# Stochastic Vehicle Routing - a 2 Stage Approach

by

Riley Becker

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Waterloo, Ontario, Canada, 2020

## Abstract

In this paper, we consider the two-stage Capacitated Vehicle Routing Problem (2SCVRP), a stochastic variant of the Capacitated Vehicle Routing Problem (CVRP) proposed by Dantzig and Ramser in 1959. In the original CVRP, one or multiple vehicles start at a depot with a specified capacity of goods and serve customers given on a graph which have pre-specified demands. This is done with the objective of minimizing the total travel cost. In the two-stage version, the demands of each of the customers are not known until a vehicle arrives to serve them, which makes the problem more difficult. In this variant, we allow for the vehicle(s) to return to the depot once along their route in case their capacity is exceeded by their customers. This allows vehicles to refill, which gives the possibility of serving a larger set of demand realizations. Finding this refill point and calculating the extra cost incurred by the return to the depot is the main challenge in solving the 2SCVRP. This problem is interesting in fleet distribution systems, for example, as the demand of their customers may not be known before the trucks leave their depot. Although interesting, the problem has not yet been formulated for this specific version, although similar versions with time windows [13]. The 2SCVRP is an NP-complete problem, and as is common for NP-complete problems, formulating it as an IP gives a method of solving the problem which is capable of solving instances of the problem that would take billions of years using a naive approach. In this paper, we'll give formulations for both the single and multi vehicle versions of the problem. In both cases, we give two different formulations of the problem, and compare them computationally using instances of varying sizes and complexities.

## Acknowledgements

I would like to acknowledge my supervisor, Dr. Ricardo Fukasawa, for his insight and encouragement.

I would also like to thank Dr. Ricardo Fukasawa and Dr. Jeff Orchard jointly for acting as my committee members.

## Dedication

This is dedicated to my fiancee, Chloe Thiessen.

# Table of Contents

# List of Tables

# Chapter 1

# Introduction

The Capacitated Vehicle Routing Problem (CVRP) is concerned with finding the shortest tour for a vehicle to serve customers with a given demand and return to its original starting position. There may be one vehicle (in which case the CVRP is really a restatement of the Travelling Salesman Problem [TSP]) or multiple vehicles. Every vehicle also has a pre-specified capacity, and in the multi vehicle case all the vehicles have the same capacity. The CVRP was first proposed by Dantzig and Ramser in 1959 [6], and been studied extensively since then. They proposed an algorithm to solve the problem without the use of linear programming in their paper, which was improved by Clarke and Wright in 1964 [5]. This algorithm did not make use of the TSP formulation known at that point, also known to Dantzig [7], so it could not simply expand on the methods of solving linear programs. Eventually, the CVRP was formulated as an integer program (IP) in various ways by Laporte et al. [9], Gavish and Graves [2], and Laporte and Nobert [8], among others. This problem is NP-complete, as it is a generalization of the bin packing and TSP problems, both of which are NP-complete. In the deterministic version of the problem, the demands of each of the customers is known before the problem is solved.

In the stochastic version, by contrast, the demand of each of the customers is not known before the problem is solved. Many different ways of defining the randomness in the system have been studied, and some examples will be given later. This situation is more useful in some applications than the deterministic situation, since sometimes the demands will be unknown for each of the customers along the route until they are reached by a vehicle. Some applications of this approach are in supply chain management, fleet distribution and delivery services, among others where the demand at each stage is unknown at the outset. We will focus our attention on the stochastic version of the problem.

As is often the case with NP-complete problems, the most effective method of obtaining an optimal solution to the CVRP in practice is to formulate it and then use an IP solver to compute an answer. There are many different ways of formulating the deterministic version of the problem as an IP. The first formulation of the CVRP, given by Laporte et al. in [9], is very similar to a common formulation of the TSP given by Dantzig et al. [7] - it is a similar edge-based formulation that ensures each vertex has the right number of incoming and outgoing edges. There are also a number of other formulations developed in the 1980s to the 2000s by Laporte and Nobert [8], Toth [18], and others, including a formulation based on re-interpreting the CVRP as a multi-commodity flow problem given in [1]. These deterministic formulations are often natural starting points for constructing formulations for the stochastic CVRP. It is important to note that different types of formulations give rise to different methods of solving IPs. For example, branch-and-cut algorithms are designed to solve edge-based formulations of the CVRP, and branch-cut-and-price algorithms are designed to solve set-partition-based formulations. This distinction will guide the analysis of computational results, since the important elements to be analyzed are dependent on how the problem is solved.

