows a_i^T of matrix A with $i \in I(\bar{x})$ which are linearly independent.*

Proof. WLG assume that polyhedron P is defined only by equations or inequalities with verse $\geqslant$ (it is obvious that every inequality with verse $\leqslant$can be transformed into an equivalent one with verse $\geqslant$). Firstly, let's prove the necessary condition: if $\bar{x}$ is a vertex of P, then there exist n constraints active in $\bar{x}$ and linearly independent. Suppose ad absurdum that the number of linearly independent constraints that are active in $\bar{x}$ is $k < n$. This implies that there exists a vector $0 \neq d \in \mathbb{R}^n$ such that:

$$a_i^T d = 0 \quad \forall i \in I(\bar{x}) \tag{10}$$

For every $i \notin I(\bar{x})$, we have that:

$$a_i^T \bar{x} > b_i$$

so there exists $\epsilon > 0$ such that vectors:

$$\begin{aligned} y &= \bar{x} - \epsilon d \\ z &= \bar{x} + \epsilon d \end{aligned} \tag{11}$$

satisfy these conditions: $a_i^T y \geqslant b_i$, $a_i^T z \geqslant b_i$ for every $i \notin I(\bar{x})$. Moreover, from (11) follows that for $i \in I(\bar{x})$:

$$\begin{aligned} a_i^T y = a_i^T \bar{x} - \epsilon a_i^T d = b_i \\ a_i^T z = a_i^T \bar{x} + \epsilon a_i^T d = b_i \end{aligned}$$

which means that vectors y and z belong to P. Since:

$$\bar{x} = 1/2y + 1/2z$$

and y and z are both different from $\bar{x}$, it follows that $\bar{x}$ is not a vertex but this is a contradiction.
For the other verse, we proceed ad absurdum again. Suppose there exist n rows of A which are active in $\bar{x}$ and linearly independent, but $\bar{x}$ is not a vertex. If $\bar{x}$ is not a vertex, it follows that $\bar{x} \in P$ and there exist two vectors $y \in P$ and $z \in P$ different from $\bar{x}$ such that:

$$\bar{x} = \lambda y + (1-\lambda)z$$

We cannot have that $\exists i \in I(\bar{x})$ such that $a_i^T y > b_i$ or $a_i^T z > b_i$ otherwise we would obtain:

$$a_i^T \bar{x} = \lambda a_i^T y + (1-\lambda) a_i^T z > \lambda b_i + (1-\lambda) b_i$$

and this contradicts the fact that $i \in I(\bar{x})$. So we must have that $a_i^T y = b_i$ and $a_i^T z = b_i \quad \forall i \in I(\bar{x})$ but, this means that the system:

$$a_i^T x = b_i \quad \forall i \in I(\bar{x})$$

has more than one solution (that is $\bar{x}, y, z$). This contradicts the hypothesis because we must have only one solution. ■

Corollary 47 *Given a polyhedron $P = \{x \in \mathbb{R}^n : Ax \sim b\}$, if matrix $A \in \mathbb{R}^{m \times n}$ has a number of linearly independent rows smaller than n, then P has got no vertexes. In particular, if $m < n$, then P has got no vertexes.*

Proof. Obviously, since a vertex needs n linearly independent rows to be defined. ■

Corollary 48 *Given a polyhedron $P = \{x \in \mathbb{R}^n : Ax \sim b\}$ and a point $\bar{x} \in P$, point $\bar{x}$ is a vertex of P if and only if it is the unique solution of the system:*

$$a_i^T \bar{x} = b_i \quad i \in I(\bar{x})$$

Proof. The direct implication is trivial in light of the previous theorem. The vice versa follows observing that if the above linear system has got a unique solution, then the number of variables must coincide with the number of equations which are linearly independent. ■

Corollary 49 *Any polyhedron $P = \{x \in \mathbb{R}^n : Ax \sim b\}$ has at most finitely many vertexes.*

Proof. If $m < n$, the polyhedron has got no vertexes. If $m > n$, every vertex corresponds to a subset of n linearly independent rows of matrix A. Since A has at most $\binom{m}{n} = \frac{m!}{n!(m-n)!}$ distinct subsets of n rows, the polyhedron has got at most $\frac{m!}{n!(m-n)!}$ vertexes. ■

3.2 Integer Linear Programming Problems

Integer linear programming problems (Integer LP problems) are linear programming problems where the feasible region is constituted only by the integer vectors satisfying constraints (8)-(9). We remind that a vector is integer if all its components are integer numbers. Using the notation like before, a generic integer LP problem can be written as follow:

$$\min \text{ or } \max \ c^T x \tag{12}$$

$$s.t.\ Ax \sim b \tag{13}$$

$$x \in \mathbb{Z}^n \tag{14}$$

In this case, variables x are called integer variables. An important class of variables for these problems are the binary variables, that can assume only values 1 and 0.
It is very common to solve integer LP problems considering their continuous relaxation.

Definition 50 *(Continuous Relaxation) The continuous relaxation of an integer LP problem (12)-(14) is the same problem (12)-(13) with* $x \in \mathbb{R}^n$.

Remark 51 *Notice that if we are working with a binary variable* x_i*, constraint* $x_i \in \{0, 1\}$ *becomes* $x_i \in [0, 1]$.

The new problem obtained from the relaxation is a LP problem and can be resolved efficiently. However, the optimal value of the objective function of the continuous relaxation of a problem can be randomly distant from the optimal value of the objective function of the initial problem. Thus the continuous relaxation of a problem is not generally a good approximation of the problem.
Integer LP problems are very useful in all the situations in which the resources, represented by the variables, are indivisible: for example they can represent objects, people, etc..

3.3 Valid Cuts

In this section we work with the feasible regions of the problems that we have previously introduced. We can immediately notice that the feasible region of any LP problem is a polyhedron and also the feasible region of the continuous relaxation of any integer LP problem is a polyhedron. Keeping in mind the definition of convex combination of a set of points in $\mathbb{R}^n$, we can introduce a new object.

Definition 52 *Given any set $\Omega \subseteq \mathbb{R}^n$, the convex envelope of Ω, denoted with $conv(\Omega)$, is the set containing all the points that are convex combinations of the points of Ω.*

It holds that $conv(\Omega)$ is the smallest convex set containing Ω, and it is the intersection of all convex sets containing Ω.

Lemma 53 *Given $\Omega \subseteq \mathbb{R}^n$ and a vector $c \in \mathbb{R}^n$, the optimal values of the following problems coincide:*

$$\max\left\{c^T x : x \in \Omega\right\} = \max\left\{c^T x : x \in conv(\Omega)\right\}$$

Proof. Since the feasible region of the second problem contains the feasible region of the first problem, it follows that the optimal value of the second problem is always greater or equal to the optimal value of the first problem. For the other side, let's consider any feasible solution of the second problem $x \in conv(\Omega)$, let's show that there exists a feasible solution of the first problem $\bar{x} \in \Omega$ such that $c^T\bar{x} \geqslant c^T x$. Since $x \in conv(\Omega)$, it has this form:

$$x = \sum_{i=1}^{k} \lambda_i x_i$$

where $\lambda_1, .., \lambda_k \geqslant 0$ are real numbers such that: $\sum_{i=1}^{k} \lambda_i = 1$ and $x_1, .., x_k$ belong to Ω. It follows that there exist $i \in \{1, .., k\}$ such that $c^T x_i \geqslant c^T x$ otherwise we would obtain:

$$c^T x = \sum_{i=1}^{k} \lambda_i c^T x_i < \sum_{i=1}^{k} \lambda_i c^T x = c^T x$$

which is a contradiction. Since $x_i \in \Omega$, we can choose $x_i = \bar{x}$. ■

We can give now an important theorem.

Theorem 54 *(Meyer) Let Ω be the feasible region of an integer LP problem with rational parameters. Then $conv(\Omega)$ is a polyhedron.*

This theorem allows us to do the following consideration. Consider the integer LP problem:

$$\min \text{ or } \max \ c^T x \tag{15}$$

$$s.t.\ Ax \sim b \tag{16}$$

$$x \in \mathbb{Z}^n \tag{17}$$

where all the parameters are rational numbers. Let's call Ω the feasible region (16)-(17). Meyer's theorem guarantees that $conv(\Omega)$ is a polyhedron and so it is described by a system $Cx \sim d$. Using the previous lemma we have that problem (15)-(17) is equivalent to the following one:

$$\min \text{ or } \max \ c^T x \tag{18}$$

$$s.t.\ Cx \sim d \tag{19}$$

Basically, this means that every integer LP problem with rational parameters is equivalent to a LP problem. This information is very useful because LP problems can be solved efficiently. However, there are two observations to do. The previous lemma guarantees that the optimal values of the problems (15)-(17) and (18)-(19) are the same, but it does not guarantee that the optimal solutions are the same. The second problem could have more solutions than the first one and it could happen that an optimal solution of (18)-(19) does not belong to Ω. However if we solve this second problem using the Simplex method, we find a solution (if it exists) in a vertex of $conv(\Omega)$ (i.e. in a vertex of the polyhedron defined by $Cx \sim d$) and all the vertexes of $conv(\Omega)$ are contained in Ω.

Proposition 55 *Let $\Omega \subseteq \mathbb{R}^n$ and assume $conv(\Omega)$ be a polyhedron. Then the vertexes of $conv(\Omega)$ belong to Ω.*

Proof. Let x be a vertex of $conv(\Omega)$, so x cannot be written as a convex combination of points of $conv(\Omega)$ different from x. Since $x \in conv(\Omega)$, it can be written as convex combination of points $x_1, .., x_k \in \Omega$. It follows that every x_i coincides with x, so $x \in \Omega$. ■

The second observation is the following: If we want to transform problem (15)-(17) into problem (18)-(19), we must know the system $Cx \sim d$. In general this is really hard and it is possible that the system $Cx \sim d$ contains a huge number of constraints even if the system $Ax \sim b$ is very simple. This means that the correspondence between integer LP problems and LP problems is very difficult to apply.
In light of this, when we have to deal with an integer LP problem, we prefer to work with an equivalent LP problem or with its continuous relaxation. Since the first alternative is very difficult, we prefer to use another approach involving the continuous relaxation. In general, the feasible region of the continuous relaxation of a problem is much larger then the ideal formulation. It is reasonable to introduce new equations or inequalities to strenghten the formulation: these (in)equalities must be satisfied by all the integer solutions but should hopefully cut away some points from the feasible region of the continuous relaxation.

Definition 56 *(Valid Cut) Given an integer LP problem, we call Valid Cut any (in)equality that is satisfied by all the integer solutions of the problem and is not satisfied by at least one of the non-integer solutions of the continuous relaxation of the problem.*

Notice that if we call Ω the feasible region of an integer LP problem, a valid cut is satisfied by all the points of Ω and also by all the points of $conv(\Omega)$. From a geometric point of view, we can interpret such a cut like an (in)equality that cuts away some points that are settled out of the convex envelope of the feasible region of a problem.
In order to clarify which kind of (in)equalities we are searching for, we introduce a problem, called Knapsack problem, for which finding valid inequalities is very simple.
In the knapsack problem we are given n items of weight $a_1, .., a_n$ and profit $p_1, .., p_n$ respectively, and a bag (the knapsack) of maximum capacity β. We have to determine a subset of the n items that can be put in the bag without exceeding the maximum capacity, maximizing the total profit. In the following we assume that $a_1, .., a_n$ and β are integer numbers. If we define binary variables $x_1, .., x_n$, where $x_i = 1$ if and only if item i is selected, we can formulate the knapsack problem as the following integer linear programming problem:

$$\max \sum_{i=1}^{n} p_i x_i \tag{20}$$

$$s.t. \sum_{i=1}^{n} a_i x_i \leqslant \beta \tag{21}$$

$$0 \leqslant x_i \leqslant 1, \quad x_i \in \mathbb{Z} \quad i = 1, .., n \tag{22}$$

We call cover any subset of the n items that exceeds the knapsack capacity: in other words, a cover is a subset $C \subseteq \{1, .., n\}$ such that:

$$\sum_{i \in C} a_i \gneqq \beta$$

Since $a_1, .., a_n$ and β are all integer numbers, the above condition can be restated as follows:

$$\sum_{i \in C} a_i \geqslant \beta + 1$$

Given a cover C, since it is impossible to put all the elements of C in the knapsack, at most $|C| - 1$ of them can be selected. This implies that the following inequality is satisfied by all integer solutions:

$$\sum_{i \in C} x_i \leqslant |C| - 1$$

This inequality is called cover inequality. If all possible cover inequalities are added to the original formulation of the knapsack problem, we obtain a better formulation.
As we can see, the idea behind the formulation of cover inequalities for the knapsack problem is very simple and intuitive. However, finding a valid cut is difficult and it takes a lot of time in general.

4 FQRP Models

In this section, we describe three integer LP models that can resolve the FQRP. Moreover, we complete this chapter with the AMPL-codes. These three models have different characteristics and structures. The first model we are concerned with, is model A and it is strongly flow-based. For each vehicle $k \in P$, we establish if that vehicle moves horizontally or vertically in correspondence to a certain node of the grid. We describe the paths of the vehicles exactly in this way. The flow guarantees that the path of every vehicle is continuous and that it starts and finishes in the correct nodes. The representation of the flow needs several binary variables. The second model we describe is model B. In this model, the paths of the vehicles are described by giving the positions of their vertical moves. In other words, we just need to state if, at each node of the grid, a vehicle moves vertically or not. Binary variables are very useful and allow us to describe perfectly the event "in a certain node a vehicle moves or not". Finally, the third model is model C. In model C, we count the number of vertical moves associated to every column in which a vehicle has to pass. As already said, there is a set of column-indexes associated to every vehicle $k \in P$, which is $I(k)$. If we establish the number of vertical moves that vehicle k performs in correspondence to each column of $I(k)$, we are able to build the path associated to k. The objective function of the three models consists in minimizing the value of z, i.e., we want to find the minimum number of levels necessary for all the vehicles to complete all horizontal moves before reaching their final destinations. There is an important fact that we need to underline immediately. In model A and in model B, z has exactly the meaning of highest level in which a horizontal move takes place. Differently, in model C, the meaning of z is a bit different: it represents the number of vertical moves that a vehicle performs in order to reach the last level of the grid in which horizontal moves take place. It is clear that, even if z always play the same role, its value is different depending on whether the model is: in model A and B we count the horizontal levels needed, in model C we count the vertical steps among the levels needed.

Remark 57 *Denoting by z_1 the optimal value of the objective function of formulation A, by z_2 the optimal value of the objective function of formulation B and by z_3 the optimal value of the objective function of formulation C, it holds:*

$$\begin{cases} z_1 = z_2 \\ z_2 = z_3 + 1 \end{cases}$$

In fact, the latter formulation does not count the last vertical step to reach

the destination.

4.1 Model A

4.1.1 Description

VARIABLES In order to describe the path of the vehicles, we need some binary variables. For each $k \in R$ let's define:

$$x^v(k,i,j)$$

which is 1 if the edge joining nodes (i,j) and $(i,j+1)$ belongs to the shortest path of vehicle k (and is 0 otherwise); these variables are defined $\forall i \in I(k)$ and $\forall j \in J$;

$$x^h(k,i,j)$$

which is 1 if the edge joining nodes (i,j) and $(i+1,j)$ belongs to the shortest path of vehicle k (and is 0 otherwise); these variables are defined $\forall i : \alpha(k) \leq i \leq \omega(k) - 1$ and $\forall j \in J$.
For each $k \in L$ let's define:

$$y^v(k,i,j)$$

which is 1 if the edge joining nodes (i,j) and $(i,j+1)$ belongs to the shortest path of vehicle k (and is 0 otherwise); these variables are defined $\forall i \in I(k)$ and $\forall j \in J$;

$$y^h(k,i,j)$$

which is 1 if the edge joining nodes (i,j) and $(i-1,j)$ belongs to the shortest path of vehicle k (and is 0 otherwise); these variables are defined $\forall i : \omega(k) + 1 \leq i \leq \alpha(k)$ and $\forall j \in J$.

LINEAR MODEL

$$\min z \tag{23}$$

s.t.

$$
\begin{aligned}
&x^h(k,\alpha(k),1) + x^v(k,\alpha(k),1) = 1 && \forall k \in R \\
&x^v(k,\omega(k),m-1) = 1 && \forall k \in R \\
&x^v(k,i,j-1) + x^h(k,i-1,j) = x^v(k,i,j) + x^h(k,i,j) && \\
& && \forall k \in R, \forall i \in I(k), \forall j \in J
\end{aligned}
\tag{24}
$$

$$
\begin{aligned}
&x^h(k,\alpha(k)-1,j) = 0 && \forall k \in R, \forall j \in J \\
&x^h(k,\omega(k),j) = 0 && \forall k \in R, \forall j \in J \\
&x^v(k,i,0) = 0 && \forall k \in R, \forall i \in I(k)
\end{aligned}
\tag{25}
$$

$$
\begin{aligned}
&y^h(k,\alpha(k),1) + y^v(k,\alpha(k),1) = 1 && \forall k \in L \\
&y^v(k,\omega(k),m-1) = 1 && \forall k \in L \\
&y^v(k,i,j-1) + y^h(k,i+1,j) = y^v(k,i,j) + y^h(k,i,j) && \\
& && \forall k \in L, \forall i \in I(k), \forall j \in J
\end{aligned}
\tag{26}
$$

$$
\begin{aligned}
&y^h(k,\alpha(k)+1,j) = 0 && \forall k \in L, \forall j \in J \\
&y^v(k,i,0) = 0 && \forall k \in L, \forall i \in I(k)
\end{aligned}
\tag{27}
$$

$$
\begin{aligned}
x^v(k_1,i,j-1) + x^h(k_1,i-1,j) + y^v(k_2,i,j-1) + y^h(k_2,i+1,j) \leq 1 \\
i = c(k_1,k_2), \forall k_1 \in R, \forall k_2 \in L, \forall j \in J
\end{aligned}
\tag{28}
$$

$$
x^h(k_1,i,j) + y^h(k_2,i+1,j) \leq 1 \quad i = c(k_1,k_2), \forall k_1 \in R, \forall k_2 \in L, \forall j \in J
\tag{29}
$$

$$
\begin{aligned}
&z \geq j \cdot x^h(k,i,j) && \forall k \in R, \forall i : \alpha(k) \leq i \leq \omega(k)-1, \forall j \in J \\
&z \geq j \cdot y^h(k,i,j) && \forall k \in L, \forall i : \omega(k)+1 \leq i \leq \alpha(k), \forall j \in J
\end{aligned}
\tag{30}
$$

The variables of this model are the following:

- z — *objective function* — *integer, >=0*
- $x^v(k,i,j)$ — $\begin{cases} \forall k \in R \\ \forall i \in I(k) \\ \forall j \in J \end{cases}$ — *binary*
- $x^h(k,i,j)$ — $\begin{cases} \forall k \in R \\ \alpha(k) \leq i \leq \omega(k) - 1 \\ \forall j \in J \end{cases}$ — *binary*
- $y^v(k,i,j)$ — $\begin{cases} \forall k \in L \\ \forall i \in I(k) \\ \forall j \in J \end{cases}$ — *binary*
- $y^h(k,i,j)$ — $\begin{cases} \forall k \in L \\ \omega(k) + 1 \leq i \leq \alpha(k) \\ \forall j \in J \end{cases}$ — *binary*

The first constraint of the group (24) assures that each vehicle $k \in R$ moves either horizontally or vertically from its starting position. The second constraint says that the last move of the vehicle $k \in R$ (that is the move the vehicle performs to reach its final position) is a vertical move. The third constraint is called "flow balance" equation and it states that the number of edges entering in a node must coincide with the number of edges leaving from that node. We have to pay attention to some border conditions, which are those in (25), and they guarantee that the "flow balance" is respected also in the borders of the grid.
The constraints in (26) regard vehicles of L and their meaning is exactly the same of the constraints in (24). In (27) we have some border conditions of the flow, like in (25).
All the above constraints guarantee that, for each $k \in P$, the edges associated with variables that take value 1 provide a shortest path. Other constraints are necessary to prevent possible conflicts, i.e., situations where two different vehicles compete to reach the same node or to use the same edge at the same time. The constraints that prevent such conflicts are those of the groups (28) and (29) respectively.
Finally, the variable z is linked to the variables x^h and y^h through the (30).