There are many ways of examining the CVRP in a stochastic setting. For example, there is the Chance-Constrained CVRP, in which a solution is considered if it is can serve all its customers with probability $1 - \epsilon$ for some given $\epsilon$. This method of examining the stochastic CVRP has been studied extensively [20, 14, 22]. Another way of examining the stochastic CVRP is the two-stage CVRP (2SCVRP), in which each possible tour can be followed by a recourse action (returning to the depot to refill to its capacity) if a vehicle does not have the capacity to serve all its customers in a given realization of demands. The recourse action we allow is returning to the depot to refill the vehicle before carrying on its route. The 2SCVRP is the version of the stochastic CVRP which will be discussed in this paper. It is good to note here that there are some pros and cons in how the randomness is handled in the 2-stage and chance-constrained variants of the CVRP. The chance-constrained variant is more robust because the tolerance can be set so there are only a few failures, but those failures could be very costly as their costs are not taken into account. By contrast, the 2-stage variant is less robust because it allows for failed routes, but the routes that fail are taken into account in the overall cost so their negative impact is minimized.

The 2SCVRP has been investigated in the past, but generally with restrictive assumptions on the demands like independence of random variables or specific distributions [3, 21]. A version of the 2SCVRP with minimal assumptions on the demands is given by Poggi and Spyrides [15]. The assumptions made are that the distribution is finite discrete (which allows for a finite number of demand realizations, and thus a finite IP) and that there is at most one recourse action per vehicle (this makes defining the recourse cost much simpler). In that work, a preliminary formulation of the two-stage CVRP with one vehicle is given for a few different recourse actions. The two actions they gave were simple recourse and reshuffle recourse. The simple recourse action they investigated is simply returning to the depot when the capacity is exceeded and returning to the next customer along the route, and the reshuffle action is determining the optimal route to follow for the remaining customers after returning to the depot. In this paper, we'll work to improve their formulation for the simple recourse action and extend it to the case where there is more than one vehicle. This formulation is given under the assumption that the vehicle can only refill once. This seems reasonable because allowing a vehicle to refill more than once can make routes more prone to variation, which would be undesirable in practice.

In Section 1.1, we give the formulation of the CVRP given by Letchford and Salazar-Gonzalez, which is based on the multi-commodity flow formulation [1]. The TSP formulations we give are similar to this formulation, so we write it out in its entirety for easy reference. In Chapter 2, we'll give a more formal definition of the problem, and in Chapters 3 and 4, we give formulations of the problem for the single and multi vehicle versions, respectively, of the problem. More specifically, the sections are divided as follows. In Chapter 3, we give a single vehicle TSP formulation in Section 3.1, and two recourse formulations in Sections 3.2 and 3.3. Both recourse sections end with a full two-stage formulation. Similarly, in Chapter 4, we give multi vehicle TSP formulations in Sections 4.1 and 4.2, and two recourse formulations in Sections 4.3 and 4.4. Here as well both recourse sections end with a full two-stage formulation. Finally, the formulations are compared empirically in Chapter 5, with the single vehicle formulations in Section 5.1 and the multi vehicle formulations in Section 5.2.

## 1.1 Multi-Commodity Flow Formulation

Before giving a formal problem definition for the 2SCVRP, we give the multi-commodity flow formulation given by Letchford and Salazar-Gonzalez for the CVRP, as the first stage we'll give for the 2SCVRP is very similar. The deterministic CVRP is as follows. Let $G = (V, E)$ be a complete graph with $V = \{1, \ldots, n\}$, $\bar{V} = \{2, \ldots, n\}$, and edge costs $\{c_{ij} : ij \in E\}$. Also let $\{d_i : 2 \leq i \leq n\}$ be the customer demands, $d(S) = \sum_{i \in S} d_i$ for $S \subseteq \bar{V}$, $\mathcal{V} = \{1, \ldots m\}$ be the vehicle set, and $C$ be the vehicle capacity. Find the edge set $P$ satisfying

$$P = \bigcup_{v \in \mathcal{V}} P_v,\ P_v \text{ is a cycle such that } 1 \in P_v, \forall v \in \mathcal{V},$$

$$V(P) = \bigcup_{v \in \mathcal{V}} V(P_v) = V,$$

$$V(P_v) \cap V(P_w) = \{1\}, \forall v, w \in \mathcal{V}, v \neq w.$$

$$d(\bar{V}(P_v)) \leq C, \forall v \in \mathcal{V}$$

such that $\sum_{ij \in P} c_{ij}$ is minimized. Here $V(\bar{V}) = \{v \in V(\bar{V}) : v \text{ is incident to some edge in } P\}$.

Their formulation of this problem, based on the multi-commodity flow formulation, is as follows:

$$\begin{cases} \min & \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}y_{ij} \qquad \text{MCF} \\ \text{s.t.} & \displaystyle\sum_{j=1}^{n} y_{ij} = 1,\ \ \forall i \in V \\ & \displaystyle\sum_{i=1}^{n} y_{ij} = 1,\ \ \forall j \in V \\ & \displaystyle\sum_{j=1}^{n} y_{1j} = m \\ & \displaystyle\sum_{i=1}^{n} y_{i1} = m \\ & \displaystyle\sum_{j=1}^{n} x_{ijk} - \sum_{j=1}^{n} x_{jik} = \begin{cases} 1, & i = 1, k \neq 1 \\ -1, & i = k \neq 1 \\ 0, & \mathit{otherwise} \end{cases} \quad \forall i, k \in V \\ & \displaystyle\sum_{k \in \bar{V}} d_k x_{ijk} \leq C y_{ij}\ \ \forall i, j \in V \\ & x_{ijk} \in \mathbb{Z}_{\geq 0},\ \ \forall i, j \in V, i \neq j, \forall k \in \bar{V} \\ & y_{ij} \in \{0, 1\}\ \ \forall i, j \in V, i \neq j \end{cases}$$

Given some feasible solution $[x, y]$ representing an edge set $P$, the variables in this formulation are given by

$$y_{ij} = \begin{cases} 1, & ij \in P \\ 0, & otherwise \end{cases}$$

and

$$x_{ijk} = \begin{cases} 1, & ij \in P, i, j, \text{ and } k \text{ are served by the same vehicle and } i \text{ is served before } k \\ 0, & otherwise \end{cases}$$

With these variables, we can outline the purpose of each of the constraints. The first two constraints ensure that each customer has an incoming and an outgoing edge, and the third and fourth constraints ensure that there are $M$ vehicles that enter and leave the depot. With only these constraints, there is no guarantee that there is one cycle corresponding to each vehicle, so we add the final two constraints to enforce that there is exactly one cycle per vehicle. This is done by tracing the route from the depot to every customer with the $x_{ijk}$ variables and matching the $y_{ij}$ variables to the corresponding $x_{ijk}$ variables.

# Chapter 2

# Problem Definition

Suppose we are given a directed graph $G = (V, E)$ with edge costs $\{c_{ij} \geq 0 : ij \in E\}$ such that $1j, j1 \in E$ for every $j \in \{2, \dots, n\}$ and $c_{1j} > 0, c_{j1} > 0$ for every $j \in \{2, \dots, n\}$. Here the vertex set is labelled $V = \{1, \dots, n\}$, where vertex 1 is the depot, and vertices $\bar{V} = \{2, \dots, n\}$ are customers who must be served. For an edge set $P \subseteq E$ we'll let

$$V(P) = \{v \in V : v \text{ is incident to some edge in } P\}$$

and

$$\bar{V}(P) = \{v \in \bar{V} : v \text{ is incident to some edge in } P\}.$$

At the depot, there are $m$ vehicles each having capacity $C$, collectively denoted by $\mathcal{V} = \{1, 2, \dots, m\}$. The customer demands are not known until a vehicle arrives to serve the customer, and are considered to be random variables as a result. We introduce the probability distribution in the form of scenarios. We define a scenario as a tuple of demands $(d_{2s}, \dots, d_{ns})$ such that $0 \leq d_{is} \leq C \ \forall i \in \bar{V}$. We'll use $\mathcal{D}$ to reference the set of possible scenarios for any instance of the problem, and we require that $|\mathcal{D}| < \infty$ so that the resulting IP is finite. Based on this set of valid scenarios, we define

$$\mathcal{S} = \{1, \dots, |\mathcal{D}|\}$$

as the set of indexes $s$. Each scenario $s$ occurs with probability $\mathbb{P}[s]$, and these probabilities defines a discrete probability distribution on the random vector of demands; that is $\mathbb{P}[s] > 0$, and $\sum\limits_{s \in \mathcal{S}} \mathbb{P}[s] = 1$. We also give the following definitions:

**Definition 2.1.** $m$-TSP solution

An edge set $P$ is called an $m$-TSP solution if it has the following properties:

$$P = \bigcup_{v \in \mathcal{V}} P_v,\ P_v \text{ is a cycle such that } 1 \in P_v, \forall v \in \mathcal{V}, \tag{2.1}$$

$$V(P) = \bigcup_{v \in \mathcal{V}} V(P_v) = V, \tag{2.2}$$

$$V(P_v) \cap V(P_w) = \{1\}, \forall v, w \in \mathcal{V}, v \neq w. \tag{2.3}$$

Each cycle $P_v$ in an $m$-TSP solution $P$ is called a route, and we say that each route starts and ends at vertex 1. □

Given a route $P$, we now define the recourse cost for $P$. Let $P$ traverse the vertices in the order $1\kappa_1 \dots \kappa_{r-1}1$, with $\kappa_r = 1$. Using this, we define

$$\alpha(P, s) = \begin{cases} \arg\min \left\{ 1 \leq k \leq r-1 : \sum\limits_{j=1}^{k} d_{\kappa_j s} \geq C \right\}, & \text{if } \sum\limits_{j=1}^{r-1} d_{\kappa_j s} \geq C \\ r, \ \textit{otherwise} \end{cases}$$

Then

$$\gamma(P, s) = \kappa_{\alpha(P,s)}$$

is the location of the refill point for route $P$ in scenario $s$.

It is good to note here that the definition of $\gamma$ enforces the policy that a vehicle refills immediately after reaching capacity. To see why this is reasonable, consider $m = 1$, $P = \{12, 23, 34, 41\}$ and the demand distribution $(C, 0, C-1)$. The vehicle could refill after customer 2 or customer 3, but because the vehicle does not know that customer 3 has 0 demand until it reaches the customer, it does not make sense for the vehicle to wait until it reaches customer 3 to refill. Rather, it should refill immediately after serving customer 2 in order to prepare for future customers. It also allows a vehicle to return to the depot to refill in the midst of serving a customer. To see why, consider $m = 1$, $P = \{12, 23, 34, 41\}$ and the demand distribution $(C-2, 3, C-2)$. Clearly the vehicle must serve customer 2, then partially serve customer 3 and return to the depot before finishing the route. If we do not allow for vehicles to refill in the midst of serving a customer, it is likely that reasonable solutions like this one will be rendered infeasible by a seemingly innocuous scenario like this one.

We should also note that this means that a refill will occur if the total demand along $P_v$ is equal to $C$, even though this should not be the case. This could be avoided by choosing $\alpha$ such that no refill occurs if $\sum_{j=1}^{r-1} d_{\kappa_j s} < C$. However, we could have a scenario in which $\sum_{j=1}^{\alpha(P,s)} d_{\kappa_j s} = \sum_{j=1}^{\alpha(P,s)+1} d_{\kappa_j s} = C$, in which case finding the maximal $t$ such that $\sum_{j=1}^{t} d_{\kappa_j s} \leq C$ will not find $\alpha(P,s)$, but rather some $t > \alpha(P,s)$. Avoiding this introduces complexity into the formulation, so for now we satisfy ourselves with $\alpha$ defined the way it is. Finding a way to avoid this issue while avoiding a refill when the total demand is equal to $C$ should be the subject of future work.The expected recourse cost for the $m$-TSP solution $P$ can be defined as the sum of recourse costs for its constituent routes, averaged over all scenarios. This sum is given by

$$\sum_{s\in\mathcal{S}}\sum_{v\in\mathcal{V}} \mathbb{P}[s]\big(c_{\gamma(P_v,s)1} + c_{1\gamma(P_v,s)}\big),$$

where $\mathbb{P}[s] > 0$ is the probability that scenario $s$ occurs. In the case where all distributions are equally likely, we have $\mathbb{P}[s] = \frac{1}{|\mathcal{S}|}$. Since there is no edge $11 \in E$, we can define $c_{11} = 0$ to make formally writing the second stage cost more notationally convenient. By the definition of $\alpha(P_v, s)$ we have a recourse cost of 0 corresponding to scenario $s$ if $\sum_{j=1}^{r-1} d_{\kappa_j s} < C$.