4.1.2 Implementing Notes

We give an advice about the implementation of model A with software AMPL. This model strongly uses "flow balance" equations and lots of variables, many of them are needed only to fix the border's conditions of the

flow. Indeed, when we consider a vehicle $k \in R$, there is a natural set of binary variables associated to it. These variables are:

$$\begin{cases} x^h(k,i,j) \text{ with: } \alpha(k) \leqslant i \leqslant \omega(k) - 1, j \in J \\ x^v(k,i,j) \text{ with: } \alpha(k) \leqslant i \leqslant \omega(k), j \in J \end{cases}$$

However, we can notice that constraints (25) need also to use variables $x^v(k,i,0)\ \forall i \in I(k)$ and $x^h(k,\alpha(k)-1,j)\ \forall j \in J$, which are not present in the natural set. If we want the model to work, we have to fix this index mismatch. In my implementation I decided to create the sets of variables $x^v(k,i,0)\ \forall i \in I(k)$ and $x^h(k,\alpha(k)-1,j)\ \forall j \in J$ and fix them at value 0, paying attention to the fact that the "flow balance" equation has to hold also for the lower row of the grid. Indeed, the flow assumes value 1 for all the nodes of the path of vehicle k, starting from node $(\alpha(k),1)$ up to node$(\omega(k),m)$. This means that (in relation to the node $(\alpha(k),1)$) one between $x^v(k,\alpha(k),0)$ and $x^h(k,\alpha(k)-1,1)$ must be 1. I decided to impose:

$$x^h(k,\alpha(k)-1,1) = 1$$

We can do a similar reasoning for every vehicle $k \in L$. The natural set of variables associated to a generic $k \in L$ is:

$$\begin{cases} y^h(k,i,j) \text{ with: } \omega(k)+1 \leqslant i \leqslant \alpha(k), j \in J \\ y^v(k,i,j) \text{ with: } \omega(k) \leqslant i \leqslant \alpha(k), j \in J \end{cases}$$

We can notice that the constraints (27) use variables $y^v(k,i,0)\ \forall i \in I(k)$ and $y^h(k,\alpha(k)+1,j)\ \forall j \in J$. Also in this case, I created these variables and imposed them equal to 0, providing the "flow balance" equation continue to hold everywhere, especially in correspondence to the node $(\alpha(k),1)$. I imposed one between $y^v(k,\alpha(k),0)$ and $y^h(k,\alpha(k)+1,1)$ equal to 1:

$$y^h(k,\alpha(k)+1,1) = 1$$

In this way, the flow assumes value 1 in correspondence to the first node of every vehicle $k \in P$ and the implementation of model A runs correctly.

4.1.3 Code

For the code, see Appendix 1.

4.2 Model B

4.2.1 Description

VARIABLES Recall that we can denote the position of a vehicle through a couple of numbers (i, j) which represent respectively the column and the row of the vehicle. We will use the following variables:

$$v^k(i, j)$$

which take value 1 if vehicle $k \in P$ moves vertically at position (i, j) of the grid and $j \leqslant z$, (0 otherwise); they are defined for each
$k \in P$, $i \in I(k)$, $j \in J$.
Then we need the variables:

$$\delta_d(p, q, l)$$

which represent the difference of the number of vertical moves under level l and up to the median column, between vehicles $p \in R$ and $q \in L$; they are defined for each level $l = 2, .., m-1$ and each pair of conflicting vehicles $(p, q) \in C$.

$$\delta_u(p, q, l)$$

which represent the difference of the number of vertical moves from level l to level $m-1$ and after the median column, between vehicles $p \in R$ and $q \in L$. It is defined for each $l \in J$ and pair of conflicting vehicles $(p, q) \in C$.

LINEAR MODEL

$$\min z \tag{31}$$

s.t.

$$\sum_{i \in I(k)} \sum_{j=1}^{m-1} v^k(i, j) = z \quad \forall k \in P \tag{32}$$

$$\sum_{j=1}^{m-1} v^k(\omega(k), j) \geqslant 1 \quad \forall k \in P \tag{33}$$

$$\sum_{i \in I(k)} v^k(i, j) \leqslant 1 \quad \forall k \in P, j \in J \tag{34}$$

$$v^k(i, j) \leqslant \sum_{t=\min\{\alpha(k), i\}}^{\max\{\alpha(k), i\}} v^k(t, j-1) \quad \forall k \in P, i \in I(p), j = 2, .., m-1 \tag{35}$$

$$\delta_d(p, q, l) = \left| \sum_{i=\alpha(p)}^{c(p,q)} \sum_{j=1}^{l-1} v^p(i, j) - \sum_{i=\lceil \frac{\alpha(p)+\alpha(q)}{2} \rceil}^{\alpha(q)} \sum_{j=1}^{l-1} v^q(i, j) \right| \quad \forall (p, q) \in C, l = 2, .., m-1 \tag{36}$$

$$\delta_u(p, q, l) = \left| \sum_{i=\lceil \frac{\alpha(p)+\alpha(q)}{2} \rceil}^{\omega(p)} \sum_{j=l}^{m-1} v^p(i, j) - \sum_{i=\omega(q)}^{c(p,q)} \sum_{j=l}^{m-1} v^q(i, j) \right| \quad \forall (p, q) \in C, l \in J \tag{37}$$

$$\delta_d(p, q, l) + \delta_u(p, q, l) \geqslant 1 \quad \forall (p, q) \in C, l = 2, .., m-1 \tag{38}$$

$$\delta_u(p, q, 1) \geqslant 1 \quad \forall (p, q) \in C \tag{39}$$

$$v^p(i, j) \in \{0, 1\} \quad \forall p \in P, i \in I(p), j \in J$$

$$\delta_d(p, q, l) \in \mathbb{Z} \quad \forall (p, q) \in C, l = 2, .., m-1$$

$$\delta_u(p,q,l) \in \mathbb{Z} \quad \forall (p,q) \in C, l \in J$$

$$z \in \mathbb{Z}$$

Constraint (32) guarantees that the number of vertical moves that each vehicle performs in its range of columns must be exactly z. Constraint (33) states that each vehicle has to perform at least 1 vertical move in correspondence to the column in which its final position is settled. The third constraint, i.e. (34), is a limitation on the number of vertical moves that each vehicle $k \in P$ can perform in a generic position (i,j) with $i \in I(k)$ and $j \in J$: vehicle k can perform at most 1 vertical step there. Constraint (35) is needed in order to guarantee the continuity of every path. Given $k \in P$, the number of vertical moves that it performs in a generic position (i,j) with $i \in I(k)$ and $j \in J$ can be 1 only if the vehicle has the possibility to pass through that position, that is it has performed at least one vertical step in the previous level $j-1$ in the columns right before column i. Constraints (36) and (37) represent exactly the meaning of the variables $\delta_d(p,q,l)$ and $\delta_u(p,q,l)$ respectively. Notice that these constraints are not linear obviously and they need to be linearized. Constraint (38) and (39) guarantee that every couple $(p,q) \in C$ does not create a conflict. Indeed, we force the two vehicles to perform different many vertical steps until reaching their conflict column(s). Finally, the other constraints only state that the variables are binary or integer.

4.2.2 Linearization

We have to linearize constraints (36) and (37). Replacing everyone of these constraints with two inequalities of type $\delta \geqslant ...$ is wrong. In fact, a feasible solution would be setting z to 1, all variables v but $v^k(\omega(k),1)$ to 0, and all variables δ to 1: the linearizing constraints would be satisfied (strictly). An alternative way consists in using the technique of "Big M". Suppose $\delta = |a-b|$, we linearize this expression in this way:

$$\begin{aligned}
&\delta \leqslant a - b + Mw \\
&\delta \leqslant b - a + M(1-w) \\
&\delta \geqslant 0 \\
&w \in \{0,1\}
\end{aligned}$$

where $w = 1$ if $a < b$, 0 otherwise. Proceeding like this, constraint (36) can be replaced with:

$$\delta_d(p,q,l) = \sum_{i=\alpha(p)}^{c(p,q)} \sum_{j=1}^{l-1} v^p(i,j) - \sum_{i=\lceil \frac{\alpha(p)+\alpha(q)}{2} \rceil}^{\alpha(q)} \sum_{j=1}^{l-1} v^q(i,j) + m w_d(p,q,l)$$
$$\forall (p,q) \in C, l = 2,..,m-1$$

$$\delta_d(p,q,l) = -\sum_{i=\alpha(p)}^{c(p,q)} \sum_{j=1}^{l-1} v^p(i,j) + \sum_{i=\lceil \frac{\alpha(p)+\alpha(q)}{2} \rceil}^{\alpha(q)} \sum_{j=1}^{l-1} v^q(i,j) + m(1 - w_d(p,q,l))$$
$$\forall (p,q) \in C, l = 2,..,m-1$$

$$w_d(p,q,l) \in \{0,1\} \qquad \forall (p,q) \in C, l = 2,..,m-1$$

Similarly, constraint (37) can be replaced with:

$$\delta_u(p,q,l) = \sum_{i=\lceil \frac{\alpha(p)+\alpha(q)}{2} \rceil}^{\omega(p)} \sum_{j=l}^{m-1} v^p(i,j) - \sum_{i=\omega(q)}^{c(p,q)} \sum_{j=l}^{m-1} v^q(i,j) + m w_u(p,q,l)$$
$$\forall (p,q) \in C, l \in J$$

$$\delta_u(p,q,l) = -\sum_{i=\lceil \frac{\alpha(p)+\alpha(q)}{2} \rceil}^{\omega(p)} \sum_{j=l}^{m-1} v^p(i,j) + \sum_{i=\omega(q)}^{c(p,q)} \sum_{j=l}^{m-1} v^q(i,j) + m(1 - w_u(p,q,l))$$
$$\forall (p,q) \in C, l \in J$$

$$w_u(p,q,l) \in \{0,1\} \quad \forall (p,q) \in C, l \in J$$

Definitely, the variables we need for this model are:

- z *objective function* *integer, >=0*
- $v^k(i,j)$ $\begin{cases} \forall k \in P \\ \forall i \in I(k) \\ \forall j \in J \end{cases}$ *binary*
- $\delta_d(p,q,l)$ $\begin{cases} (p,q) \in C \\ l = 2,..,m-1 \end{cases}$ *integer*
- $\delta_u(p,q,l)$ $\begin{cases} (p,q) \in C \\ \forall l \in J \end{cases}$ *integer*
- $w_d(p,q,l)$ $\begin{cases} (p,q) \in C \\ l = 2.,,.m-1 \end{cases}$ *binary*
- $w_u(p,q,l)$ $\begin{cases} (p,q) \in C \\ \forall l \in J \end{cases}$ *binary*

4.2.3 Code

For the code, see Appendix 1.

4.3 Model C

4.3.1 Description

VARIABLES We need the following integer variables:

$$v^k(i)$$

which represents the number of vertical movement that a vehicle $k \in P$ performs in correspondence to column i; these variables are defined
$\forall k \in P$ and $\forall i \in I(k)$.
Given a pair of vehicles $(p,q) \in C$, we also need the following binary variables:

$$w(p,q)$$

that take value one (resp. zero) if vehicle p crosses the conflicting column(s) at a strictly higher (resp. lower) level than vehicle q.

LINEAR MODEL

$\min z$

(40)

$s.t.$

$$\sum_{i \in I(k)} v^k(i) = z \quad \forall k \in P \tag{41}$$

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) - 1 + mw(p,q) \quad \forall (p,q) \in C_{odd} \tag{42}$$

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant \sum_{i=\alpha(p)}^{c(p,q)} v^p(i) - 1 + m(1 - w(p,q)) \quad \forall (p,q) \in C_{odd} \tag{43}$$

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) - 1 + mw(p,q) \quad \forall (p,q) \in C_{even} \tag{44}$$

$$\sum_{i=c(p,q)}^{\alpha(q)} v^q(i) \leqslant \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) - 1 + m(1 - w(p,q)) \quad \forall (p,q) \in C_{even} \tag{45}$$

$z \in \mathbb{Z}$

$v^k(i) \in \mathbb{Z}_+ \quad \forall k \in P, \forall i \in I(k)$

$w(p,q) \in \{0,1\} \quad \forall (p,q) \in C$

The variables of this last model are:

- z *objective function* *integer, >=0*
- $v^k(i)$ $\begin{cases} \forall k \in P \\ \forall i \in I(k) \end{cases}$ *integer*
- $w(p,q)$ $(p,q) \in C$ *binary*

The first equation (40) is the objective function and consists in minimizing z. The second equation (41) states that for every vehicle $p \in P$, the number of vertical steps that it performs in its range of columns should be exactly z. The constraints (42) and (43) consider an arc-conflict between vehicles $p \in R$ and $q \in L$. Notice that the sums at the first member of these two constraints represent the maximum level reached by vehicle p (resp. q) at column $\lfloor c(p,q) \rfloor$ (resp. $\lceil c(p,q) \rceil$), that is the level at which vehicle p (resp. q) crosses the arc between $\lfloor c(p,q) \rfloor$ and $\lceil c(p,q) \rceil$ (resp. $\lceil c(p,q) \rceil$ and $\lfloor c(p,q) \rfloor$). These constraints state that the level in which p and q cross the conflicting columns is not the same, in other words we are stating that arc-conflicts are not allowed. Similarly, constraints (44) and (45) consider a node-conflict between vehicles $p \in R$ and $q \in L$. The constraints state that the levels in which vehicle p and q cross the conflict-column must differ at least by one unit if we want to avoid a node-conflict.

4.3.2 Code

For the code, see Appendix 1.

Remark 58 *It is very important to underline that all these models always admit one optimal solution (Theorem 44). Indeed, the polyhedron representing their feasible region is non-empty and bounded. We guarantee the existence of a solution choosing $m \geqslant m^*$, that is using a grid with a number of levels greater or equal then the minimum number of levels m^* that ensures the existence of a solution for any instance of FQRP given the number of vehicles n. Moreover, the polyhedron is bounded because it is contained in the hypersphere (having dimension equal to the total number of variables) of radius m^*.*

5 Computer Science Devices

5.1 Software AMPL and Connection to LABNUM

Making several examples has been fundamental for the realization of this work. Especially at the beginning, it was useful to see concretely some solutions and some variables' values. We have tested several instances of FQRP, solving model C with AMPL (initially we used version CPLEX 12.8.0.0, which is the basic one). Since we will provide many AMPL-outputs and AMPL's front is not universal, we briefly describe how to read such an output. For instance, consider the output given in Figure 1.

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 1
44 dual simplex iterations (0 in phase I)
z = 1

v [*,*]
:   1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 :=
4   .  .  .  1  0  0  .  .  .  .   .   .   .   .   .   .   .   .   .
6   .  .  .  0  0  1  .  .  .  .   .   .   .   .   .   .   .   .   .
9   .  0  1  0  0  0  0  0  0  .   .   .   .   .   .   .   .   .   .
...
;

w [*,*]
:    6      8      9      11     14     15     18    19     :=
1    .      .    0.25   0.25   0.25    .      .     .
4   0.25   0.25   .      .      .      .      .     .
...
;
```

Figure 1: Example of AMPL's output

The first line is a prompt-line: with command "include" we ask AMPL to solve a specific instance. The output gives us several information about the instance. Firstly, we see the value of the objective function, indicated as "objective .." or "$z = ..$". Secondly, we can be interested in the value of variables $v^p(i)$, for some $p \in P$. We read it in matrix $v[*,*]$: every line is associated to a particular vehicle $p \in P$ and the columns represent orderly the columns of the grid. If vehicle p performs a vertical step in column i we read 1 in position $v[p,i]$, if vehicle p performs no one vertical step in column i we read 0 in position $v[p,i]$, if column i does not belong to $I(p)$ we read a $\cdot$ in position $v[p,i]$. Sometimes, it is possible that AMPL prints $v[*,*]^T$ (T stands for transpose) instead of $v[*.*]$. Thirdly, we can be interested in

the values of variables $w(p,q)$ for some $(p,q) \in C$. We read these values in matrix $w[*,*]$: every line contains a vehicle of R and the columns contain vehicles of L. If $(p,q) \in C$, we find the value of $w(p,q)$ in position $w[p,q]$. If (p,q) does not belong to C, we find a dot $\cdot$ in position $w[p,q]$. Sometimes, it is possible that AMPL prints $w[*,*]^T$.
AMPL's version CPLEX 12.8.0.0 is a basic version, that is it can solve only limited problems with 500 variables and 500 constraints at most. We used it to solve small and medium instances, containing from 4 to 20 vehicles. However, it was possible that some instances containing 20 vehicles were too large to be solved. In order to test greater instances, it became necessary to dispose of a stronger version of AMPL. For this purpose, we connected with the computer laboratories of Torre Archimede and particularly to the Labnum, which contains ILOG AMPL, a stronger version of this software.
The procedure occurs from Command Line and is quite easy (we describe it for a Windows user). It is sufficient to open 2 command lines and give the following commands in the order in which they are written (see Figure 2).

1^ COMMAND LINE:

1) ssh gbortolu@sshtorre.math.unipd.it
2) ssh labnum01

N.B: Connection with the laboratory Labnum.

2^ COMMAND LINE:

3) scp namefile.mod gbortolu@sshtorre.unipd.it:~
4) scp namefile.dat gbortolu@sshtorre.unipd.it:~
5) scp namefile.run gbortolu@sshtorre.unipd.it:~

N.B: Upload files in directory "home" of Labnum.

1^ COMMAND LINE:

6) ls (optional)
7) ampl
8) include namefile.run;

N.B: After checking the content of directory "home", open AMPL and solve the problem.

Figure 2: Commands

This way, we can solve instances of any dimension. Every time we want to change an instance, it is sufficient to modify file ".dat" and upload the new file like before.

5.2 Software PORTA 1.4.1

In this part, we focus our attention on the use of software PORTA. The name PORTA is an abbreviation for POlyhedron Representation Transformation Algorithm. Indeed, PORTA is a collection of routines for analyzing polyhedra. The polyhedra are either given as the convex envelope of a set of points, or as a system of linear equations and inequalities. The software is always able to provide one of the two representations, given the other.
We have used software PORTA this way: given an instance of FQRP and established the set of points representing all its feasible solutions, we asked PORTA to transform the set of points in a set of (in)equalities. The goal was using the (in)equalities provided by the software as hints in order to write valid cuts for model C. We now describe how we have worked with PORTA.
First of all, we translated an instance of the problem into PORTA's language: we renamed all variables $v^k(i)$ $k \in P, i \in I(k)$ with names x_i orderly. We created a file with the extension *.poi* with the points/ feasible solutions of the instance. Then, using the command *traf* from the command lines, the software created another file *.ieq* with the (in)equalities representing the polyhedron of the instance.

Example 59 *Consider an instance with 4 vehicles, assume* $\alpha = (1, 3, 4, 2)$ *and* $\omega = (3, 1, 2, 4)$. *Vehicles 1 and 2 can move in columns* $\{1, 2, 3\}$ *and vehicles 3 and 4 can move in columns* $\{2, 3, 4\}$. *In each one of these columns a vehicle can perform a certain number of vertical moves, which can be 0 or at most* m^* *(remembering that* m^* *is the number of needed lines that ensures the existence of a feasible solution and* $m^* = 3$ *for this instance) Instead of writing all the variables* $v^k(i)$ *for* $k \in P, i \in I(k)$ *assuming values in* $\{0, .., m^*\}$, *we make the variables* v *assume only values only in* $\{0, 1\}$ *because we have previously resolved this instance of the problem and we have found that an optimal solution exists with* $m = 1$. *Obviously the polyhedron associated changes depending on the points that we assume to be set of all feasible points. However we only want to explain how PORTA works and so the polyhedral configuration does not matter. We renamed the variables* v *as follow:*

- *Vehicle 1* $I(1) = \{1, 2, 3\}$ $\begin{cases} x_1 = v^1(1) \\ x_2 = v^1(2) \\ x_3 = v^1(3) \end{cases}$
- *Vehicle 2* $I(2) = \{1, 2, 3\}$ $\begin{cases} x_4 = v^2(1) \\ x_5 = v^2(2) \\ x_6 = v^2(3) \end{cases}$
- *Vehicle 3* $I(3) = \{2, 3, 4\}$ $\begin{cases} x_7 = v^3(2) \\ x_8 = v^3(3) \\ x_9 = v^3(4) \end{cases}$
- *Vehicle 4* $I(4) = \{2, 3, 4\}$ $\begin{cases} x_{10} = v^4(2) \\ x_{11} = v^4(3) \\ x_{12} = v^4(4) \end{cases}$

Using these new variables, a feasible solution is the following: $(1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0)$ *which means that vehicle 1 can move 1 time vertically in column 1, vehicle 2 can move 1 time vertically in column 1, vehicle 3 can move 1 time vertically in column 2 and vehicle 4 can move 1 time vertically in column 2. The file .poi has this form (see Figure 3):*

```
DIM = 12

CONV_SECTION
 1 0 0 1 0 0 1 0 0 1 0 0
 0 0 1 0 0 1 0 0 1 0 0 1

END
DIMENSION OF THE POLYHEDRON : 12
```

Figure 3: file of type "poi"

So the polyhedron represented by the convex hull of these points is transformed into the same polyhedron represented now by a set of (in)equalities. The output is the following (see figure 4):

```
DIM = 12

VALID
0 0 1 0 0 1 0 0 1 0 0 1

INEQUALITIES_SECTION
( 1)   -x2                      == 0
( 2)        -x5                 == 0
( 3)             -x8            == 0
( 4)                  -x11      == 0
( 5)               +x9      -x12 == 0
( 6)            +x7    -x10      == 0
( 7)          +x6    -x9         == 0
( 8)       +x4    -x7            == 0
( 9)     +x3    -x6              == 0
( 10) +x1    -x4                 == 0
( 11)                +x10   +x12 == 1

( 1) -x12 <= 0
( 2) +x12 <= 1

END
```

Figure 4: file of type "ieq"

As already said, the goal was finding one or more (in)equalities that could suggest ideas for valid cuts of the problem. Looking at the (in)equalities provided by PORTA for every single example, we tried to deduce general rules which could hold in general. In other words, we tried to generalize the constraints of a generic instance, in order to obtain a constraint valid for every problem.
This process, that apparently is easy, turned out to be very difficult for different reasons. We started considering simple examples like above, that is instances with only 4 or 5 vehicles. It was very easy to find all the feasible solutions of such an instance and we proceeded "by hand". In these cases the convex hull was composed only of few points (4 or 5 at most) but, the negative fact was that the (in)equalities provided by PORTA where trivial and did not suggest any interesting ideas.

Recalling the previous example, notice that most of the constraints are equations of type: $x_i = 0$ or $x_i + x_j = 0$.

Later we started using larger instances, composed of 10 vehicles, hoping that we could obtain stronger information. Working with these instances, another problem arose. Indeed, it is difficult to write all the feasible solutions

of huge instances but, this is a necessary request if we want PORTA to work correctly. However writing all the feasible solutions is not impossible and we provide a possible way to follow in order not to forget any point.

Criterion 60 *1. Consider the possible values of* x_1*:* $x_1 \in \{a', b', c'..\}$*;*

2. *Fix* x_1 *at the first value:* $x_1 = a'$*;*
3. *Now consider* x_2 *and all its possible values:* $x_2 \in \{a'', b'', c''...\}$*;*
4. *Consider only the values of* x_2 *that allow a feasible pattern with* x_1 *(for example* (a', b'') *and* (a', c'')*);*
5. *Consider the values of* x_3*:* $x_3 \in \{a''', b''', c'''...\}$ *and for each one of the previous couples establish which values of* x_3 *gives a feasible solution (for example* (a', b'', a''')*);*
6. *Do the same for* x_4*;*
7. *Start from the first point with the second value of* x_1*.*

Written all the needed points, we could obtain PORTA's results. Even if we was able to manage with huge instances, the interpretation of these results revealed to be hard because the number of variables was high (about 40 variables). Equations and inequalities provided for such examples were not trivial like above and the huge number of involved variables did not allow us to find an idea for a valid inequality. Using the software this way, we did not find any significative result.
Later in time we tried to use PORTA for a different purpose. After finding some valid inequalities for model C, we tried using PORTA in order to understand if our inequalities were, afterwards, the same that PORTA suggested. However, we did not find any commonality.

6 New Valid Inequalities

We started the research of new valid inequalities for model C considering small instances of the problem, looking both at the integer solution and also at the relaxed solution (provided by AMPL). For every test, we tried to understand if the pattern associated to the relaxed solution was feasible or not and, in the case it was not, we tried to understand the reason why that pattern cannot go right. Putting together the observations obtained from several cases and generalizing them, we obtained the following inequalities. After making sure of the fact that they were really valid cuts, we added them to the model and proceeded with the trial, continuing to search the reason why the continuous relaxation still gave non-feasible solutions. In the end, we have found 7 sets of valid inequalities. In the following part we suggest them. They are written in this work in the same order in which they have been discovered. As we already said, we add these new inequalities to model C and, every time we add an inequality, we keep all the previous ones in the model. In this way, we test if the most recent inequalities cut other points with respect to the previous ones.

6.1 First Set of Inequalities

The first set of inequalities we provide consists in a relation between the value of the objective function and the value of $w(p,q)$ $\forall (p,q) \in C$.

Theorem 61 *Given any instance of FQRP, for any feasible solution of formulation C, it holds that:*

$$z \geqslant w(p,q) \quad \forall (p,q) \in C \tag{46}$$

Proof. This inequality follows from constraints (43) and (45) respectively for odd and even conflicts. Obviously, if $w(p,q) = 0$ the inequality is trivial for any type of conflict. Consider any couple $(p,q) \in C_{odd}$ such that $w(p,q) = 1$ and let's prove the inequality holds. Recall constraint (43) states that:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant \sum_{i=\alpha(p)}^{c(p,q)} v^p(i) - 1 + m(1 - w(p,q))$$

Since $w(p,q) = 1$, in relation to this pair the constraint becomes:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant \sum_{i=\alpha(p)}^{c(p,q)} v^p(i) - 1$$

It can be rewritten as follow:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \geqslant \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) + 1$$

and we can limit (from the top) the first sum with z and we can limit (from the bottom) the second sum with 0. This way we obtain:

$$z \geqslant \sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \geqslant \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) + 1 \geqslant 0 + 1$$

but $w(p,q) = 1$ and the pair we are using is generic. So we have:

$$z \geqslant w(p,q) \quad \forall (p,q) \in C_{odd}$$

Now consider any couple $(p,q) \in C_{even}$ such that $w(p,q) = 1$ and let's prove the inequality for an even conflict. Recall constraint (45), it states that:

$$\sum_{i=c(p,q)}^{\alpha(q)} v^q(i) \leqslant \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) - 1 + m(1 - w(p,q))$$

Since $w(p,q) = 1$, it becomes:

$$\sum_{i=c(p,q)}^{\alpha(q)} v^q(i) \leqslant \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) - 1$$

It can be rewritten as follow:

$$\sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) \geqslant \sum_{i=c(p,q)}^{\alpha(q)} v^q(i) + 1$$

and limiting the sums like above we obtain:

$$z \geqslant \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) \geqslant \sum_{i=c(p,q)}^{\alpha(q)} v^q(i) + 1 \geqslant 1$$

Recalling that $w(p,q) = 1$ and the fact that we are using a generic pair, it holds:

$$z \geqslant w(p,q) \quad \forall (p,q) \in C_{even}$$

■

Example 62 *Let's consider an instance of the problem with 5 vehicles and assume* $\alpha = (1,2,3,4,5)$ *and* $\omega = (2,1,3,5,4)$. *When we solve the continuous relaxation, an optimal solution is given by* $z = 0$ *with several variables* w *that assume values strictly greater than 0, for example* $w(1,2) = 0.25$ *(see Figure 5). Adding the new constraint (46), obviously this solution becomes unfeasible because:*

$$z = 0 \ngeqslant w(1,2) = 0.25$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 0
2 dual simplex iterations (0 in phase I)
z = 0

w :=
1 2   0.25
4 5   0.25
;
```

Figure 5: AMPL output example 62

Example 63 *Let's consider an instance of the problem with 15 vehicles and assume* $\alpha = (1,2,3,..,14,15)$ *and* $\omega = (2,5,9,10,3,4,14,7,12,8,15,1,6,11,13)$. *When we solve the continuous relaxation, an optimal solution is given by* $z = 0$ *with several variables* w *that assume values strictly greater than 0, for example* $w(3,5) = 0.25$ *(see Figure 6). Adding the new constraint, obviously this solution becomes unfeasible because:*

$$z = 0 \ngeqslant w(3,5) = 0.25$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 0
22 dual simplex iterations (0 in phase I)
z = 0

w [*,*] (tr)
:     2      3      4      7      9      11     :=
5    0.25   0.25   0.25    .      .      .
6    0.25   0.25   0.25    .      .      .
8     .      .      .     0.25    .      .
10    .      .      .     0.25   0.25    .
12    .     0.25   0.25   0.25   0.25   0.25
13    .     0.25   0.25   0.25   0.25   0.25
14    .      .      .      .     0.25   0.25
15    .      .      .      .      .     0.25
;
```

Figure 6: AMPL output example 63

Example 64 *Let's consider an instance of the problem with 20 vehicles and assume* $\alpha = (1, 2, 3, .., 14, 15)$ *and*
$\omega = (2, 10, 3, 5, 20, 4, 14, 17, 19, 18, 15, 16, 6, 11, 8, 9, 7, 12, 1, 13)$*. An optimal solution of the continuous relaxation is given by* $z = 0$ *and several variables* w *assume values strictly greater than 0, for example* $w(2, 13) = 0.25$ *(see Figure 7). Adding the new constraint, this solution becomes unfeasible:*

$$z = 0 \not\geq w(2, 13) = 0.25$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 0
61 dual simplex iterations (0 in phase I)
z = 0

w [*,*]
:     6      13     14     15     16     17     18     19     20     :=
2    0.25   0.25    .     0.25   0.25   0.25    .      .      .
4    0.25    .      .      .      .      .      .      .      .
5    0.25   0.25    .     0.25   0.25   0.25    .     0.25    .
7     .     0.25    .     0.25   0.25   0.25   0.25   0.25   0.25
8     .     0.25   0.25   0.25   0.25   0.25   0.25   0.25   0.25
9     .     0.25   0.25   0.25   0.25   0.25   0.25   0.25   0.25
10    .     0.25   0.25   0.25   0.25   0.25   0.25   0.25   0.25
11    .     0.25   0.25   0.25   0.25   0.25   0.25   0.25    .
12    .     0.25   0.25   0.25   0.25   0.25   0.25   0.25   0.25
;
```

Figure 7: AMPL output example 64

6.2 Second Set of Inequalities

The second set of valid inequalities that we suggest is very simple and it guarantees that, given an instance with conflicts, the value of the objective function is at least 1. The new constraint uses a new parameter δ that assumes value 1 if the instance contains conflicts, 0 otherwise.

Theorem 65 *Given any instance of FQRP, for any feasible solution of formulation C, it holds that:*

$$z \geqslant \delta$$

where $\delta = 1$ *if* $C \neq \varnothing$ *and* $\delta = 0$ *if* $C = \varnothing$.

Notice that, if the instance does not contain any conflict, the value of the parameter δ is 0 and the inequality becomes trivial. In a conflict-free instance, all vehicles belong to S and, obviously, the optimal value of the objective function must be 0.

Proof. Let's distinguish 2 cases: if $\delta = 0$, the constraint is trivial. If $\delta = 1$, the constraint states that the value of the objective function is at least 1. In this situation, there is at least 1 pair of vehicles that create a conflict. Since they cannot cross their conflict column(s) in the same level, it must be true that one of them crosses their conflict column(s) at least at level 0 and the other one crosses their conflict column(s) at least at level 1. Definitely, one of them has to perform at least one vertical move. This means exactly that $z \geqslant 1 = \delta$. ■

Example 66 *Consider again example 63. This instance contains several conflicts, for example* $(2, 5) \in C_{odd}$ *and* $(7, 13) \in C_{even}$ *and so* $\delta = 1$ *by definition. The solution provided by the continuous relaxation after adding to the model only the first new inequality gives* $z = 0.2$ *(see Figure 8). Adding the second inequality, we require that* $z \geqslant \delta$ *and so the above solution becomes unfeasible:*

$$z = 0.2 \ngeqslant \delta = 1$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 0.2
42 dual simplex iterations (0 in phase I)
z = 0.2
```

Figure 8: AMPL output example 66

Example 67 *Let's consider another instance of the problem with 15 vehicles and assume* $\alpha = (1, 2, 3, .., 14, 15)$ *and*
$\omega = (6, 9, 12, 3, 2, 4, 11, 10, 5, 1, 7, 13, 8, 15, 14)$. *This instance contains several conflicts, for example* $(7, 10) \in C_{odd}$ *and* $(3, 11) \in C_{even}$ *and so* $\delta = 1$. *Solving the continuous relaxation, an optimal solution is given by* $z = 0.2$ *(see Figure 9). Adding the second inequality, the previous solution becomes unfeasible because:*

$$z = 0.2 \ngeq \delta = 1$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 0.2
42 dual simplex iterations (0 in phase I)
z = 0.2
```

Figure 9: AMPL output example 67

Remark 68 *These two inequalities are needed essentially for one specific reason. When we started solving the continuous relaxation of certain instances, we noticed that the solution given by* $z = 0$ *was always an optimal solution. According to constraint (41),* $z = 0$ *implies that* $v^k(i) = 0$ $\forall k \in P, \forall i \in I(k)$. *The pattern associated to this solution cannot always be accepted: indeed, it is sufficient that an instance contains one conflict and z cannot be 0, as we have just seen. Consequently, with these inequalities we ensure that the solution with* $z = 0$ *is cut away from the feasible region of the continuous relaxation of the instances which are not conflict-free.*

6.3 Third Set of Inequalities

The third set of inequalities regards all the couples of vehicles that belong to C_{even} and it consists in a limitation on the number of vertical moves they can perform in their conflict column. Let $(p, q) \in C_{even}$ be a pair of vehicles that could create an even conflict. Recall that in this case $c(p, q)$ is the column:

$$c(p, q) = [\alpha(p) + \alpha(q)]/2$$

Theorem 69 *Given any instance of FQRP, for any feasible solution of formulation C, it holds that:*

$$v^p(c(p, q)) + v^q(c(p, q)) \leqslant z - 1 \quad \forall (p, q) \in C_{even} \tag{47}$$

Proof. Let's suppose ad absurdum that:

$$v^p(c(p, q)) + v^q(c(p, q)) \geqslant z$$

and let's consider two different cases.

1. It holds that:

$$v^p(c(p, q)) + v^q(c(p, q)) = z$$

Column $c(p, q)$ contains z edges and, in this case, they are all used by vehicles p and q. Notice that it is not possible that an edge of $c(p, q)$ used by vehicle p is also used by vehicle q or vice versa. [Suppose vehicle p uses an edge of $c(p, q)$ starting from a certain node M in column $c(p, q)$. If vehicle q also uses this edge, it means that q at some point uses the node M. However, column $c(p, q)$, by definition, is exactly in the middle of columns $\alpha(p)$ and $\alpha(q)$ so vehicles p and q need the same time to reach the generic node M. In this way, p and q would create a node-conflict in correspondence to M, but this is impossible because we cannot allow node-conflicts]. At this point, we have that the paths of p and q cannot share any edge of column $c(p, q)$ but, all the edges of column $c(p, q)$ are used by exactly one of them. This is impossible because a node-conflict would arise. [Let's suppose WLG p is the first vehicle that reaches column $c(p, q)$ (both p and q must cross column $c(p, q)$ at some point by definition of C_{even} pairs), this means that there exists a node M' in column $c(p, q)$ such that the

portion of path that p performs in column $c(p,q)$ ends in $M\prime$ and the portion of path that q performs in column $c(p,q)$ starts in $M\prime$. Like before, p and q need the same time to reach $M\prime$ and they would create a node-conflict there]. Consequently, the previous assumption brings always to conflicts so it cannot be true. This means that at least one edge of column $c(p,q)$ cannot be used by any vehicle of the pair and it is true that:

$$v^p(c(p,q)) + v^q(c(p,q)) < z$$

2. It holds that:

$$v^p(c(p,q)) + v^q(c(p,q)) > z$$

The column $c(p,q)$ contains z edges. If the sum of the vertical movements over these edges is greater than z, it means that the two vehicles share some of them. In this situation we have that at least one edge belongs to the path of both vehicles but, this is impossible because a conflict would arise (see point 1). Also the second assumption brings to a conflict so it cannot be true.

Both cases give a contradiction, it follows that the thesis must be true.

■

Remark 70 *Let's show that this new constraint is stronger than the one we could obtain combining some constraints of the model. We can rewrite constraint (44) in this way:*

$$\begin{aligned}\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) &\leqslant \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) - 1 + mw(p,q) + v^q(c(p,q)) - v^q(c(p,q)) \\ &= \sum_{i=c(p,q)}^{\alpha(q)} v^q(i) - 1 + mw(p,q) - v^q(c(p,q))\end{aligned}$$

and we can use constraint (45) at the second member:

$$\begin{aligned}\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) &\leqslant \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) - 1 + m(1 - w(p,q)) - 1 + mw(p,q) - v^q(c(p,q)) \\ &= \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) - 2 + m - v^q(c(p,q))\end{aligned}$$

Simplifying the last line, the constraint becomes:

$$\begin{aligned}\sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) + v^p(c(p,q)) &\leqslant \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) - 2 + m - v^q(c(p,q)) \\ v^p(c(p,q)) + v^q(c(p,q)) &\leqslant m - 2\end{aligned}$$

but it holds generally $z - 1 \leqslant m - 2$ and so our constraints is stronger than the combination of the others.

We give some examples in order to show that this new constraint cuts some solutions of the linear relaxation of lots of problems.

Example 71 *Assume we are working with an instance of the problem composed of 5 vehicles, assume the starting positions and the final positions of the vehicles are respectively $\alpha = (1, 2, 3, 4, 5)$ and $\omega = (2, 4, 5, 3, 1)$. It is clear that the couple $(2, 4)$ belongs to C_{even} and the conflict column associated to this couple is column 3. When we solve the continuos relaxation of this instance, we obtain a solution in which $z = 1$ and vehicle 4 moves one time vertically in correspondence of column 3 (see Figure 10). After adding the new constraint, this solution becomes unfeasible because:*

$$v^2(3) + v^4(3) = 1 \nleqslant z - 1 = 0$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 1
4 dual simplex iterations (0 in phase I)
z = 1

v :=
1 1   0
1 2   1
2 2   0
2 3   0
2 4   1
3 3   0
3 4   0
3 5   1
4 3   1
4 4   0
5 1   0
5 2   0
5 3   1
5 4   0
5 5   0
;

w :=
2 4   0.2
2 5   0.2
3 4   0.2
3 5   0.2
;
```

Figure 10: AMPL output example 71

Notice that it seems that vehicles 2 and 4 create a node conflict in correspondence of column 3, level 1. It is easy to see this, it is sufficient to draw the patterns associated to the vehicles. However, no conflict arises because the value of variables w are smaller than 1 and so all the constraints of the model that prevent node-conflicts are respected.

Example 72 *Assume we are working with another instance composed of 5 vehicles and assume* $\alpha = (1, 2, 3, 4, 5)$ *and* $\omega = (4, 5, 2, 1, 3)$*. It is clear that the couples* $(1, 3)$ *and* $(1, 5)$ *belong to* C_{even} *and the conflict column associated to this couples are respectively column 2 and 3. When we solve the continuos relaxation, we obtain a solution in which* $z = 1$*, vehicle 3 moves one time in column 2 and vehicle 5 moves one time in column 3 (see Figure 11). After adding the new constraint, this solution becomes unfeasible because:*

$$v^1(2) + v^3(2) = 1 \nleqq z - 1 = 0$$

$$v^1(3) + v^5(3) = 1 \nleqq z - 1 = 0$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 1
7 dual simplex iterations (0 in phase I)
z = 1

v :=
1 1   0
1 2   0
1 3   0
1 4   1
2 2   0
2 3   0
2 4   1
2 5   0
3 2   1
3 3   0
4 1   0
4 2   1
4 3   0
4 4   0
5 3   1
5 4   0
5 5   0
;

w :=
1 3   0.2
1 4   0.2
1 5   0.2
2 3   0.2
2 4   0.2
2 5   0.2
;
```

Figure 11: AMPL output example 72

Also in this case, there are no conflicts in level 1 thanks to the value of variables $w < 1$.

6.4 Fourth Set of Inequalities

These constraints regard the maximum number of vertical moves that a vehicle, involved in a potential conflict, can perform in a certain area of the grid. For every couple $(p, q) \in C_{odd}$ we can add four constraints. Notice that each constraint is about a different group of columns. If we consider vehicle p, we have a constraint regarding columns between $\alpha(p)$ and $c(p, q)$ and another constraint regarding columns between $c(p, q) + 1$ and $\omega(p)$. If we consider vehicle q, we have a constraint regarding columns between $\omega(q)$ and $c(p, q)$ and another constraint regarding columns between $c(p, q)+1$ and $\alpha(q)$.

Theorem 73 *Given any instance of FQRP, for any feasible solution of formulation C, it holds that:*

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant z - w(p, q) \quad \forall (p, q) \in C_{odd} \tag{48}$$

$$\sum_{i=c(p,q)+1}^{\omega(p)} v^p(i) \leqslant z - w(p, q) \quad \forall (p, q) \in C_{odd} \tag{49}$$

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant z - 1 + w(p, q) \quad \forall (p, q) \in C_{odd} \tag{50}$$

$$\sum_{i=\omega(q)}^{c(p,q)} v^q(i) \leqslant z - 1 + w(p, q) \quad \forall (p, q) \in C_{odd} \tag{51}$$

All these constraints state that the number of vertical moves that vehicles p and q can perform in a certain set of columns is dominated.