Since the first stage cost is $\sum_{ij\in P} c_{ij}$, this gives a total cost for a stochastic $m$-TSP solution $P$ of

$$\sum_{ij\in P} c_{ij} + \sum_{s\in\mathcal{S}}\sum_{v\in\mathcal{V}} \mathbb{P}[s]\big(c_{\gamma(P_v,s)1} + c_{1\gamma(P_v,s)}\big). \tag{2.4}$$

The problem we pose is to find the stochastic $m$-TSP solution $P$ which minimizes this cost.

**Definition 2.2.** Stochastic $m$-TSP solution

An $m$-TSP solution $P$ is called a stochastic $m$-TSP solution if it satisfies

$$\sum_{i\in\bar{V}(P_v)} d_{is} \leq 2C - 1, \forall s \in \mathcal{S}, v \in \mathcal{V}, \tag{2.5}$$

or equivalently

$$D_{max}(\bar{V}(P_v)) \leq 2C - 1, \forall v \in \mathcal{V}, \tag{2.6}$$

where

$$D_{max}(S) := \max_{s \in \mathcal{S}} \sum_{i \in S} d_{is}$$

for $S \subseteq \bar{V}$. $\square$

The constraint that the total demand must be less than $2C - 1$ along every cycle $P_v$ in every scenario is required so that every vehicle refills at most once on its journey. This also should be the subject of future work, since a vehicle should be able to serve a route with a total demand of $2C$. The issue is closely linked to the one mentioned above about refilling if a route has total demand $C$.

# Chapter 3

# Single Vehicle Variant

The single vehicle variant is largely based on a presentation by Poggi and Spyrides [15]. However, the list of recourse constraints is pared down significantly from this initial formulation. We make a slight modification to the range of total demands, restricting it only to be less than $2C - 1$. This is to ensure that each route is divided into customers before the capacity is reached ($\leq C - 1$) and those after the capacity is reached ($\geq C$) with both ranges having the same measure (0 to $C - 1$ and $C$ to $2C - 1$). This is also the reason for condition (2.5) having $2C - 1$ instead of $2C$.

## 3.1 TSP Constraints

The variables defined for the first stage constraints are

- $y_{ij}$ (binary): An indicator for whether the edge $ij$ is in the given TSP solution.
- $x_{ijk}$ (binary): An indicator for whether the edge $ij$ is used along the path from 1 to $k$ on the given TSP solution.

**Definition 3.1.** We say that $[x\, y]^T$ represents a TSP solution $P$ if

- $y_{ij} = 1$ if and only if $ij \in P$.
- If $P$ traverses the vertices in the order $1\kappa_1 \dots \kappa_{n-1}1$, and $j, k \in V$ are such that $j = \kappa_a$ and $k = \kappa_b$, then $x_{ijk} = 1$ if and only if $ij \in P$ and $a \leq b$. $\square$

In our constraints, we'll also use the function

$$f(i,k) = \begin{cases} 1 & i = 1, k \neq 1 \\ -1 & i = k \neq 1 \\ 0 & \textit{otherwise} \end{cases} .$$

With this function, we can write down the TSP formulation, which is similar in nature to MCF.

$$\begin{cases} \min & \sum_{i=1}^{n} \sum_{j=1}^{n} y_{ij} c_{ij} & \text{(3.1a)} \\ \text{s.t.} & \sum_{j=1}^{n} y_{ij} = 1, \ \ \forall i \in V & \text{(3.1b)} \\ & \sum_{i=1}^{n} y_{ij} = 1, \ \ \forall j \in V & \text{(3.1c)} \\ & \sum_{j=1}^{n} x_{1jk} = 1, \ \ \forall k \in V & \text{(3.1d)} \\ & \sum_{j=1}^{n} x_{ijk} - \sum_{j=1}^{n} x_{jik} = f(i,k) \ \ \forall i,k \in V & \text{(3.1e)} \\ & x_{ijk} - y_{ij} \leq 0 \ \ \forall i,j,k \in V & \text{(3.1f)} \\ & y_{ij}, x_{ijk} \in \{0,1\} \ \ \forall i,j,k \in V & \text{(3.1g)} \end{cases}$$

Now we show that a vector $[x\, y]^T$ is feasible for this set of constraints if and only if it represents some feasible TSP solution.