Proof. Let's consider the case $(p, q) \in C_{odd}$. We want to demonstrate constraint (48):

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant z - w(p, q)$$

If $w(p, q) = 0$, constraint (48) becomes:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant z$$

which is trivially true as consequence of constraint (41) of the model stating that:

$$\sum_{i \in I(k)} v^k(i) = z \quad \forall k \in P$$

The second case occurs when $w(p,q) = 1$ and constraint (48) becomes:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant z - 1 \tag{52}$$

The quantity at the first member of (52) cannot be strictly greater than z, let's show that it can not be equal to z. Recall that $w(p,q) = 1$ means that vehicle p crosses the conflict-columns $c(p,q)$ and $c(p,q)+1$ in a strictly greater level than vehicle q. If vehicle q performed z vertical moves in columns between $c(p,q)+1$ and $\alpha(q)$, vehicle p could not manage to cross the conflict-columns above vehicle q but, this contradicts the fact that $w(p,q) = 1$. Consequently, we have:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) < z$$

and (52) holds.
The proof of (49) is similar to the previous one. We proceed with the proofs of the other two constraints (50) and (51).
Let's consider constraint (50):

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant z - 1 + w(p,q)$$

and again we have two cases. The case $w(p,q) = 1$ is trivial. If $w(p,q) = 0$, the constraint becomes:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant z - 1 \tag{53}$$

The quantity at the first member of (53) cannot be greater than z, let's show that it can not be equal to z. Recall that $w(p,q) = 0$ means that vehicle p crosses the conflict-columns $c(p,q)$ and $c(p,q)+1$ in a strictly lower level than vehicle q. If vehicle p performed z vertical moves in columns between

$\alpha(p)$ and $c(p,q)$, vehicle q could not manage to cross the conflict-columns above vehicle p but, this contradicts the fact that $w(p,q) = 0$. Consequently, we have:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) < z$$

and (53) holds.
The proof of (51) is similar. ■

Theorem 74 *Inequalities (48) and (49) are equivalent.*

Proof. We are going to show that (48) implies (49) and vice versa.
Given $(p,q) \in C_{odd}$, the case with $w(p,q) = 0$ is trivial for both the implications. Suppose $w(p,q) = 1$, so constraint (48) becomes:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant z - 1$$

We can rewrite it in this way:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) = z - 1 - j \qquad j \in \{0, .., z-1\}$$

This means that:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \geqslant (z - 1 - j) + 1 = z - j \tag{54}$$

because $w(p,q) = 1$ and vehicle p crosses the column $c(p,q)$ in a strictly higher level than vehicle q. At this point, equation (54) implies that:

$$\sum_{i=c(p,q)+1}^{\omega(p)} v^p(i) \leqslant z - (z - j) = j \leqslant z - 1$$

which is exactly 49.
The other implication is very similar. Indeed, constraint (49) states:

$$\sum_{i=c(p,q)+1}^{\omega(p)} v^p(i) \leqslant z-1$$

We can rewrite it in this way:

$$\sum_{i=c(p,q)+1}^{\omega(p)} v^p(i) = z-1-j \qquad j \in \{0,..,z-1\}$$

This means that:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) = z-(z-1-j) = j+1$$

because the sum of the vertical moves over all the columns associated to p is z. Since $w(p,q)=1$, it holds that:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant (j+1)-1 = j \leqslant z-1$$

which is exactly constraint (48). ■

Theorem 75 *Inequalities (50) and (51) are equivalent.*

Proof. We are going to show that (50) implies (51) and vice versa.
Given $(p,q) \in C_{odd}$, the case $w(p,q)=1$ is trivial for both the implications.
Assume $w(p,q)=0$, constraint (50) becomes:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant z-1$$

We can rewrite it in this way:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) = z-1-j \qquad j \in \{0,..,z-1\}$$

This means that:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \geqslant (z-1-j)+1 = z-j$$

because $w(p,q) = 0$ and vehicle p crosses the column $c(p,q)$ in a strictly lower level than vehicle q. At this point, the previous equation implies that:

$$\sum_{i=\omega(q)}^{c(p,q)} v^q(i) \leqslant z-(z-j) = j \leqslant z-1$$

which is exactly constraint (51). The other implication is very similar. Constraint (51) can be rewritten as follow:

$$\sum_{i=\omega(q)}^{c(p,q)} v^q(i) = z-1-j \qquad j \in \{0,..,z-1\}$$

This means that:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) = z-(z-1-j) = j+1$$

Since $w(p,q) = 0$, it holds:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant (j+1)-1 = j \leqslant z-1$$

which is constraint (50). ■

Similarly, we can add 4 new constraints for every pair of vehicles that create an even conflict.

Theorem 76 *Given any instance of FQRP, for any feasible solution of formulation C, it holds that:*

$$\sum_{i=c(p,q)}^{\omega(p)} v^p(i) \leqslant z-w(p,q) \quad \forall (p,q) \in C_{even} \tag{55}$$

$$\sum_{i=c(p,q)}^{\alpha(q)} v^q(i) \leqslant z - w(p,q) \quad \forall (p,q) \in C_{even} \tag{56}$$

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant z - 1 + w(p,q) \quad \forall (p,q) \in C_{even} \tag{57}$$

$$\sum_{i=\omega(q)}^{c(p,q)} v^q(i) \leqslant z - 1 + w(p,q) \quad \forall (p,q) \in C_{even} \tag{58}$$

Likewise these constraints establish an upper bound for the number of vertical moves that a vehicle can perform in a certain set of columns.

Proof. Let's consider $(p,q) \in C_{even}$. We want to demonstrate constraint (55):

$$\sum_{i=c(p,q)}^{\omega(p)} v^p(i) \leqslant z - w(p,q)$$

If $w(p,q) = 0$, the constraint becomes:

$$\sum_{i=c(p,q)}^{\omega(p)} v^p(i) \leqslant z$$

which is trivially true. If $w(p,q) = 1$, the constraint becomes:

$$\sum_{i=c(p,q)}^{\omega(p)} v^p(i) \leqslant z - 1 \tag{59}$$

The quantity at the first member of (59) is dominated by z. We show that it cannot be equal to z. Indeed, if vehicle p performed all the vertical moves after the conflict-column $c(p,q)$, it could not cross column $c(p,q)$ at a strictly higher level than vehicle q. But, this is a contradiction because we have supposed that $w(p,q) = 1$.

The proof of (56) is similar to this one. Let's proceed with the proof of (57) which states that:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant z - 1 + w(p,q)$$

We have two cases: the case with $w(p,q) = 1$ is trivial. Assume $w(p,q) = 0$, the constraint becomes:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant z - 1$$

which is true because p crosses the conflict column in a strictly lower level than q and, if p performed z vertical moves in columns between $\alpha(p)$ and $c(p,q)$, q could not pass the column $c(p,q)$ above p.
Finally, the proof of (58) is similar to this previous one. ■

Theorem 77 *Inequalities (55) and (56) are equivalent.*

Proof. We want to show that constraint (55) implies constraint (56) and vice versa. Given $(p,q) \in C_{even}$, if $w(p,q) = 0$ both the implications are trivial. Let's assume $w(p,q) = 1$, we show first that (55) implies (56). Constraint (55) can be rewritten this way:

$$\sum_{i=c(p,q)}^{\omega(p)} v^p(i) = z - 1 - j \qquad j \in \{0, .., z-1\}$$

This implies that:

$$\sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) = z - (z - 1 - j) = j + 1$$

Since $w(p,q) = 1$ and q has to cross the conflict column with p in a strict lower level than p, it holds:

$$\sum_{i=c(p,q)}^{\alpha(q)} v^q(i) \leqslant (j+1) - 1 = j \leqslant z - 1$$

as we wanted. For the other implication, we can proceed like this. Constraint (56) can be rewritten this way:

$$\sum_{i=c(p,q)}^{\alpha(q)} v^q(i) = z - 1 - j \qquad j \in \{0, .., z-1\}$$

Since $w(p,q) = 1$ and q has to cross the conflict column with p in a strict lower level than p, it holds:

$$\sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) \geqslant (z-1-j)+1 = z-j$$

which implies:

$$\sum_{i=c(p,q)}^{\omega(p)} v^p(i) \leqslant z-(z-j) = j \leqslant z-1$$

as we wanted to prove. ■

Theorem 78 *Inequalities (57) and (58)are equivalent.*

Proof. We want to show that constraint (57) implies constraint (58) and vice versa. Given $(p,q) \in C_{even}$, if $w(p,q) = 1$ both the implications are trivial. Let's assume $w(p,q) = 0$, we show first that (57) implies (58). Constraint (57) can be rewritten this way:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) = z-1-j \quad j \in \{0,..,z-1\}$$

Since $w(p,q) = 0$ and q has to cross the conflict column with p in a strict higher level than p, it holds:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \geqslant (z-1-j)+1 = z-j$$

which implies:

$$\sum_{i=\omega(q)}^{c(p,q)} v^q(i) \leqslant z-(z-j) = j \leqslant z-1$$

as we wanted. The other implication proceeds similarly:

$$\sum_{i=\omega(q)}^{c(p,q)} v^q(i) = z-1-j \quad j \in \{0,..,z-1\}$$

This means that:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) = z - (z - 1 - j) = j + 1$$

Since $w(p, q) = 0$ and q has to cross the conflict column with p in a strict higher level than p, it holds:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant (j + 1) - 1 = j \leqslant z - 1$$

as we wanted to prove. ■

Remark 79 *Even if we have just shown that these constraints are equivalent 2 by 2, we add them all to the model. Indeed, they are all valid cuts, as we can see from the following examples.*

Example 80 *Suppose* $\alpha = (1, 2, 3, .., 14, 15)$ *and suppose* $\omega = (1, 12, 14, 2, 13, 10, 6, 7, 5, 1, 15, 3, 8, 9, 4)$. *When we solve the continuous relaxation, we obtain an optimal solution such that* $z = 1$. *The optimal solution of the continuous relaxation violates constraint (48) in correspondence of couple* $(6, 13) \in C_{odd}$ *(see Figure 12). Provided that the conflict columns of this couple are columns 9 and 10 and that* $w(6, 13) = 0.2$, *it holds:*

$$\sum_{i=10}^{13} v^{13}(i) = 1 \nleqq 0.8$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 1
38 dual simplex iterations (0 in phase I)
z = 1

v [*,*] (tr)
:    2   3   4   5   6   7   8   9  10  11  12  13  14  15    :=
10   0   0   .   0   0   .   .   .   0   .   0   1   0   0
11   1   1   .   1   .   .   .   .   .   0   0   0   0   0
12   0   0   .   0   .   .   .   .   .   0   0   0   0   0
13   .   0   .   0   .   .   .   .   .   0   .   0   0   0
...
;

w [*,*] (tr)
:     2     3     5     6    11     :=
...
13    .    0     0     0.2   0.2
;
```

Figure 12: AMPL output example 80

Example 81 *Suppose* $\alpha = (1, 2, 3, .., 19, 20)$ *and suppose* $\omega = (14, 8, 7, 16, 11, 4, 3, 15, 6, 17, 10, 13, 20, 9, 12, 1, 18, 5, 2, 19)$. *When we solve the continuous relaxation, we obtain an optimal solution such that* $z = 1$. *This solution in correspondence of couple* $(2, 7) \in C_{odd}$ *violates constraint (49). Provided that the conflict columns of this couple are columns 4 and 5 and that* $w(2, 7) = 0.25$, *it holds:*

$$\sum_{i=5}^{8} v^2(i) = 1 \nleqq 0.75$$

Constraint (50) is violated by the solution in correspondence of couple $(17, 18) \in C_{odd}$. *Provided that the conflict columns of this couple are columns 17 and 18 and that* $w(17, 18) = 0.5$, *it holds:*

$$\sum_{i=17}^{17} v^{17}(i) = 1 \nleqq 0.5$$

A violation of inequality (51) occurs in correspondence of couple $(1, 16) \in C_{odd}$. *The conflict columns of this couple are columns 8 and 9 and* $w(1, 16) = 0.25$ *(see Figure 13). We can see easily that:*

$$\sum_{i=1}^{8} v^{16}(i) = 1 \nleqq 0.25$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 1
47 dual simplex iterations (0 in phase I)
z = 1

v [*,*]
:    1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19 :=
1    0   0   0   0   0   0   0   0   0   0   0   0   0   1   .   .   .   .   .
2    .   0   0   0   0   0   0   1   .   .   .   .   .   .   .   .   .   .   .
...
7    .   .   1   0   0   0   0   .   .   .   .   .   .   .   .   .   .   .   .
16   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   .   .   .
17   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   0   .
18   .   .   .   .   1   0   0   0   0   0   0   0   0   0   0   0   0   0   .
;

w [*,*]
:    6    7      9      11     14     15     16     18     19     :=
1    .   0.25    .      .      .      .     0.25   0.25   0.25
2    0   0.25    .      .      .      .      .      .      .
...
17   .    .      .      .      .      .      .     0.5    0.5
;
```

Figure 13: AMPL output example 81

Example 82 *Suppose* $\alpha = (1, 2, 3, .., 19, 20)$ *and suppose* $\omega = (8, 14, 7, 6, 10, 4, 9, 5, 2, 11, 1, 13, 19, 3, 12, 17, 18, 15, 16, 20)$. *When we solve the continuous relaxation, we obtain an optimal solution such that* $z = 1$. *This solution in correspondence of couple* $(1, 11) \in C_{even}$ *violates constraint (55). Provided that the conflict column of this couple is column 6 and that* $w(1, 11) = 0.25$*, it holds:*

$$\sum_{i=6}^{8} v^1(i) = 1 \nleq 0.75$$

Constraint (56) is violated by the solution in correspondence of couple $(4, 6)$. *Provided that the conflict column of this couple is column 5 and that* $w(4, 6) = 0.25$*, it holds:*

$$\sum_{i=5}^{6} v^6(i) = 1 \nleq 0.75$$

A couple of even conflicts where there is a violation of inequality (57)is couple $(4, 8)$*, their conflict column is column 6 and* $w(4, 8) = 0.25$. *We can*

see that:

$$\sum_{i=4}^{6} v^4(i) = 1 \nleq 0.25$$

We provide a last couple of even conflicts where occurs a violation of inequality (58):the couple is $(1, 9)$*, their conflict column is column 5 and* $w(1, 9) = 0.25$ *(see Figure 14). We can see that:*

$$\sum_{i=2}^{5} v^9(i) = 1 \nleq 0.25$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 1
44 dual simplex iterations (0 in phase I)
z = 1

v [*,*]
:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19 :=
1   0   0   0   0   0   0   0   1   .   .   .   .   .   .   .   .   .   .   .
4   .   .   .   1   0   0   .   .   .   .   .   .   .   .   .   .   .   .   .
6   .   .   .   0   0   1   .   .   .   .   .   .   .   .   .   .   .   .   .
9   .   0   1   0   0   0   0   0   0   .   .   .   .   .   .   .   .   .   .
...
;

w [*,*]
:    6      8      9     11     14     15     18     19     :=
1    .      .    0.25   0.25   0.25    .      .      .
4   0.25   0.25    .      .      .      .      .      .
...
;
```

Figure 14: AMPL output example 82

Remark 83 *This remark is an empirical observation. After lots of tests with instances of different size, we have noticed that the continuous relaxation of model C together with the inequalities in sections "First Set of Inequalities", "Second Set of Inequalities", "Third Set of Inequalities" and "Fourth Set of Inequalities" always provides an optimal integer solution if* $z = 1$*. In other words, if the optimal value of the objective function of the integer problem is 1, it happens that the relaxation of the instance provides an optimal solution such that* $z = 1$ *and all the other variables* v *and* w *are integer.*

6.5 Fifth Set of Inequalities

These inequalities come from the attempt to find a minimum number of vertical moves that a vehicle, involved in a conflict, can perform in a certain area of the grid.

Theorem 84 *Given any instance of FQRP, for every feasible solution of formulation C, it holds:*

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \geqslant w(p,q) \quad \forall (p,q) \in C_{odd} \tag{60}$$

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \geqslant 1 - w(p,q) \quad \forall (p,q) \in C_{odd} \tag{61}$$

Similarly, we suggest a couple of inequalities also for even conflicts.

Theorem 85 *Given any instance of FQRP, for every feasible solution of formulation C, it holds:*

$$\sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) \geqslant w(p,q) \quad \forall (p,q) \in C_{even} \tag{62}$$

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \geqslant 1 - w(p,q) \quad \forall (p,q) \in C_{even} \tag{63}$$

We proceed giving the proofs of these inequalities.

Proof. Inequality (60) follows from constraint (43) which states:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant \sum_{i=\alpha(p)}^{c(p,q)} v^p(i) - 1 + m(1 - w(p,q))$$

If $w(p,q) = 0$, the inequality is trivial. If $w(p,q) = 1$, the inequality becomes:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) - 1 \geqslant \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \geqslant 0$$

It follows that:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \geqslant 1 = w(p,q)$$

which is exactly (60). The proof of (61) follows from (42), which states:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) - 1 + mw(p,q)$$

The case $w(p,q) = 1$ is trivial. Assume $w(p,q) = 0$, the constraint becomes:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) - 1 \geqslant \sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \geqslant 0$$

It follows that:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \geqslant 1 - 0 = 1 - w(p,q)$$

as we wanted to prove. ■

The proof of the other two inequalities is very similar to the previous one.

Proof. Inequality (62) follows from constraint (45) which states:

$$\sum_{i=c(p,q)}^{\alpha(q)} v^q(i) \leqslant \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) - 1 + m(1 - w(p,q))$$

The case with $w(p,q) = 0$ is trivial. Assume $w(p,q) = 1$, the constraint becomes:

$$\sum_{i=c(p,q)}^{\alpha(q)} v^q(i) \leqslant \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) - 1$$

We can notice that:

$$\sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) - 1 \geqslant \sum_{i=c(p,q)}^{\alpha(q)} v^q(i) \geqslant 0$$

i.e.:

$$\sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) \geqslant 1 = w(p,q)$$

Finally, the proof of (63) follows from constraint (44), which states:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) - 1 + mw(p,q)$$

The case $w(p,q) = 1$ is trivial. If $w(p,q) = 0$, the previous constraint becomes:

$$\sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) - 1$$

which can be rewritten as:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) - 1 \geqslant \sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \geqslant 0$$

It follows that:

$$\sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \geqslant 1 - 0 = 1 - w(p,q)$$

■

Up to now, we have tested the validity of a new constraint keeping in model C all the inequalities that we have previously discovered. This way, we do not test the validity of a new inequality alone, but we test the validity of a new inequality in relation to the inequalities that have been previously added to the model. This means that we try to see if the very last inequality cuts away different points from the feasible region of a relaxed problem with respect to the previous inequalities. In particular, we can notice that these last 4 inequalities are not valid cuts if we keep inequalities (48)-(51) and (55)-(58) in the model, otherwise they become valid inequalities.

Theorem 86 *Inequalities (60)-(63) are equivalent to inequalities (48)-(51) and (55)-(58).*

Proof. Inequality (60) comes from inequality (49) and vice versa. Indeed, inequality (49) can be rewritten this way, using constraint (41) of the model:

$$\sum_{i=c(p,q)+1}^{\omega(p)} v^p(i) \leqslant z - w(p,q)$$
$$z - \sum_{i=\alpha(p)}^{c(p,q)} v^p(i) \leqslant z - w(p,q)$$

Simplifying the last line we obtain inequality (60).
Inequality (61) comes from inequality (51) and vice versa. Indeed, inequality (51) can be rewritten this way, using constraint (41) of the model:

$$\sum_{i=\omega(q)}^{c(p,q)} v^q(i) \leqslant z - 1 + w(p,q)$$
$$z - \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant z - 1 + w(p,q)$$

Simplifying the last line we obtain inequality (61).
Inequality (62) comes from inequality (55) and vice versa. Indeed, inequality (55) can be rewritten this way, using constraint (41) of the model:

$$\sum_{i=c(p,q)}^{\omega(p)} v^p(i) \leqslant z - w(p,q)$$
$$z - \sum_{i=\alpha(p)}^{c(p,q)-1} v^p(i) \leqslant z - w(p,q)$$

Simplifying the last line we obtain inequality (62).
Inequality (63) comes from inequality (58) and vice versa. Indeed, inequality (58) can be rewritten this way, using constraint (41) of the model:

$$\sum_{i=\omega(q)}^{c(p,q)} v^q(i) \leqslant z - 1 + w(p,q)$$
$$z - \sum_{i=c(p,q)+1}^{\alpha(q)} v^q(i) \leqslant z - 1 + w(p,q)$$

Simplifying the last line we obtain inequality (63). ■

We can show that inequalities (60)-(63) are valid inequalities if added to model C without constraints (48)-(51) and (55)-(58).

Example 87 *Assume* $\alpha = (1, 2, .., 14, 15)$ *and* $\omega = (9, 13, 2, 14, 12, 1, 6, 15, 5, 11, 8, 3, 7, 10, 4)$. *The optimal solution of the continuous relaxation (see Figure 15) in correspondence to couple* $(1, 6) \in C_{odd}$ *does not respect inequality (60):*

$$\sum_{i=1}^{3} v^1(i) = 0 \ngeq w(1, 6) = 0.25$$

The optimal solution of the continuous relaxation in correspondence to couple $(2, 15) \in C_{odd}$ *does not respect inequality (61):*

$$\sum_{i=9}^{15} v^{15}(i) = 0 \ngeq 1 - w(2, 15) = 0.75$$

The optimal solution of the continuous relaxation in correspondence to couple $(8, 12) \in C_{even}$ *does not respect inequality (62):*

$$\sum_{i=8}^{9} v^8(i) = 0 \ngeq w(8, 12) = 0.25$$

The optimal solution of the continuous relaxation in correspondence to couple $(5, 9) \in C_{even}$ *does not respect inequality (63):*

$$\sum_{i=8}^{9} v^9(i) = 0 \ngeq w(5, 9) = 0.25$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 1
37 dual simplex iterations (0 in phase I)
z = 1

v [*,*]
:    1   2   3   4   5   6   7   8   9  10  11  12  13  14  15    :=
1    0   0   0   0   0   0   0   0   1   .   .   .   .   .   .
2    .   0   0   0   0   0   0   0   1   0   0   0   0   .   .
3    .   0   1   .   .   .   .   .   .   .   .   .   .   .   .
4    .   .   .   0   0   0   0   0   0   0   1   0   0   0   .
5    .   .   .   .   0   0   0   0   0   0   0   1   .   .   .
6    0   1   0   0   0   0   .   .   .   .   .   .   .   .   .
7    .   .   .   .   .   0   1   .   .   .   .   .   .   .   .
8    .   .   .   .   .   .   .   0   0   0   0   0   1   0   0
9    .   .   .   .   0   1   0   0   0   .   .   .   .   .   .
10   .   .   .   .   .   .   .   .   .   1   0   .   .   .   .
11   .   .   .   .   .   .   .   0   1   0   0   .   .   .   .
12   .   .   0   0   1   0   0   0   0   0   0   0   .   .   .
13   .   .   .   .   .   .   0   1   0   0   0   0   0   .   .
14   .   .   .   .   .   .   .   .   .   1   0   0   0   0   .
15   .   .   .   0   0   1   0   0   0   0   0   0   0   0   0
;

w [*,*] (tr)
:     1      2      4      5      8     10     :=
3     0      0      .      .      .      .
6     0.25   0.25   0.25   0.25   .      .
7     .      .      .      0      .      .
9     0      0      0.25   0.25   0.25   .
11    .      .      .      0      0.25   0.5
12    0.25   0.25   0.25   0.25   0.25   0.5
13    0      0      0.25   0.25   0.25   .
14    .      .      .      .      0.25   .
15    0.25   0.25   0.25   0.25   0.25   .
;
```

Figure 15: AMPL output example 87

Remark 88 *Notice that, in the case in which we decide to use valid inequalities (60)-(63), inequalities (60) and (62) dominate inequality (46) due to (41), because the sums of the LHS in (60) and (62) contain a subset of indexes appearing in (41).*

Remark 89 *We continue our study using inequalities of sections "First Set of Inequalities", "Second Set of Inequalities", "Third Set of Inequalities" and "Fourth Set of Inequalities".*

6.6 Sixth Set of Inequalities

The inequalities we are going to suggest in this part regard chains of conflicting vehicles. We do not consider simple pairs of vehicles but we combine some of them in order to obtain a chain of conflicts.

Definition 90 *(Chain of Conflicts) Let's consider A_i with $i \in \{1, .., t\}$ and $t \leqslant n$ vehicles. They constitute a chain of conflicts if each vehicle A_i creates a conflict with vehicle A_{i+1} with $i \in \{1, .., t-1\}$ and vehicle $A_1 \in R$.*

Example 91 *Let's consider a chain formed by 4 vehicles. We have that $A_1 \in R$ by definition, so A_2 must be an element of set L. The couple (A_1, A_2) belongs to C_{even} or to C_{odd} indifferently. Vehicle A_2 creates a conflict with vehicle A_3 by definition. Since $A_2 \in L$, vehicle A_3 must be an element of set R. This way, the second couple of conflicts is (A_3, A_2) and, it belongs to C_{even} or to C_{odd} indifferently. Finally, since $A_3 \in R$, vehicle A_4 must be an element of set L. This way, the third couple of conflicts is (A_3, A_4) and, it belongs to C_{even} or to C_{odd} indifferently.*

The structural idea of these new constraints is the following: given a chain of conflicts, we want to limit the number of vertical moves that the first vehicle of the chain can perform in a certain area of the grid. Indeed, let's suppose vehicle A_1 has to pass above A_2 over their conflict column(s) and vehicle A_2 has to pass above A_3 over their conflict column(s) and so on, in this way it must be true that vehicle A_1 has to perform a certain number of vertical moves before a specific column if we want it to pass above all the other vehicles and if we want to avoid conflicts (see Appendix 2, Figure 5). In order to force vehicle A_1 to move early in the grid, we have to understand if this constriction can be done for a generic chain or if we need a chain with particular properties and we also have to clarify which is the column until which vehicle A_1 has to perform the moves. We can start considering a chain composed by only three vehicles: there is a vehicle A_1, which is in conflict with a vehicle A_2 and this last one is in conflict with a vehicle A_3. It is obvious that vehicles A_1 and A_3 belong to R, while vehicle A_2 belongs to L.
If we want to control the column up to which vehicle A_1 has to perform the moves, the vehicles cannot be randomly settled. Let's distinguish two cases, depending on the nature of the conflict of the first pair of vehicles. Suppose the first pair (A_1, A_2) belongs to C_{even}. The starting-point of vehicle A_3 cannot be settled in any column. It is obvious that $\alpha(A_3)$ cannot be greater than $\alpha(A_2)$; if $\alpha(A_3)$ stands between $\alpha(A_1)$ and $\alpha(A_2)$, it is possible to find

a constraint about the number of vertical moves that A_1 performs up to column $c(A_1, A_2) - 1$; if $\alpha(A_3)$ is smaller than $\alpha(A_1)$, we cannot always state how many vertical moves A_1 performs up to $c(A_1, A_2) - 1$, indeed if $\alpha(A_3)$ is too far from $\alpha(A_1)$ we cannot say anything. We can show that if $\alpha(A_3)$ is in one of the two nearest positions on the left of $\alpha(A_1)$, we can continue to limit the number of vertical moves of A_1 up to the column $c(A_1, A_2) - 1$.

Definition 92 *(3-vehicle chain of type 1) Let's consider A_i with $i \in \{1, 2, 3\}$, a 3-vehicle chain of type 1 is a chain of conflicts such that the pair (A_1, A_2) belongs to C_{even}, the pair (A_3, A_2) belongs to C_{odd} or to C_{even} indifferently and $\alpha(A_1) - 2 \leqslant \alpha(A_3) < \alpha(A_2)$.*

The second case occurs when the pair (A_1, A_2) belongs to C_{odd}. Similarly in this case, A_3 cannot be settled in any column. It is obvious that $\alpha(A_3)$ cannot be greater than $\alpha(A_2)$; if $\alpha(A_3)$ stands between $\alpha(A_1)$ and $\alpha(A_2)$, it is possible to find a constraint about the number of vertical moves that A_1 performs up to column $c(A_1, A_2)$. We can show that if $\alpha(A_3)$ is at most in the first position on the left of $\alpha(A_1)$, we can continue to limit the number of vertical moves of A_1 up to the column $c(A_1, A_2)$ (for a detailed explanation see the following remark).

Definition 93 *(3-vehicle chains of type 2) Let's consider A_i with $i \in \{1, 2, 3\}$, a 3-vehicle chain of type 2 is a chain of conflicts such that the pair (A_1, A_2) belongs to C_{odd}, the pair (A_3, A_2) belongs to C_{odd} or to C_{even} indifferently and $\alpha(A_1) - 1 \leqslant \alpha(A_3) < \alpha(A_2)$.*

Remark 94 *We want to clarify the reason why we choose to put the starting-position of A_3 only in a limited range of columns. Indeed, if the conflict column(s) between A_2 and A_3 is too far left with respect to the conflict column(s) of A_1 and A_2, it is not true any more that vehicle A_1 has to move early in order to avoid conflicts and pass above the other vehicles.*
Recall the previous distinction: suppose the first pair (A_1, A_2) belongs to C_{even} and suppose A_3 creates any conflict with A_2 and has its starting-position in one of the above columns. Suppose also that A_1 has to pass above A_2 over their conflict column and vehicle A_2 has to pass above A_3 over their conflict column(s). Due to the position of A_3, if vehicle A_2 has to perform one move to pass above A_3, it does it at most in correspondence of column $c(A_1, A_2)$. This obviously implies that, if A_1 has to pass above A_2, it performs at least two moves up to column $c(A_1, A_2) - 1$. The fact that A_2

moves (forced by A_3) before the conflicting column with A_1 is the real reason thanks to that we are allowed to force vehicle A_1 to move so early. In other words, we can say this because A_2 reaches $c(A_1, A_2)$ after having performed at least one vertical step. By contrast, if $\alpha(A_3)$ stands in the far left of A_1, it could be possible that A_1 passes above A_2 and A_2 passes above A_3 in their respective conflict columns but, at the same time, A_1 is not forced to perform the same number of moves like before up to column $c(A_1, A_2) - 1$. Indeed, A_2 could perform its vertical move also strictly before $c(A_1, A_2)$ (so it passes the column at a strict lower level than before) and so A_1 can perform one less move in the same range of column, avoiding conflicts.
Similarly, suppose the first pair (A_1, A_2) belongs to C_{odd} and suppose A_3 create any conflict with A_2 and has its starting-position in one of the above columns. Suppose also that A_1 has to pass above A_2 over their conflict columns and vehicle A_2 has to pass above A_3 over their conflict column(s). Due to the position of A_3, if vehicle A_2 has to perform one move to pass above A_3, it does it at most in correspondence of column $c(A_1, A_2)+1$. This obviously implies that, if A_1 has to pass above A_2, it performs at least two moves up to column $c(A_1, A_2)$ because A_2 reaches $c(A_1, A_2)$ after having performed at lest one vertical step.
Limiting the distance between $\alpha(A_1)$ and $\alpha(A_3)$ is good to define the column up to which A_1 has to perform some moves.

We are ready to give the inequalities.

Theorem 95 *Given any instance of FQRP, the following inequalities hold for every feasible solution of formulation C. The first one regards a 3-vehicle chain of type 1 and the second regard a 3-vehicle chain of type 2:*

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)-1} v^{A_1}(i) \geqslant 2w(A_1, A_2) - w(A_3, A_2) \tag{64}$$

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)} v^{A_1}(i) \geqslant 2w(A_1, A_2) - w(A_3, A_2) \tag{65}$$

The inequalities states that vehicle A_1 has to perform at least a certain number of vertical moves up to a certain column. We are going to show the proof of these new constraints.

Proof. For the proof of (64), we can proceed in this way. Let's consider all the cases that can occur with 3 vehicles.

1. We can have that A_1 passes above A_2 and A_2 passes above A_3. This means that $w(A_1, A_2) = 1$ and $w(A_3, A_2) = 0$. In this case the constraint states that:

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)-1} v^{A_1}(i) \geqslant 2$$

   In order to prove so, let's suppose ad absurdum that the previous inequality does not hold, i.e. vehicle A_1 performs a number of vertical moves strictly smaller than 2 up to column $c(A_1, A_2) - 1$. This means that it performs either 1 vertical move or no vertical moves but, in both these cases we obtain a contradiction. Indeed, if A_1 performs only 1 vertical move it creates a conflict with vehicle A_2 in correspondence to column $c(A_1, A_2)$. This is because A_2 has to move one time vertically in order to leave A_3 to pass under it and A_2 reaches column $c(A_1, A_2)$ at level 1 (the same of A_3). In the other case, if A_1 does not perform any vertical move, so it has no chance to pass above A_2 but this contradicts the fact that $w(A_1, A_2) = 1$.

2. We can have that A_1 passes under A_2 and A_2 passes under A_3, and this means that $w(A_1, A_2) = 0$ and $w(A_3, A_2) = 1$. In this case the constraint is trivial.

3. We can have that A_1 passes under A_2 and A_2 passes above A_3, and it means that $w(A_1, A_2) = 0$ and $w(A_3, A_2) = 0$. In this case the constraint is trivial.

4. We can have that A_1 passes above A_2 and A_2 passes under A_3, and it means that $w(A_1, A_2) = 1$ and $w(A_3, A_2) = 1$. The constraint becomes:

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)-1} v^{A_1}(i) \geqslant 1$$

   Let's suppose ad absurdum that the previous inequality does not hold, i.e. vehicle A_1 performs a number of vertical moves strictly smaller than 1 up to column $c(A_1, A_2) - 1$. This means that it performs no one vertical move but, this is a contradiction because in this way, it has no chance to pass above A_2 in contrast with the fact that $w(A_1, A_2) = 1$.

For the proof of (65), we proceed again by cases.

1. We can have that A_1 passes above A_2 and A_2 passes above A_3. This means that $w(A_1, A_2) = 1$ and $w(A_3, A_2) = 0$. In this case the constraint states that:

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)} v^{A_1}(i) \geqslant 2$$

   In order to prove so, let's suppose ad absurdum that the previous inequality does not hold, i.e. vehicle A_1 performs a number of vertical moves strictly smaller than 2 up to column $c(A_1, A_2)$. This means that it performs either 1 vertical move or no vertical moves but, in both these cases we obtain a contradiction. Indeed, if A_1 performs only 1 vertical move it creates a conflict with vehicle A_2 in correspondence to an edge joining columns $c(A_1, A_2)$ and $c(A_1, A_2) + 1$. This is because A_2 has to move one time vertically in order to leave A_3 to pass under it and A_2 reaches column $c(A_1, A_2)+1$ at level 1 (the same level in which A_1 reaches $c(A_1, A_2)$). In the other case, if A_1 does not perform any vertical move, it has no chance to pass above A_2, but this contradicts the fact that $w(A_1, A_2) = 1$.

2. We can have that A_1 passes under A_2 and A_2 passes under A_3, and this means that $w(A_1, A_2) = 0$ and $w(A_3, A_2) = 1$. In this case the constraint is trivial.

3. We can have that A_1 passes under A_2 and A_2 passes above A_3, and it means that $w(A_1, A_2) = 0$ and $w(A_3, A_2) = 0$. In this case the constraint is trivial.

4. We can have that A_1 passes above A_2 and A_2 passes under A_3, and it means that $w(A_1, A_2) = 1$ and $w(A_3, A_2) = 1$. The constraint becomes:

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)} v^{A_1}(i) \geqslant 1$$

   Let's suppose ad absurdum that the previous inequality does not hold, i.e. vehicle A_1 performs a number of vertical moves strictly smaller than 1 up to column $c(A_1, A_2)$. This means that it performs no one

vertical move but, this is a contradiction because in this way, it has no chance to pass above A_2 in contrast with the fact that $w(A_1, A_2) = 1$.

■

Example 96 *Suppose* $\alpha = (1, 2, 3, .., 19, 20)$ *and suppose* $\omega = (15, 18, 4, 2, 9, 13, 17, 6, 5, 20, 19, 16, 10, 8, 14, 3, 7, 11, 1, 12)$. *The optimal solution of the continuous relaxation violates inequality (64) in correspondence of the first type chain (12,18,11), indeed (see Figure 16):*

$$\sum_{i=12}^{14} v^{12}(i) = 0.8 \ngeq 2w(12, 18) - w(11, 18) = 1$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 1.2
198 dual simplex iterations (0 in phase I)
z = 1.2

v [*,*]
:      9          10      11    12    13    14    15    16    17   18  19  20 :=
11     .          .       0.4   0.2   0     0.2   0.2   0     0.2  0   0   .
12     .          .         .   0.6   0     0.2   0.2   0.2    .   .   .   .
13     .          0       0.6   0     0.6    .     .     .     .   .   .   .
...
;

w [*,*] (tr)
:       1          2       3       5          6          7       11    12 :=
4    0          0          0     .          .          .          .     .
8     .          .         .   0.0555556   0.0851852   0.0555556   .     .
18    .          .         .    .         0.8         0.6         0.6   0.8
...
;
```

Figure 16: AMPL output example 96

Example 97 *Suppose* $\alpha = (1, 2, 3, .., 19, 20)$ *and* $\omega = (19, 18, 14, 20, 9, 12, 17, 16, 5, 2, 15, 6, 10, 8, 4, 13, 7, 11, 1, 3)$. *The optimal solution of the continuous relaxation violates inequality (65) in correspondence to the second type chain (6,9,8), indeed (see Figure 17):*

$$\sum_{i=6}^{7} v^{6}(i) = 0.00911458 \ngeq 2w(6, 9) - w(8, 9) = 0.01093749$$

```
ampl: include model3.run;
CPLEX 12.8.0.0: optimal solution; objective 2
178 dual simplex iterations (0 in phase I)
z = 2

v [*,*]
:    1   2   3   4   5       6        7       8            9          10     :=
5    .   .   .   .   0   0           0   0           2           .
6    .   .   .   .   .   0.00911458   0   0           0.0364583   0
...
;