**Lemma 3.1.** *$[x\, y]^T$ is feasible for (3.1) if and only if it represents some TSP solution $P$.*

*Proof.* ($\Longleftarrow$) Suppose we have a TSP solution $P$ represented by $[x\, y]^T$.

(3.1b): Let $i \in V$. By (2.2), $i \in V(P)$, and since $P$ is a cycle there is exactly one outgoing edge $ij$ in $P$. Then $y_{ij} = 1$ and $y_{ik} = 0$ for all vertices $k \neq j$, so $\sum_{j=1}^{n} y_{ij} = 1$.

(3.1c): Let $i \in V$. By (2.2), $i \in V(P)$, and since $P$ is a cycle there is exactly one incoming edge $ji$ in $P$. Then $y_{ji} = 1$ and $y_{ki} = 0$ for all vertices $k \neq j$, so $\sum_{j=1}^{n} y_{ji} = 1$.

(3.1d): Let $k \in V$. By (2.2), $1 \in V(P)$, and since $P$ is a cycle there is exactly 1 outgoing edge $1j$ from vertex 1. Since $P$ begins at vertex 1, this edge precedes every vertex $i$ along $P$, so $x_{1jk} = 1$. Moreover, since $1i$ is not on $P$ for every other vertex $i$, we have that $x_{1ik} = 0$ for every $i \neq j$. Thus, $\sum_{l=1}^{n} x_{1lk} = 1$.

(3.1e): Let $i, k \in V$. By (2.2) and the fact that $P$ is a cycle, there is exactly one incoming edge $j_1 i$ to $i$ and one outgoing edge $ij_2$ from $i$.

(a) Let $i = 1$, $k \neq 1$, so $f(i,k) = 1$. Then only $1j_2$ occurs before we reach $k$ and not $j_1 1$. This gives $\sum_{j=1}^{n} x_{1jk} = 1$ and $\sum_{j=1}^{n} x_{j1k} = 0$, so the constraint is satisfied.

(b) Let $i = k$, $k \neq 1$, so $f(i,k) = -1$. Then only $j_1 i$ occurs before we reach $k$ and not $ij_2$. This gives $\sum_{j=1}^{n} x_{ijk} = 0$ and $\sum_{j=1}^{n} x_{jik} = 1$, so the constraint is satisfied.

(c) Let $k = 1$, $i \in V$, so $f(i,k) = 0$. Then both $j_1 i$ and $ij_2$ occur before we reach $k$. This gives $\sum_{j=1}^{n} x_{ij1} = \sum_{j=1}^{n} x_{ji1} = 1$, so the constraint is satisfied.

(d) Let $i, k \neq 1$ be such that $i \neq k$ so $f(i,k) = 0$. Either both $j_1 i$ and $ij_2$ occur before we reach $k$ (if $i$ comes before $k$ along $P$) or neither of them do (if $i$ comes after). This gives either $\sum_{j=1}^{n} x_{1j1} = \sum_{j=1}^{n} x_{j11} = 1$ (in the first case) or $\sum_{j=1}^{n} x_{1j1} = \sum_{j=1}^{n} x_{j11} = 0$ (in the second case), so the constraint is satisfied.

(3.1f): If $y_{ij} = 0$, then the edge $ij \notin P$ and $x_{ijk} = 0$ for every $k \in V$. If $y_{ij} = 1$, then $x_{ijk} \leq y_{ij}$ since $x_{ijk} \leq 1$. Thus, the constraint is satisfied.

(3.1g): Clearly the variables are binary because there are indicators for the edges included in $P$.

($\Longrightarrow$) Now we need to show that if $[x\ y]^T$ is feasible given (3.1b) to (3.1g), then it represents a feasible TSP solution. To show that $[x\,y]^T$ represents a TSP solution, we need to show that it satisfies properties (2.1), (2.2), and (2.3), and also that $x$ and $y$ represent the same solution.