w [*,*] (tr)
:    1   2   3   4   5       6           7           8           11        :=
9    0   0   0   0   0   0.00911458   0          0.00729167    .
...
;
```

Figure 17: AMPL output example 97

Remark 98 *Inequalities (64) and (65) use the hypothesis that $A_1 \in R$. It is possible to build symmetric inequalities with $A_1 \in L$, fixing analogous conditions and adapting all the observations to this case. However, we do not provide the details in this work.*

6.7 Seventh Set of Inequalities

This group of inequalities is similar to the previous one. We continue to use chains of conflicts and, especially, chains of conflicts formed by 4 vehicles. Recall that we have a vehicle $A_1 \in R$ which creates a conflict with a vehicle $A_2 \in L$, vehicle A_2 creates a conflict with a vehicle $A_3 \in R$ and this last one creates a conflict with a vehicle $A_4 \in L$. The central idea of these inequalities is basically the same of the previous section: assume vehicle A_1 has to pass above A_2 over their conflict column(s), vehicle A_2 has to pass above A_3 over their conflict column(s) and vehicle A_3 has to pass above A_4 over their conflict column(s), in this way it must be true that vehicle A_1 has to perform a certain number of vertical moves before a specific column if we want it to pass above all the other vehicles and if we want to avoid conflicts (see Appendix 2, Figure 6). Also in this case, we have to understand until which column vehicle A_1 is forced to move and also if the starting-positions of the vehicles must be assigned only in specific ranges of columns. As consequence of this structure, it is intuitive to think that we want to force A_1 to move vertically before meeting vehicle A_2, in other words, before reaching the conflict column(s) with vehicle A_2. In this way, the starting-positions of the vehicles cannot be randomly settled, so we introduce 4 types of chains which we are going to work with.

Definition 99 *(4-vehicle chain of type 1) Let's consider A_i with $i \in \{1,2,3,4\}$, a 4-vehicle chain of type 1 is a chain of conflicts such that the pair (A_1, A_2) belongs to C_{even}, the pair (A_3, A_2) belongs to C_{even}, the pair (A_3, A_4) belongs to C_{even} or to C_{odd} indifferently. Moreover, it holds $\alpha(A_1) - 2 \leqslant \alpha(A_3) < \alpha(A_2)$ and $\alpha(A_3) < \alpha(A_4) \leqslant \alpha(A_2) + 2$.*

Definition 100 *(4-vehicle chain of type 2) Let's consider A_i with $i \in \{1,2,3,4\}$, a 4-vehicle chain of type 2 is a chain of conflicts such that the pair (A_1, A_2) belongs to C_{even}, the pair (A_3, A_2) belongs to C_{odd}, the pair (A_3, A_4) belongs to C_{even} or to C_{odd} indifferently. Moreover, it holds $\alpha(A_1) - 2 \leqslant \alpha(A_3) < \alpha(A_2)$ and $\alpha(A_3) < \alpha(A_4) \leqslant \alpha(A_2) + 1$.*

Definition 101 *(4-vehicle chain of type 3) Let's consider A_i with $i \in \{1,2,3,4\}$, a 4-vehicle chain of type 3 is a chain of conflicts such that the pair (A_1, A_2) belongs to C_{odd}, the pair (A_3, A_2) belongs to C_{even}, the pair (A_3, A_4) belongs to C_{even} or to C_{odd} indifferently. Moreover, it holds $\alpha(A_1) - 1 \leqslant \alpha(A_3) < \alpha(A_2)$ and $\alpha(A_3) < \alpha(A_4) \leqslant \alpha(A_2) + 2$.*

Definition 102 *(4-vehicle chain of type 4) Let's consider A_i with $i \in \{1,2,3,4\}$, a 4-vehicle chain of type 4 is a chain of conflicts such that the pair (A_1, A_2)*

belongs to C_{odd}, the pair (A_3, A_2) belongs to C_{odd}, the pair (A_3, A_4) belongs to C_{even} or to C_{odd} indifferently. Moreover, it holds $\alpha(A_1)-1 \leqslant \alpha(A_3) < \alpha(A_2)$ and $\alpha(A_3) < \alpha(A_4) \leqslant \alpha(A_2)+1$.

Remark 103 *Notice that, in the previous definitions, the limitation on the range of columns in which every vehicle can start is "general". It represents the maximum limit until which these reasonings work. For example, in definition (100), it is clear that if $\alpha(A_3) = \alpha(A_1) - 2$ then the couple (A_3, A_2) would belong to C_{even} instead to C_{odd} as definition says. Generally, there are also other positions between $\alpha(A_1) - 2$ and $\alpha(A_2)$ such that vehicle A_3 (if settled there) would create an even conflict with vehicle A_2. For this reason, we ask the reader to choose correct starting points (according to the definitions) in the range of columns we provide.*

We can notice that the previous types of chains differ from each other by the nature of the conflicts involved and the starting-positions of the vehicles. It is important to establish this limitation on the starting-positions because in this way, A_1 is forced to move early if it has to pass above the other vehicles. Indeed, if the vehicles start from points which are too far away from each other, it is possible that A_1 passes above all the other vehicles but, at the same time, it is not forced any more to move early.

Theorem 104 *Given any instance of FQRP, the following two inequalities hold for any feasible solution of formulation C. The first one regards the first two types of 4-vehicle chains and the other regards the second two types of 4-vehicle chains.*

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)-1} v^{A_1}(i) \geqslant w(A_1,A_2) + w(A_1,A_2)(1-w(A_3,A_2))+ \\ + w(A_1,A_2)(1-w(A_3,A_2))w(A_3,A_4) \tag{66}$$

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)} v^{A_1}(i) \geqslant w(A_1,A_2) + w(A_1,A_2)(1-w(A_3,A_2))+ \\ + w(A_1,A_2)(1-w(A_3,A_2))w(A_3,A_4) \tag{67}$$

These inequalities state that A_1 has to perform at least a certain number of vertical moves up to a certain column if it passes over other vehicles. It has to perform at least 1 vertical move if it passes above A_2 over their conflict

column(s), it has to perform at least 2 vertical moves if it passes above A_2 over their conflict column(s) and A_2 passes above A_3 over their conflict column(s) and finally, it has to perform at least 3 vertical moves if it passes above A_2 over their conflict column(s), if A_2 passes above A_3 over their conflict column(s) and A_3 passes above A_4 over their conflict column(s).
We can immediately notice that these inequalities are not linear. We need to perform a linearization before putting them in the model. Since the RHS of the previous inequalities is the same, let's consider only the first one. The first term is trivially linear, because it is just a binary variable $w(A_1, A_2)$. The second and the third term are not linear but they both can assume only values 1 or 0: this property suggests that we can substitute them with other binary variables, paying attention to the fact that these variables must be correctly linked with the w-variables through some constraints. The second term $w(A_1, A_2)(1 - w(A_3, A_2))$ depends on A_1, A_2, A_3 and we can substitute it with a binary variable $y(A_1, A_2, A_3)$ that assumes certain values in correspondence to certain combination of values of $w(A_1, A_2)$ and $w(A_3, A_2)$. We want this variable assumes value 1 only if $w(A_1, A_2) = 1$ and $w(A_3, A_2) = 0$, which means that A_1 passes over A_2 and A_2 passes over A_3 in their conflict-column(s) respectively. We have to impose that this variable is 0 in correspondence to the three other combinations of values of w ($w(A_1, A_2) = 1, w(A_3, A_2) = 1$; $w(A_1, A_2) = 0, w(A_3, A_2) = 0$; $w(A_1, A_2) = 0, w(A_3, A_2) = 1$). In this way, we are imposing that A_1 has to perform at least 2 vertical moves if it passes above A_2 over their conflict column(s) and A_2 passes above A_3 over their conflict column(s). The following constraints guarantee that binary variable $y(A_1, A_2, A_3)$ assumes the values we want:

$$\begin{cases} y(A_1, A_2, A_3) \geqslant w(A_1, A_2) - w(A_3, A_2) \\ y(A_1, A_2, A_3) \leqslant w(A_1, A_2) \\ y(A_1, A_2, A_3) \leqslant 1 - w(A_3, A_2) \end{cases} \tag{68}$$

Let's show what happens in each case:

- $\begin{cases} w(A_1, A_2) = 1 \\ w(A_3, A_2) = 0 \end{cases}$ $\begin{cases} y(A_1, A_2, A_3) \geqslant 1 \\ y(A_1, A_2, A_3) \leqslant 1 \\ y(A_1, A_2, A_3) \leqslant 1 \end{cases}$ $\rightarrow \quad y(A_1, A_2, A_3) = 1$
- $\begin{cases} w(A_1, A_2) = 1 \\ w(A_3, A_2) = 1 \end{cases}$ $\begin{cases} y(A_1, A_2, A_3) \geqslant 0 \\ y(A_1, A_2, A_3) \leqslant 1 \\ y(A_1, A_2, A_3) \leqslant 0 \end{cases}$ $\rightarrow \quad y(A_1, A_2, A_3) = 0$
- $\begin{cases} w(A_1, A_2) = 0 \\ w(A_3, A_2) = 0 \end{cases}$ $\begin{cases} y(A_1, A_2, A_3) \geqslant 0 \\ y(A_1, A_2, A_3) \leqslant 0 \\ y(A_1, A_2, A_3) \leqslant 1 \end{cases}$ $\rightarrow \quad y(A_1, A_2, A_3) = 0$
- $\begin{cases} w(A_1, A_2) = 0 \\ w(A_3, A_2) = 1 \end{cases}$ $\begin{cases} y(A_1, A_2, A_3) \geqslant -1 \\ y(A_1, A_2, A_3) \leqslant 0 \\ y(A_1, A_2, A_3) \leqslant 0 \end{cases}$ $\rightarrow \quad y(A_1, A_2, A_3) = 0$

The third term $w(A_1, A_2)(1-w(A_3, A_2))w(A_3, A_4)$ depends on A_1, A_2, A_3, A_4 and we can substitute it with another binary variable $y(A_1, A_2, A_3, A_4)$ that assumes certain values in correspondence to specific values of $w(A_1, A_2)$, $w(A_3, A_2)$ and $w(A_3, A_4)$. We want variable $y(A_1, A_2, A_3, A_4)$ to assume value 1 only if $w(A_1, A_2) = 1$, $w(A_3, A_2) = 0$ and $w(A_3, A_4) = 1$, which means that A_1 passes above A_2, A_2 passes above A_3 and A_3 passes above A_4 in their conflict-columns respectively. This variable has to assume value 0 in correspondence of the other 7 combinations of value of w-variables because in these situations vehicle A_1 cannot be forced to move early in the grid.
Variable $y(A_1, A_2, A_3, A_4)$ is such that:

$$\begin{cases} y(A_1, A_2, A_3, A_4) \leqslant w(A_1, A_2) \\ y(A_1, A_2, A_3, A_4) \leqslant 1 - w(A_3, A_2) \\ y(A_1, A_2, A_3, A_4) \leqslant w(A_3, A_4) \\ y(A_1, A_2, A_3, A_4) \geqslant w(A_1, A_2) - 2w(A_3, A_2) - (1 - w(A_3, A_4)) \end{cases} \tag{69}$$

Let's show what happens in each case:

- $\begin{cases} w(A_1, A_2) = 1 \\ w(A_3, A_2) = 0 \\ w(A_3, A_4) = 1 \end{cases} \quad \begin{cases} y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \geqslant 1 \end{cases} \quad \rightarrow \quad y(A_1, A_2, A_3, A_4) = 1$
- $\begin{cases} w(A_1, A_2) = 1 \\ w(A_3, A_2) = 1 \\ w(A_3, A_4) = 1 \end{cases} \quad \begin{cases} y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \geqslant -1 \end{cases} \quad \rightarrow \quad y(A_1, A_2, A_3, A_4) = 0$
- $\begin{cases} w(A_1, A_2) = 1 \\ w(A_3, A_2) = 1 \\ w(A_3, A_4) = 0 \end{cases} \quad \begin{cases} y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \geqslant -2 \end{cases} \quad \rightarrow \quad y(A_1, A_2, A_3, A_4) = 0$
- $\begin{cases} w(A_1, A_2) = 1 \\ w(A_3, A_2) = 0 \\ w(A_3, A_4) = 0 \end{cases} \quad \begin{cases} y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \geqslant 0 \end{cases} \quad \rightarrow \quad y(A_1, A_2, A_3, A_4) = 0$
- $\begin{cases} w(A_1, A_2) = 0 \\ w(A_3, A_2) = 0 \\ w(A_3, A_4) = 0 \end{cases} \quad \begin{cases} y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \geqslant -1 \end{cases} \quad y(A_1, A_2, A_3, A_4) = 0$
- $\begin{cases} w(A_1, A_2) = 0 \\ w(A_3, A_2) = 0 \\ w(A_3, A_4) = 1 \end{cases} \quad \begin{cases} y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \geqslant 0 \end{cases} \quad y(A_1, A_2, A_3, A_4) = 0$
- $\begin{cases} w(A_1, A_2) = 0 \\ w(A_3, A_2) = 1 \\ w(A_3, A_4) = 0 \end{cases} \quad \begin{cases} y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \geqslant -3 \end{cases} \quad y(A_1, A_2, A_3, A_4) = 0$
- $\begin{cases} w(A_1, A_2) = 0 \\ w(A_3, A_2) = 1 \\ w(A_3, A_4) = 1 \end{cases} \quad \begin{cases} y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \leqslant 0 \\ y(A_1, A_2, A_3, A_4) \leqslant 1 \\ y(A_1, A_2, A_3, A_4) \geqslant -2 \end{cases} \quad y(A_1, A_2, A_3, A_4) = 0$

Using these new binary variables, we can rewrite inequalities (66) and (67) as follow:

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)-1} v^{A_1}(i) \geqslant w(A_1,A_2) + y(A_1,A_2,A_3) + y(A_1,A_2,A_3,A_4) \tag{70}$$

$$\sum_{i=\alpha(A_1)}^{c(A_1,A_2)} v^{A_1}(i) \geqslant w(A_1,A_2) + y(A_1,A_2,A_3) + y(A_1,A_2,A_3,A_4) \tag{71}$$

Adding to the model inequalities (70) and (71) together with constraints (68) and (69), we obtain another valid inequality.

Proof. We prove inequality (66). The RHS can assume only values 0,1,2,3 since it is the sum of 3 terms which can assume only values 1 and 0.
If the RHS takes value 0, the inequality is trivial.
If the RHS takes value 1, it must be true that $w(A_1,A_2)=1$ and the other two terms are equal to 0. Notice that it is not possible that the second or the third term are equal to 1, while the first term is equal to zero because $w(A_1,A_2)=0$ implies that everything is 0. The condition $w(A_1,A_2)=1$ means that A_1 passes above A_2 over their conflict column, so A_1 has to perform at least 1 vertical move in columns between $\alpha(A_1)$ and $c(A_1,A_2)-1$.
If the RHS takes value 2, it must be true that $w(A_1,A_2)=1$, $w(A_3,A_2)=0$ and $w(A_3,A_4)=0$ [Indeed, variable $w(A_1,A_2)$ must be equal to 1, otherwise all the terms vanish and the RHS would be equal to 0. Variable $w(A_3,A_2)$ must be equal to 0, otherwise the second and the third terms vanish and the RHS would be equal to 1. Provided that $w(A_1,A_2)=1$ and $w(A_3,A_2)=0$, the third variable $w(A_3,A_4)$ must be equal to 0 otherwise the third term would be equal to 1 and the RHS would assume value 3 instead of 2.]. This means that A_1 passes above A_2, A_2 passes above A_3 and A_3 passes under A_4 in their conflict-columns respectively. As consequence of their position, A_2 performs 1 vertical step (forced by A_3) before reaching the conflict column with A_1, so A_1 has to perform at least 2 vertical steps in order to pass above A_2 over their conflict column. This means exactly that A_1 has to perform at least 2 vertical moves in columns between $\alpha(A_1)$ and $c(A_1,A_2)-1$.
If the RHS takes value 3, it holds that $w(A_1,A_2)=1$, $w(A_3,A_2)=0$ and $w(A_3,A_4)=1$ [The reasoning is like above]. This means that A_1 passes above A_2, A_2 passes above A_3 and A_3 passes above A_4 in their conflict-columns respectively. As consequence of the starting-positions of the vehicles, it happens that A_3 performs one vertical step (forced by A_4) before

meeting A_2 and A_2 performs 2 vertical steps (forced by A_3) before meeting A_1. Consequently, when A_2 reaches $c(A_1, A_2) + 1$ it is already at level 2. This way, A_1 is forced to perform at least 3 vertical steps before reaching $c(A_1, A_2)$. ■

Proof. The proof of inequality (67) is similar. Provided the RHS can assume only value 0,1,2, and 3, we develop each case like before. ■

Example 105 *It's hard to find an example because these structures are very particular and the only case in which we could see the validity of this inequality is the one in which there is a 4-vehicle chain of conflicts and an optimal pattern is such that vehicle A_1 passes above vehicle A_2, which passes above A_3, which passes above A_4 respectively in their conflict columns.*

7 Computational Results

In this final section, we show some results about the inequalities that we have previously given: we would like to understand in which way our inequalities affect the formulation of model C. All the information provided in the following part is based on the resolution of different instances. We have generated ten random instances per size using a small script for Python Numpy (see Appendix 1). In all the tests, vector α is an ordered vector (for example, if $n = 5$ then $\alpha = (1, 2, 3, 4, 5)$) and vector ω is obtained by shuffling the elements of α. For these tests we have used software AMPL: both CPLEX 12.8.0.0 and also ILOG AMPL.

7.1 Continuous Relaxation

Firstly, we focus on the resolution of the continuous relaxation. In particular, we want to see how much the value of the objective function of the continuous relaxation gets close to the value of the objective function of the integer problem, thanks to the new inequalities. We expect to see that the value of z of the continuous relaxation increases together with the insertion of the inequalities. Let us consider Table 1, containing the average values of z:

Table 1: Average value of z

	ILP	**CR**	**CR+1,2**	**CR+1-3**	**CR+1-4**	**CR+1-6**	**CR+1-7**
5 vehicles	1.20	0.00	1.00	1.00	1.15	1.15	1.15
10 vehicles	1.00	0.00	1.00	1.00	1.00	1.00	1.00
15 vehicles	1.80	0.00	1.00	1.00	1.50	1.50	1.50
20 vehicles	1.80	0.00	1.00	1.00	1.50	1.50	1.50
30 vehicles	1.80	0.00	1.00	1.00	1.50	1.50	1.50
40 vehicles	2.20	0.00	1.00	1.00	1.60	1.61	1.61
Total	1.63	0.00	1.00	1.00	1.43	1.44	1.44

This Table shows the average values of z for instances containing different number of vehicles (rows of the table) and for different formulations (columns of the table). Indeed, the first column contains the values of z obtained from the resolution of the integer LP problem (ILP), the second column contains the values of z obtained from the resolution of the continuous relaxation (CR) without new inequalities. The third column contains the values of z obtained from the continuous relaxation with new inequalities of the first and the second set (CR+1,2). In the fourth column we add to the continuous relaxation of the previous column also the inequalities of the third set (CR+1-3) and so on.
We have summarized these values in Figure 18:

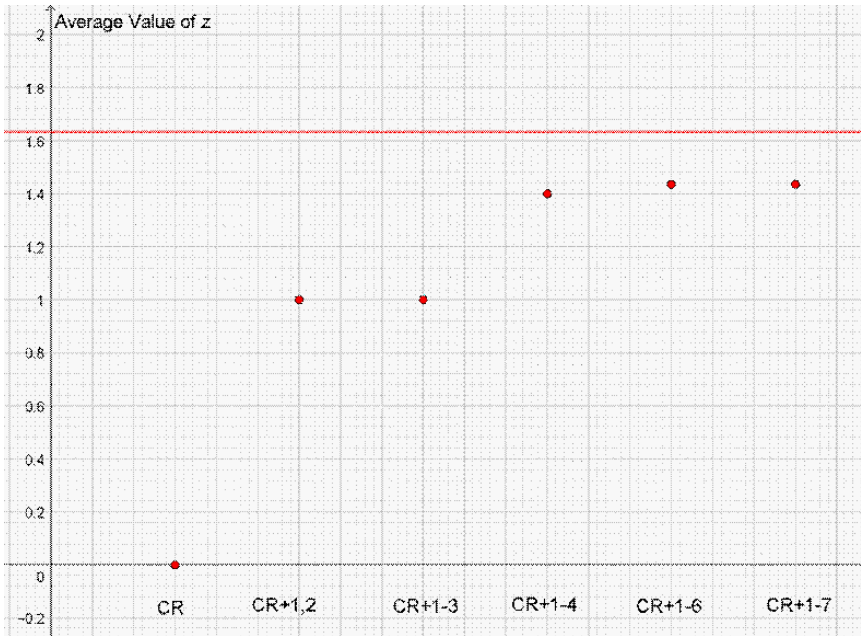

Figure 18: Average value of z (continuous relaxation)

In the graph, we can see the average value of z of the integer problems (red line) and the average values of z of the continuous relaxations (red points). The value of z obtained from CR is always zero, as expected. In general, we see that these inequalities make the value of z of the continuous relaxation grow and get quite close to the value provided by the resolution of the integer problems.
Regarding inequalities of the first and the second set, we can immediately

notice that they are similar and, in particular, that inequality $z \geqslant \delta$ dominates all inequalities $z \geqslant w(p,q) \ \ \forall (p,q) \in C$ since $w(p,q) \leqslant 1 \ \ \forall (p,q) \in C$ and if one variable w is different from 0, automatically, $\delta = 1$. This is the reason why we have tested them together. Moreover, when we have to decide if to add or not these inequalities to the implementation of model C, obviously it is sufficient to add only $z \geqslant \delta$. Even if this inequality appears very simple, it can be used to cut away the solution $z = 0$ from the feasible region of the continuous relaxation of any instance. However, it is not the only one that plays this role because it could be possible that the continuous relaxation provides a solution different from $z = 0$ also when inequality $z \geqslant \delta$ is not present in the implementation (but other new inequalities must be present).

Inequalities of the third set do not seem to produce an increase in the value of z but usually they produce a change of the optimal solution. We can deduce that these inequalities only make the optimal solution move to another vertex of the polyhedron.

Remind that inequalities of Section 6.4 and 6.5 are just alternative formulations of the same property. Obviously, if we want to add them to the model it is sufficient to choose only one set and we have worked with the inequalities of section 6.4. Adding to the model inequalities up to the fourth set, the value of z of the continuous relaxations gets close to its effective value as a percentage of 87.7%.

Inequalities of the sixth and the seventh set, do not cause a significative improvement in the value of z, which now gets close to its effective value as a percentage of 87.8% (only 0.1% more then the previous case).

7.2 Integer Problem

A second topic we investigate is the time of resolution of integer model. In practical cases, we need to solve integer instances and so we want to see how the insertion of these valid inequalities affects the time we need to solve the instances. Let us consider Table 2:

Table 2: Average time of resolution of integer instances (in second)

	ILP	**ILP+1,2**	**ILP+1-3**	**ILP+1-4**	**ILP+1-6**	**ILP+1-7**
5 vehicles	0.03	0.01	0.02	0.01	0.01	0.01
10 vehicles	0.03	0.02	0.02	0.03	0.02	0.02
15 vehicles	0.06	0.06	0.04	0.04	0.07	0.07
20 vehicles	0.05	0.04	0.04	0.01	0.07	0.06
30 vehicles	0.03	0.04	0.02	0.03	0.03	0.16
40 vehicles	0.50	0.06	0.04	0.24	0.27	5.20
Total	0.12	0.04	0.03	0.06	0.08	0.92

This Table shows the average values of the resolution's time of integer instances. The first column regards the model without new inequalities (ILP), second column regards the model with inequalities of the first and the second set (ILP+1,2), third column regards the model with inequalities up to the third set (ILP+1-3) and so on..
We show in a graph the average trend of the time (Figure 19):

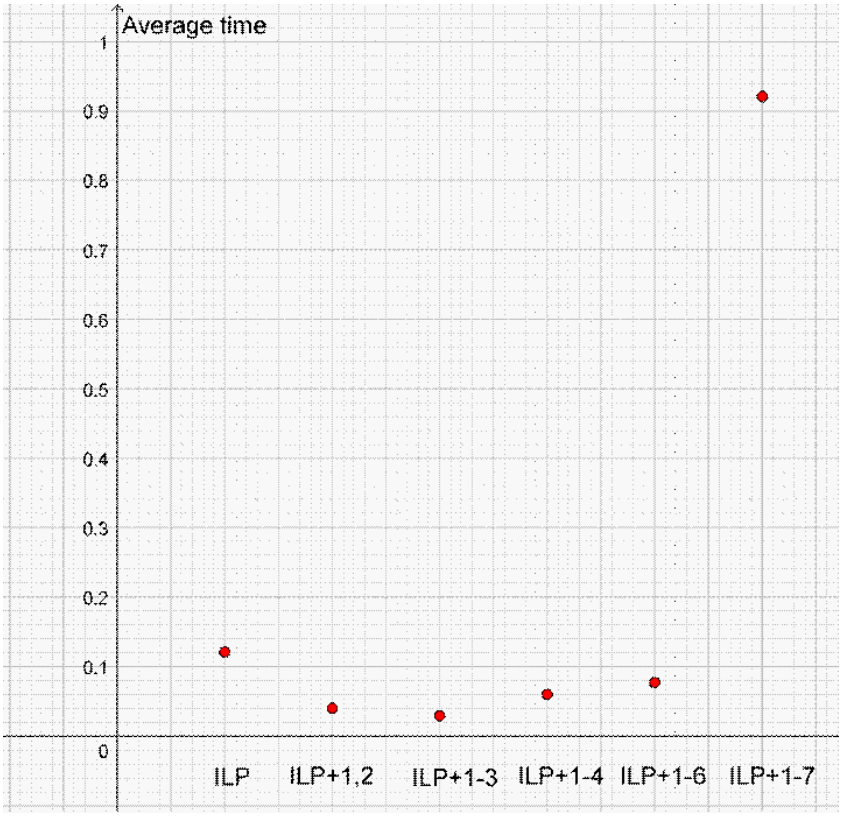

Figure 19: Average time of resolution of integer instances

We can see that, after the insertion of the first three sets of inequalities the time of resolution decreases, then it grows very slowly up to the insertion of the sixth sets of inequalities remaining under the level of the cases without inequalities. The seventh set makes the time blow up, due to the huge presence of variables and inequalities.
This computational study does not expect to be exhaustive in general. However, this is a good starting point for future studies.

8 Conclusion

This work concerns Integer Linear Programming Models and their application to the FQRP. We introduced 3 integer linear programming models to solve the FQRP and focused on the search of valid cuts for formulation C. We introduced 7 sets of valid inequalities and proved their validity. In the end we developed a brief computational section that involves some practical and applicative observations about the use of this inequalities.
The realization of this work took 6 months, from April 2020 to September 2020. At the beginning, I had to get lots of informations in order to put myself in the right point of view to develop this work and so I started reading many journals and papers. Afterwards, I studied the models and wrote their implementations with AMPL. When I felt enough confident with the matter, I started searching for valid cuts. It was hard and challenging to understand which form an inequality like these should have. Some of the initial ideas were too easy or related only to easy instances (for example instances constituted only by 2 or 3 vehicles) instead of being applicable to every instance, independent on the number of vehicles involved.
In order to find the inequalities, I made hundreds of examples: every time the solution of the continuous relaxation was unfeasible, I tried to understand what was the "bug" of the pattern and to remove it with an inequality. The difficult part consisted in generalizing the rules that I noticed for one case to every case.
The ability of generalization and research of commonalities among things that appear different is typical for a mathematician and, thanks of this work, I tested it personally.

I would like to thank Prof. De Francesco and Prof. De Giovanni for their teachings, their time and their help.

Appendix 1-codes

Model A, file .mod

```
param n_col;
param n_row;
param n_veic;

set VEIC:=1..n_veic;  #total vehicles

param alpha{VEIC};  #starting position
param omega{VEIC};  #final position


#sets of vehicles
set S:= {k in VEIC : omega[k] = alpha[k]};
set R:= {k in VEIC : omega[k] >= alpha[k]} diff S;
set L:= {k in VEIC} diff (R union S);

#indexing sets
set J:=1..n_row-1;
set I:=1..n_col;
set ZERO:=0..0;
set SUP:=n_col+1..n_col+1;
set J0:=J union ZERO;

set Ir{k in R}:={i in I: i>=alpha[k] and i<=omega[k]};
set IrAMPL{k in R}:={i in I: i>=alpha[k]-1 and i<=omega[k]} union ZERO;

set Il{k in L}:={i in I: i<=alpha[k] and i>=omega[k]};
set IlAMPL{k in L}:={i in I: i<=alpha[k]+1 and i>=omega[k]} union SUP;

#binary variables
var x_vert{i in R,Ir[i],J0} binary;
var x_hor{i in R, IrAMPL[i],J} binary;

var y_vert{i in L,Il[i],J0} binary;
var y_hor{i in L, IlAMPL[i],J} binary;

#variable to be minimized
var z integer >=0;

minimize f: z;  #objective

#initialization constraints
#R
set J_start:=2..n_row-1; #new set
s.t. v1{k in R, j in J_start}:x_hor[k,alpha[k]-1,j]=0;
s.t. v2{k in R, j in J_start}:x_hor[k,0,j]=0;
s.t. v3{k in R}:x_hor[k,alpha[k]-1,1]=1;
s.t. v4{k in R}:x_hor[k,0,1]=1;
s.t. v5{k in R, j in J}:x_hor[k,omega[k],j]=0;
```

```
s.t. v6{k in R, i in Ir[k]}:x_vert[k,i,0]=0;
#L
s.t. v7{k in L, j in J_start}:y_hor[k,alpha[k]+1,j]=0;
s.t. v8{k in L, j in J}:y_hor[k,omega[k],j]=0;
s.t. v9{k in L}:y_hor[k,alpha[k]+1,1]=1;
s.t. v10{k in L, i in Il[k]}:y_vert[k,i,0]=0;

#constraints over R
s.t. vinc_startR{k in R}: x_vert[k,alpha[k],1]+x_hor[k,alpha[k],1]=1;
s.t. vinc_endR{k in R}: x_vert[k,omega[k],n_row-1]=1;
s.t. vinc_contR{k in R, i in Ir[k], j in J}:
x_hor[k,i-1,j]+x_vert[k,i,j-1]=x_vert[k,i,j]+x_hor[k,i,j];

#constraints over L
s.t. vinc_startL{k in L}: y_vert[k,alpha[k],1]+y_hor[k,alpha[k],1]=1;
s.t. vinc_endL{k in L}: y_vert[k,omega[k],n_row-1]=1;
s.t. vinc_contL{k in L, i in Il[k], j in J}:
y_hor[k,i+1,j]+y_vert[k,i,j-1]=y_vert[k,i,j]+y_hor[k,i,j];

#constraints to avoid node-conflits
set Confl_nod:={r in R, l in L: alpha[r]<=alpha[l] and
round((alpha[r]+alpha[l])/2)=(alpha[r]+alpha[l])/2 and
omega[r]>=(alpha[r]+alpha[l])/2 and omega[l]<=(alpha[r]+alpha[l])/2};
set col_mezzo{(m,n) in Confl_nod}:={(alpha[m]+alpha[n])/2};
s.t. non_confl_nod{(m,n) in Confl_nod, i in col_mezzo[m,n], j in J}:
x_hor[m,i-1,j]+x_vert[m,i,j-1]+y_hor[n,i+1,j]+y_vert[n,i,j-1]<=1;

#constraints to avoid arc-conflicts
set Confl_arc:={r in R, l in L:
alpha[r]<=alpha[l]-1 and
(alpha[r]+alpha[l])/2>=floor((alpha[r]+alpha[l])/2)+0.1 and
omega[r]>=floor((alpha[r]+alpha[l])/2) and
omega[l]<=floor((alpha[r]+alpha[l])/2)};
set col_arc{(m,n) in Confl_arc}:={floor((alpha[m]+alpha[n])/2)};
s.t. non_confl_arc{(m,n) in Confl_arc, i in col_arc[m,n], j in J}:
x_hor[m,i,j]+y_hor[n,i+1,j]<=1;

#links between z and x/y_hor
s.t. z1{k in R, i in IrAMPL[k], j in J}: z>=j*x_hor[k,i,j];
s.t. z2{k in L, i in IlAMPL[k], j in J}: z>=j*y_hor[k,i,j];
```

Model B, file .mod

```
param n_row;
param n_col;
param n_veic;

set VEIC:=1..n_veic;
param alpha{VEIC};  #starting position
param omega{VEIC};  #final position

#sets of vehicles
set S:= {k in VEIC : omega[k] = alpha[k]};
set R:= {k in VEIC : omega[k] >= alpha[k]} diff S;
set L:= {k in VEIC} diff (R union S);

#
set J:=1..n_row-1;
set I:= 1..n_col;
set P:= R union L;
set Ir{k in R}:={i in I: i>=alpha[k] and i<=omega[k]};
set Il{k in L}:={i in I: i>=omega[k] and i<=alpha[k]};
set Ip{k in P}:={i in I: i>=min(omega[k], alpha[k]) and i<= max(omega[k]
,alpha[k])};

#variables vertical movements
var v{k in P, i in Ip[k], j in J} binary;

var z>=0 integer;  #to be minimized

minimize f: z;  #objective

s.t. c1{k in P}: sum{i in Ip[k]} sum{j in J} v[k,i,j]=z;
s.t. c2{k in P}: sum{j in J} v[k,omega[k],j]>=1;
s.t. c3{k in P, j in J}: sum{i in Ip[k]} v[k,i,j]<=1;

set J_rid:=2..n_row-1;
set I_5{k in P, i in Ip[k]}:={ii in I: ii>= min(alpha[k],i) and
ii<=max(alpha[k],i)};
s.t. c4{k in P, i in Ip[k], j in J_rid}: v[k,i,j]<= sum{ii in I_5[k,i]}
v[k,ii,j-1];

#conflicts
set Confl:={r in R, l in L:alpha[r]<=alpha[l]-1 and
omega[r]>=ceil((alpha[r]+alpha[l])/2) and
omega[l]<=floor((alpha[r]+alpha[l])/2)};
var delta_d{(r,l) in Confl, j in J_rid} integer;
var delta_u{(r,l) in Confl, j in J} integer;

s.t. c5{(r,l) in Confl}:delta_u[r,l,1]>=1;
s.t. c6{(r,l) in Confl, j in J_rid}:delta_d[r,l,j]+delta_u[r,l,j]>=1;

var aus_d{(r,l) in Confl, j in J_rid} binary;
var aus_u{(r,l) in Confl, j in J} binary;
#
```

```
set I_6{(r,l) in Confl}:={i in I: i>=alpha[r] and
i<=floor((alpha[r]+alpha[l])/2)};
set I_6p{(r,l) in Confl}:={i in I: i<=alpha[l] and
i>=ceil((alpha[r]+alpha[l])/2)};
set J_r{j in J_rid}:={i in J: i>=1 and i<=j};

s.t. c7{(r,l) in Confl, j in J_rid}:delta_d[r,l,j]<=sum{i in I_6[r,l]}
sum{jj in J_r[j]} v[r,i,jj]- sum{ii in I_6p[r,l]} sum{jjj in J_r[j]}
v[l,ii,jjj] +n_row*aus_d[r,l,j];
s.t. c8{(r,l) in Confl, j in J_rid}:delta_d[r,l,j]<= -sum{i in I_6[r,l]}
sum{jj in J_r[j]} v[r,i,jj]+ sum{ii in I_6p[r,l]} sum{jjj in J_r[j]}
v[l,ii,jjj] +n_row*(1-aus_d[r,l,j]);

#
set J_rr{j in J}:={i in J: i>=1 and i<=j};
set I_7{(r,l) in Confl}:={i in I: i<=omega[r] and
i>=ceil((alpha[r]+alpha[l])/2)};
set I_7p{(r,l) in Confl}:={i in I: i>=omega[l] and
i<=floor((alpha[r]+alpha[l])/2)};
s.t. c9{(r,l) in Confl, j in J}:delta_u[r,l,j]<=sum{i in I_7[r,l]}
sum{jj in J_rr[j]} v[r,i,jj]- sum{ii in I_7p[r,l]} sum{jjj in J_rr[j]}
v[l,ii,jjj] +n_row*aus_u[r,l,j];
s.t. c10{(r,l) in Confl, j in J}:delta_u[r,l,j]<= -sum{i in I_7[r,l]}
sum{jj in J_rr[j]} v[r,i,jj]+ sum{ii in I_7p[r,l]} sum{jjj in J_rr[j]}
v[l,ii,jjj] +n_row*(1-aus_u[r,l,j]);
```

Model C, file .mod

```
param n_row;
param n_col;
param n_veic;

set VEIC:=1..n_veic;
param alpha{VEIC};  #starting positions
param omega{VEIC};  #final positions

#sets of vehicles
set S:= {k in VEIC : omega[k] = alpha[k]};
set R:= {k in VEIC : omega[k] >= alpha[k]} diff S;
set L:= {k in VEIC} diff (R union S);

#
set J:=1..n_row-1;
set I:= 1..n_col;
set P:= R union L;
set Ir{k in R}:={i in I: i>=alpha[k] and i<=omega[k]};
set Il{k in L}:={i in I: i>=omega[k] and i<=alpha[k]};
set Ip{k in P}:={i in I: i>=min(omega[k], alpha[k]) and i<= max(omega[k]
,alpha[k])};

#variables vertical movements
var v{k in P, i in Ip[k]}>=0 integer;

var z>=0 integer;  #to be minimized

set Confl:={r in R, l in L: alpha[r]<=alpha[l]-1 and
omega[r]>=ceil((alpha[r]+alpha[l])/2) and
omega[l]<=floor((alpha[r]+alpha[l])/2)};
set C_even:={(r,l) in Confl:
floor((alpha[r]+alpha[l])/2)=ceil((alpha[r]+alpha[l])/2)};  #node-confl
set C_odd:=Confl diff C_even;  #arc-confl

var w{(r,l) in Confl} binary;


minimize f: z;  #objective
s.t. summ{k in P}: sum{i in Ip[k]}v[k,i]=z;

set Iodd1{(r,l) in C_odd}:={i in I: i>=alpha[r] and
i<=floor((alpha[r]+alpha[l])/2)};
set Iodd2{(r,l) in C_odd}:={i in I: i>=ceil((alpha[r]+alpha[l])/2) and
i<=alpha[l]};
s.t. v1{(r,l) in C_odd}:sum{i in Iodd1[r,l]}v[r,i]<=sum{ii in
Iodd2[r,l]}v[l,ii]-1+n_row*w[r,l];
s.t. v2{(r,l) in C_odd}:sum{i in Iodd2[r,l]}v[l,i]<=sum{ii in
Iodd1[r,l]}v[r,ii]-1+n_row*(1-w[r,l]);

set Ieven1{(r,l) in C_even}:={i in I: i>=alpha[r] and
i<=(alpha[r]+alpha[l])/2};
```

```
set Ieven2{(r,l) in C_even}:={i in I: i>=(alpha[r]+alpha[l])/2+1 and
i<=alpha[l]};
set Ieven4{(r,l) in C_even}:={i in I: i>=alpha[r] and
i<=(alpha[r]+alpha[l])/2-1};
set Ieven3{(r,l) in C_even}:={i in I: i>=(alpha[r]+alpha[l])/2 and
i<=alpha[l]};

s.t. v3{(r,l) in C_even}:sum{i in Ieven1[r,l]}v[r,i]<=sum{ii in
Ieven2[r,l]}v[l,ii]-1+n_row*w[r,l];
s.t. v4{(r,l) in C_even}:sum{i in Ieven3[r,l]}v[l,i]<=sum{ii in
Ieven4[r,l]}v[r,ii]-1+n_row*(1-w[r,l]);


#############################################################################
#INEQUALITY 1
s.t. nuovo2{(r,l) in Confl}:z>=w[r,l];

#INEQUALITY 2
param delta:= max(ceil(card(Confl)/(n_veic*card(Confl)+1)),0);
s.t. nuovo1{k in P}:sum{i in Ip[k]}v[k,i]>=delta;

#INEQUALITY 3
s.t. nuovo3{(r,l) in
C_even}:v[r,(alpha[r]+alpha[l])/2]+v[l,(alpha[r]+alpha[l])/2]<=
z-1;

#INEQUALITY 4
#odd
set Ind1{(r,l) in C_odd}:={i in I: i<=floor((alpha[r]+alpha[l])/2) and
i>=alpha[r]};
set Ind2{(r,l) in C_odd}:={i in I: i<=floor((alpha[r]+alpha[l])/2) and
i>=omega[l]};
set Ind3{(r,l) in C_odd}:={i in I: i>=ceil((alpha[r]+alpha[l])/2) and
i<=omega[r]};
set Ind4{(r,l) in C_odd}:={i in I: i>=ceil((alpha[r]+alpha[l])/2) and
i<=alpha[l]};

s.t. limit_col1{(r,l) in C_odd}:sum{i in Ind1[r,l]}v[r,i]<=w[r,l]+z-1;
s.t. limit_col2{(r,l) in C_odd}:sum{i in Ind2[r,l]}v[l,i]<=w[r,l]+z-1;
s.t. limit_col3{(r,l) in C_odd}:sum{i in Ind3[r,l]}v[r,i]<=(1-w[r,l])+z-
1;
s.t. limit_col4{(r,l) in C_odd}:sum{i in Ind4[r,l]}v[l,i]<=(1-w[r,l])+z-
1;

#even
set Ind5{(r,l) in C_even}:={i in I: i<=(alpha[r]+alpha[l])/2 and
i>=alpha[r]};
set Ind6{(r,l) in C_even}:={i in I: i<=(alpha[r]+alpha[l])/2 and
i>=omega[l]};
set Ind7{(r,l) in C_even}:={i in I: i>=(alpha[r]+alpha[l])/2 and
i<=omega[r]};
set Ind8{(r,l) in C_even}:={i in I: i>=(alpha[r]+alpha[l])/2 and
i<=alpha[l]};

s.t. limit_col5{(r,l) in C_even}:sum{i in Ind5[r,l]}v[r,i]<=w[r,l]+z-1;
```

```
s.t. limit_col6{(r,l) in C_even}:sum{i in Ind6[r,l]}v[l,i]<=w[r,l]+z-1;
s.t. limit_col7{(r,l) in C_even}:sum{i in Ind7[r,l]}v[r,i]<=(1-w[r,l])+
z-1;
s.t. limit_col8{(r,l) in C_even}:sum{i in Ind8[r,l]}v[l,i]<=(1-w[r,l])+
z-1;

#INEQUALITY 5
#s.t. nuovo4{(r,l) in C_odd}:sum{i in Ind1[r,l]}v[r,i]>=w[r,l];
#s.t. nuovo5{(r,l) in C_odd}:sum{i in Ind4[r,l]}v[l,i]>=1-w[r,l];

#s.t. nuovo6{(r,l) in C_even}:sum{i in Ieven4[r,l]}v[r,i]>=w[r,l];
#s.t. nuovo7{(r,l) in C_even}:sum{i in Ieven2[r,l]}v[l,i]>=1-w[r,l];

#INEQUALITY 6
set coppie{(r1,l1) in C_even}:={(r2,l2) in Confl: l1=l2 and r1!=r2 and
alpha[r2]>=max(1,alpha[r1]-2)};
set indsx{(r1,l1) in C_even, (r2,l2) in coppie[r1,l1]}:={i in I:
i>=alpha[r1] and i<=(alpha[r1]+alpha[l1])/2-1};
s.t. nuovo8{(r1,l1) in C_even, (r2,l2) in coppie[r1,l1]}:
sum{i in indsx[r1,l1,r2,l2]}v[r1,i]>=2*w[r1,l1]-w[r2,l2];

set coppie01{(r1,l1) in C_odd}:={(r2,l2) in Confl: l1=l2 and r1!=r2 and
alpha[r2]>=max(1,alpha[r1]-1)};

set indsx01{(r1,l1) in C_odd, (r2,l2) in coppie01[r1,l1]}
:={i in I: i>=alpha[r1] and i<=floor((alpha[r1]+alpha[l1])/2)};
s.t. nuovo9{(r1,l1) in C_odd, (r2,l2) in coppie01[r1,l1]}:
sum{i in indsx01[r1,l1,r2,l2]}v[r1,i]>=2*w[r1,l1]-w[r2,l2];


#INEQUALITY 7
#even even
set coppie1{(r1,l1) in C_even}:={(r2,l2) in C_even: l1=l2 and r1!=r2 and
alpha[r2]>=max(1,alpha[r1]-2)};
set catene3_1{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1]}:={(r3,l3) in
Confl: r3=r2 and l3!=l2 and alpha[l3]<=min(n_col,alpha[l2]+2)};
# new variables y
var y1_1{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1]}>=0 binary;
var y1_2{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}>=0 binary;
# linking constraints y1_1
s.t. vincy1_11{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}: y1_1[r1,l1,r2,l2]>=w[r1,l1]-w[r2,l2];
s.t. vincy1_12{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}: y1_1[r1,l1,r2,l2]<=w[r1,l1];
s.t. vincy1_13{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}: y1_1[r1,l1,r2,l2]<=1-w[r2,l2];
# linking constraints y1_2
s.t. vincy1_21{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}: y1_2[r1,l1,r2,l2,r3,l3]<=w[r1,l1];
s.t. vincy1_22{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}: y1_2[r1,l1,r2,l2,r3,l3]<=1-w[r2,l2];
s.t. vincy1_23{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}:y1_2[r1,l1,r2,l2,r3,l3]<=w[r3,l3];
```

```
s.t. vincy1_24{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}:y1_2[r1,l1,r2,l2,r3,l3]>=w[r1,l1]-2*w[r2,l2]-(1-
w[r3,l3]);

set indsx1{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}:={i in I: i>=alpha[r1] and
i<=((alpha[r1]+alpha[l1])/2)-1};
s.t. nuovo10{(r1,l1) in C_even, (r2,l2) in coppie1[r1,l1], (r3,l3) in
catene3_1[r1,l1,r2,l2]}:sum{i in indsx1[r1,l1,r2,l2,r3,l3]}v[r1,i]>=
w[r1,l1]+y1_1[r1,l1,r2,l2]+y1_2[r1,l1,r2,l2,r3,l3];

#even odd
set coppie2{(r1,l1) in C_even}:={(r2,l2) in C_odd: l1=l2 and r1!=r2 and
alpha[r2]>=max(1,alpha[r1]-2)};
set catene3_2{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1]}:={(r3,l3) in
Confl: r3=r2 and l3!=l2 and alpha[l3]<=min(n_col,alpha[l2]+1)};
# new variables y
var y2_1{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1]}>=0 binary;
var y2_2{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]}>=0 binary;
# linking constraints y2_1
s.t. vincy2_11{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]}:y2_1[r1,l1,r2,l2]>=w[r1,l1]-w[r2,l2];
s.t. vincy2_12{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]}:y2_1[r1,l1,r2,l2]<=w[r1,l1];
s.t. vincy2_13{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]}:y2_1[r1,l1,r2,l2]<=1-w[r2,l2];

# linking constraints y2_2
s.t. vincy2_21{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]}:y2_2[r1,l1,r2,l2,r3,l3]<=w[r1,l1];
s.t. vincy2_22{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]}:y2_2[r1,l1,r2,l2,r3,l3]<=1-w[r2,l2];
s.t. vincy2_23{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]}:y2_2[r1,l1,r2,l2,r3,l3]<=w[r3,l3];
s.t. vincy2_24{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]}:y2_2[r1,l1,r2,l2,r3,l3]>=w[r1,l1]-2*w[r2,l2]-(1-
w[r3,l3]);

set indsx2{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]} :={i in I: i>=alpha[r1] and
i<=((alpha[r1]+alpha[l1])/2)-1};
s.t. nuovo11{(r1,l1) in C_even, (r2,l2) in coppie2[r1,l1], (r3,l3) in
catene3_2[r1,l1,r2,l2]}:sum{i in indsx2[r1,l1,r2,l2,r3,l3]}v[r1,i]>=
w[r1,l1]+y2_1[r1,l1,r2,l2]+y2_2[r1,l1,r2,l2,r3,l3];

#odd even
set coppie3{(r1,l1) in C_odd}:={(r2,l2) in C_even: l1=l2 and r1!=r2 and
alpha[r2]>=max(1,alpha[r1]-1)};
set catene3_3{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1]}
:={(r3,l3) in Confl: r3=r2 and l3!=l2 and
alpha[l3]<=min(n_col,alpha[l2]+2)};
# new variables y
var y3_1{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1]}>=0 binary;
```

```
var y3_2{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}>=0 binary;
# linking constraints y3_1
s.t. vincy3_11{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}:y3_1[r1,l1,r2,l2]>=w[r1,l1]-w[r2,l2];
s.t. vincy3_12{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}:y3_1[r1,l1,r2,l2]<=w[r1,l1];
s.t. vincy3_13{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}:y3_1[r1,l1,r2,l2]<=1-w[r2,l2];
# linking constraints y3_2
s.t. vincy3_21{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}:y3_2[r1,l1,r2,l2,r3,l3]<=w[r1,l1];
s.t. vincy3_22{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}:y3_2[r1,l1,r2,l2,r3,l3]<=1-w[r2,l2];
s.t. vincy3_23{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}:y3_2[r1,l1,r2,l2,r3,l3]<=w[r3,l3];
s.t. vincy3_24{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}:y3_2[r1,l1,r2,l2,r3,l3]>=w[r1,l1]-2*w[r2,l2]-(1-
w[r3,l3]);

set indsx3{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}:={i in I: i>=alpha[r1] and
i<=floor((alpha[r1]+alpha[l1])/2)};
s.t. nuovo12{(r1,l1) in C_odd, (r2,l2) in coppie3[r1,l1], (r3,l3) in
catene3_3[r1,l1,r2,l2]}:sum{i in indsx3[r1,l1,r2,l2,r3,l3]}v[r1,i]>=
w[r1,l1]+y3_1[r1,l1,r2,l2]+y3_2[r1,l1,r2,l2,r3,l3];

#odd odd
set coppie4{(r1,l1) in C_odd}:={(r2,l2) in C_odd: l1=l2 and r1!=r2 and
alpha[r2]>=max(1,alpha[r1]-1)};
set catene3_4{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1]}
:={(r3,l3) in Confl: r3=r2 and l3!=l2 and
alpha[l3]<=min(n_col,alpha[l2]+1)};
# new variables y
var y4_1{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1]}>=0 binary;
var y4_2{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}>=0 binary;
# linking constraints y4_1
s.t. vincy4_11{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}:y4_1[r1,l1,r2,l2]>=w[r1,l1]-w[r2,l2];
s.t. vincy4_12{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}:y4_1[r1,l1,r2,l2]<=w[r1,l1];
s.t. vincy4_13{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}:y4_1[r1,l1,r2,l2]<=1-w[r2,l2];
# linking constraints y4_2
s.t. vincy4_21{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}:y4_2[r1,l1,r2,l2,r3,l3]<=w[r1,l1];
s.t. vincy4_22{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}:y4_2[r1,l1,r2,l2,r3,l3]<=1-w[r2,l2];
s.t. vincy4_23{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}:y4_2[r1,l1,r2,l2,r3,l3]<=w[r3,l3];
s.t. vincy4_24{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}:y4_2[r1,l1,r2,l2,r3,l3]>=w[r1,l1]-2*w[r2,l2]-(1-
w[r3,l3]);
```

```
set indsx4{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}:={i in I: i>=alpha[r1] and
i<=floor((alpha[r1]+alpha[l1])/2)};
s.t. nuovo13{(r1,l1) in C_odd, (r2,l2) in coppie4[r1,l1], (r3,l3) in
catene3_4[r1,l1,r2,l2]}:sum{i in indsx4[r1,l1,r2,l2,r3,l3]}v[r1,i]>=
w[r1,l1]+y4_1[r1,l1,r2,l2]+y4_2[r1,l1,r2,l2,r3,l3];
```

Model A-B-C, file .dat

```
param n_col:= ; # insert number of columns of the grid
param n_veic:= ; # insert number of vehicles (it coincides with the
number of columns)
param n_row:= ; # insert number of rows of the grid


param alpha:= ;
param omega := ; # insert the components of these two vectors
```

Model A-B-C, file .run

```
reset;

model namefile.mod;
data namefile.dat;

option solver cplex;
solve;

display z;
```

Script for Python Numpy (Shuffle)

```
import numpy as np;
import random;
from random import shuffle;

# ordered list representing vector α
vehicles=np.arange(1, ) # insert number of vehicles+1

np.random.shuffle(vehicles)
print(vehicles) #shuffled list representing vector ω
```

Appendix 2-draws

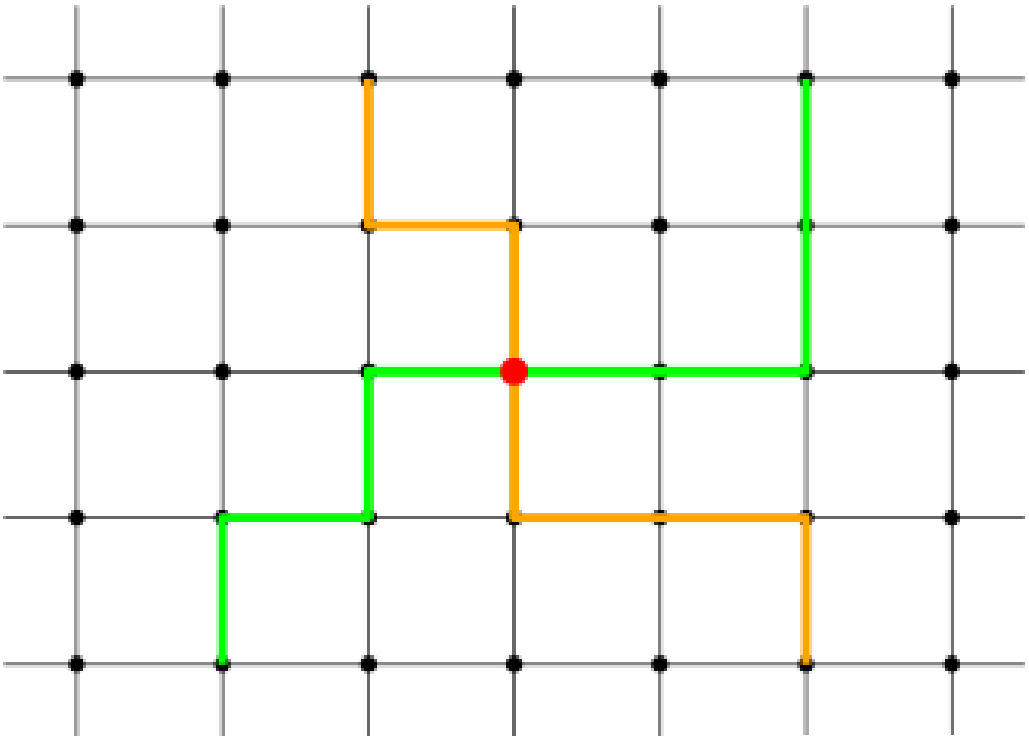

Figure 1: Node Conflict

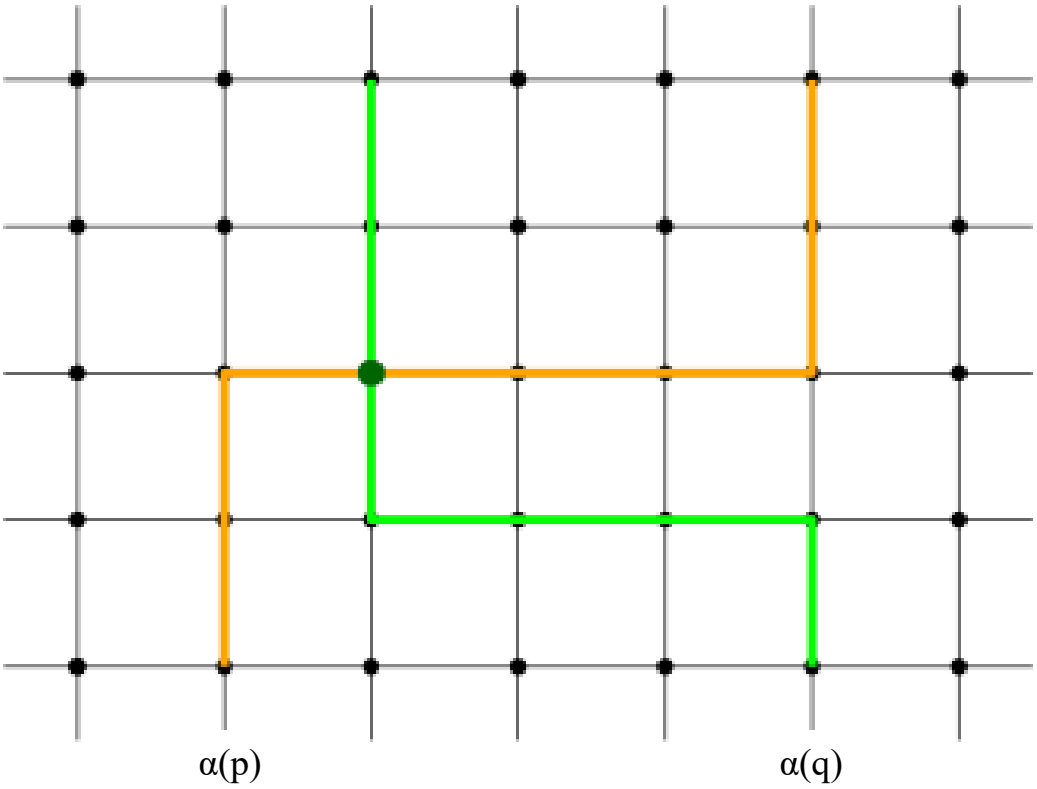

Figure 2: Intersection of 2 routes without Node-Conflict

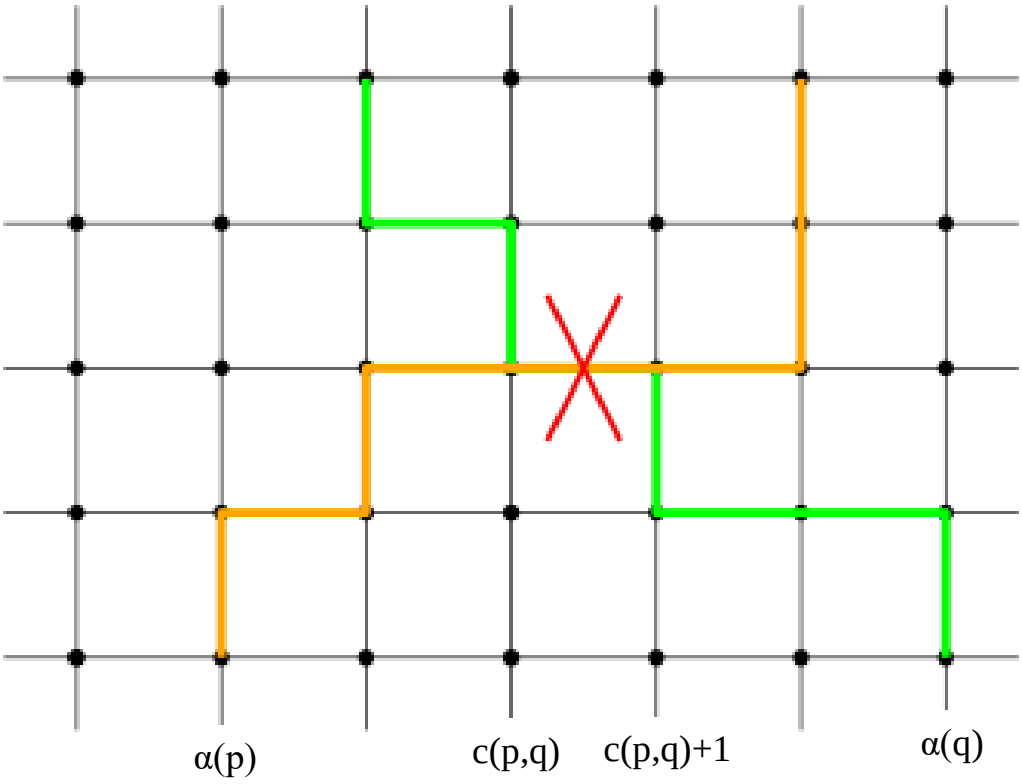

Figure 3: Arc-Conflict

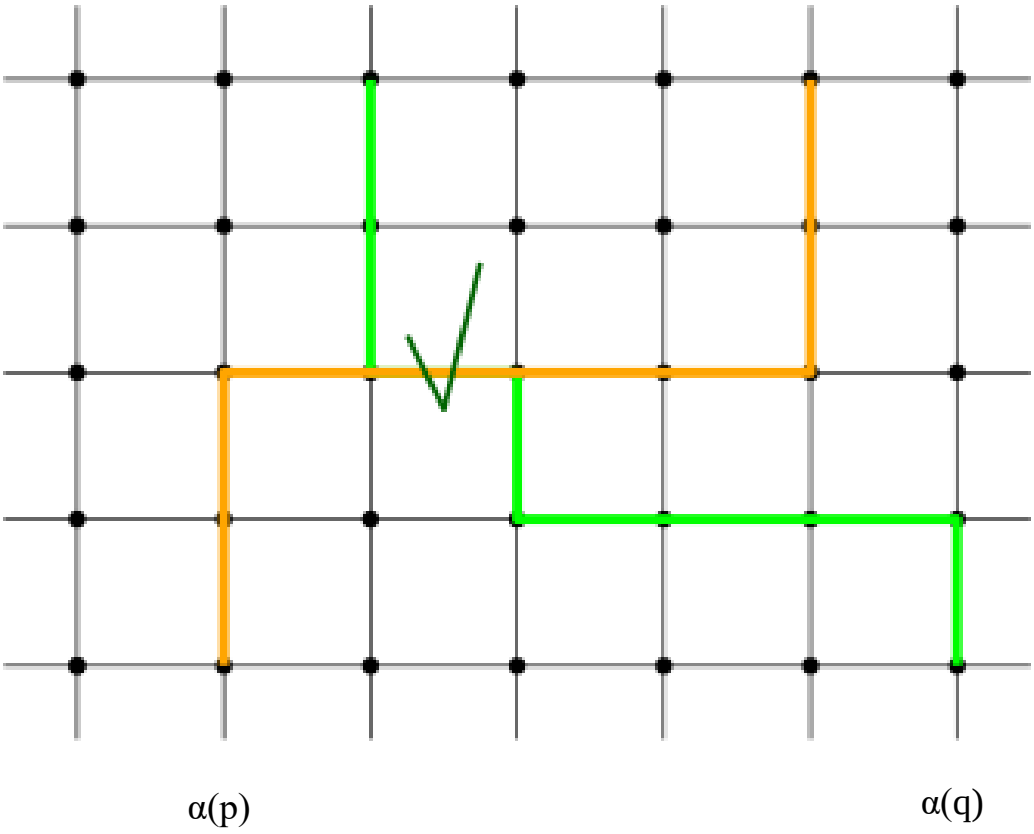

Figure 4: Intersection of 2 routes without Arc-Conflict

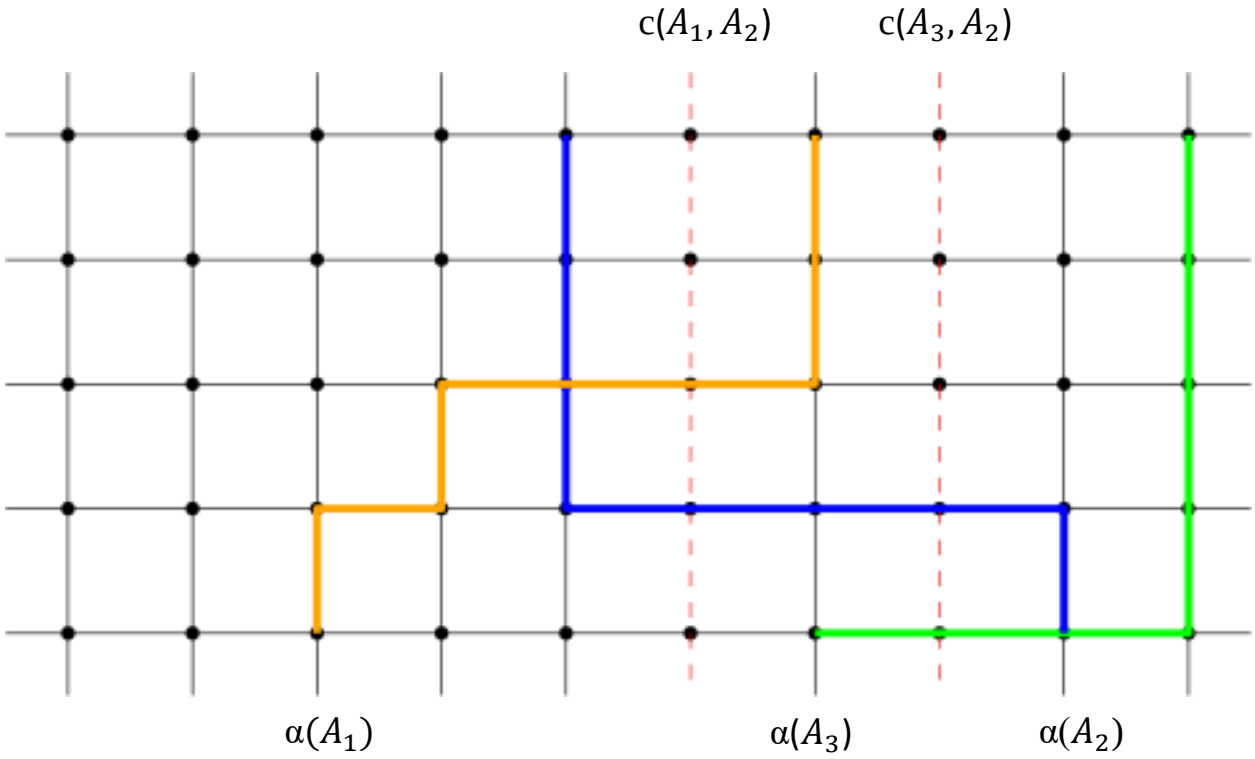

Figure 5: Generic 3-vehicle Conflict Chain

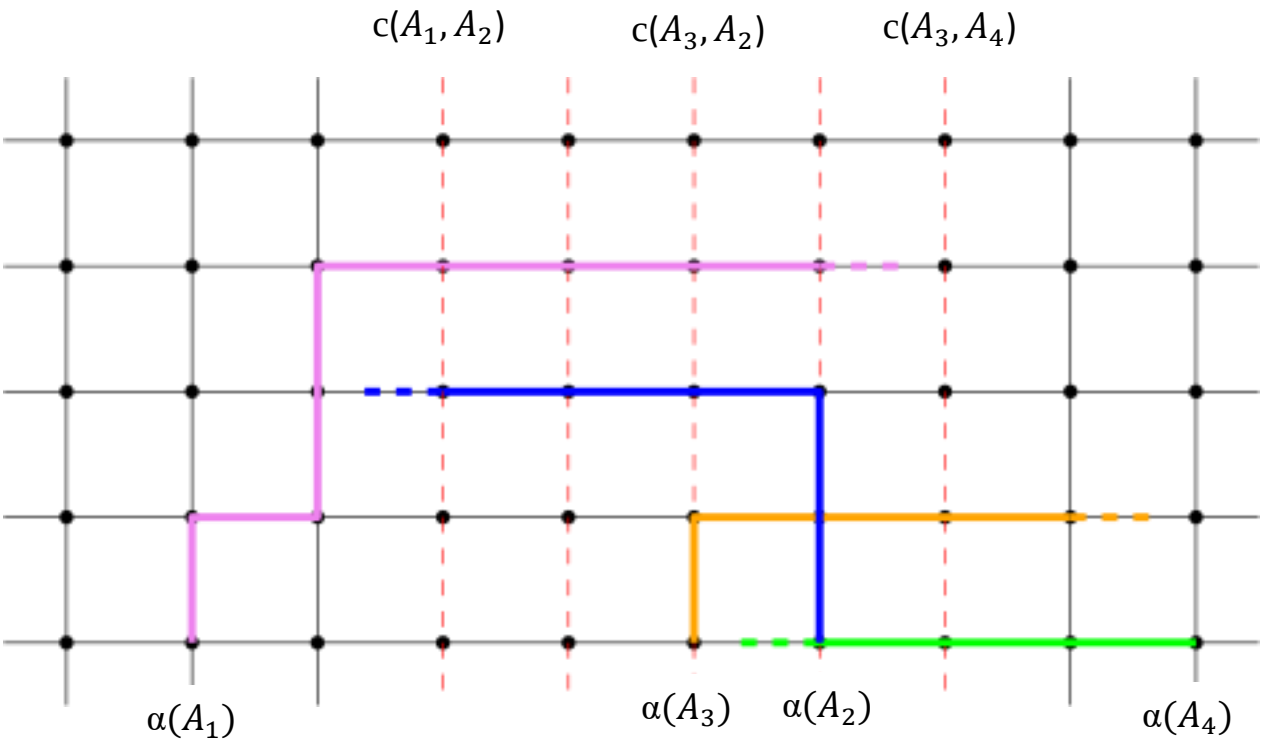

Figure 6: Generic 4-vehicle Conflict Chain

References

[1] G. Andreatta, L. De Giovanni, G. Salmaso, *Fleet Quickest Routing on Grids: a Polynomial Algorithm*, International Journal of Pure and Applied Mathematics, Vol.62 (2010)

[2] M. Cenci, M. Di Giacomo, F. Mason, *A Note on a Mixed Routing and Scheduling Problem on a Grid Graph*, Journal of the Operational Research Society, Vol.68 (2017), DOI: 10.1057/s41274-016-0152-9

[3] M. Cenci, M. Di Giacomo, F. Mason, *A Note on Solving the Fleet Quickest Routing Problem on a Grid Graph*, Central European Journal of Operations Research, Vol.28 (2020), https://doi.org/10.1007/s10100-019-00620-5

[4] http://www.math.unipd.it/~luigi/courses/ricop/ricop.html

[5] M. Di Summa, Note del corso Ricerca Operativa, Università degli Studi di Padova 2011-2012

[6] http:// www.ampl.com

[7] G. Andreatta, private communication

[8] L. De Giovanni, C. De Francesco, private communication