1. It is clear, since the indegree and outdegree for each vertex is set to 1, that $[x\ y]^T$ represents a composition of cycles. To show that there is exactly one cycle, we look at the $x$ vector. Suppose, for sake of contradiction, that there is more than one cycle. Only one of these cycles contains vertex 1 since the cycles together cover every vertex exactly once. Let the cycle containing vertex 1 traverse the vertices in the

order $1\kappa_1 \ldots \kappa_r 1$, and consider a vertex $k$ which is in a cycle that doesn't contain 1. We'll show that constraint (3.1e) breaks for $k$ and some $i \in \{\kappa_1, \ldots, \kappa_r, 1\}$. From (3.1d) there is an edge $1j$ such that $x_{1jk} = 1$, and (3.1f) this edge is $1\kappa_1$. In order to satisfy $\sum_{j=1}^{n} x_{\kappa_1 jk} - x_{j\kappa_1 k} = 0$, then, we require that $\sum_{j=1}^{n} x_{\kappa_1 jk} = 1$. By (3.1f) again, the edge $\kappa_1 j$ with $x_{\kappa_1 jk} = 1$ is $\kappa_1\kappa_2$. If we continue this so that constraint (3.1e) is satisfied for $i \in \{\kappa_1, \ldots, \kappa_r\}$, we require that $\sum_{j=1}^{n} x_{\kappa_t jk} - x_{j\kappa_t k} = 0$ for $1 \le t \le r$, since $k \ne \kappa_t$ for any $t$. Thus, we have $\sum_{j=1}^{n} x_{\kappa_t jk} \sum_{j=1}^{n} x_{j\kappa_t k} = 1$ for all $1 \le t \le r$, so $x_{\kappa_{t-1}\kappa_t k} = 1$ for every $2 \le t \le r$ and $x_{1\kappa_1 k} = 1$. For $i = 1$, we require that $\sum_{j=1}^{n} x_{1jk} - x_{j1k} = 1$. However, we have $x_{1\kappa_1 k} = x_{\kappa_r 1k} = 1$ to satisfy (3.1e) for $i = \kappa_1, \kappa_r$, and from (3.1f) we have $x_{1jk} = 0$ for all $j \ne \kappa_1$ and $x_{j1k} = 0$ for all $j \ne \kappa_r$. Thus, we have $\sum_{j=1}^{n} x_{1jk} = \sum_{j=1}^{n} x_{j1k} = 1$ and (3.1e) is not satisfied for $i = 1$ and $k$.

2. This is satisfied by constraints (3.1b) and (3.1c), which ensure that each vertex has exactly one incoming and one outgoing edge.

3. This is automatically satisfied, since $|\mathcal{V}| = 1$.

4. Let the TSP solution described by $y$ be denoted by $P$, and let $P$ traverse the vertices in the order $1\kappa_1 \ldots \kappa_{n-1} 1$. We expect that $x_{ijk} = 1$ if and only if $ij = \kappa_{s-1}\kappa_s$ and $k = \kappa_t$ for $t \ge s$. By (3.1f), we have $x_{ijk} = 0$ if $ij \ne \kappa_{s-1}\kappa_s$ for some $1 \le s \le n$, so we need only concern ourselves with variables of the form $x_{\kappa_{s-1}\kappa_s\kappa_t}$.

   Set $k \in V$, and let $t$ be such that $k = \kappa_t$. We'll show that $x_{\kappa_{s-1}\kappa_s k} = 1$ if and only if $s \le t$ by induction on $s$. First, for $s = 1$ this is true, since by (3.1d) we have $x_{1\kappa_1 k} = 1$ for every $k \in V$. Now let $1 < s \le t$, and assume that $x_{\kappa_{s-2}\kappa_{s-1}k} = 1$. By (3.1e), we have that $f(\kappa_{s-1}, k) = 0$, so $x_{\kappa_{s-2}\kappa_{s-1}k} = x_{\kappa_{s-1}\kappa_s k}$, and both are equal to 1 by the induction hypothesis. For $s = t+1$, we make two observations - first, $x_{\kappa_{t-1}kk} = 1$, and second that $f(k,k) = -1$. By (3.1e), we have $x_{k\kappa_s k} - x_{\kappa_{t-1}kk} = -1$, and since $x_{\kappa_{t-1}kk} = 1$ we have $x_{k\kappa_s k} = 0$. Finally, let $t+1 < s \le n$, and assume that $x_{\kappa_{s-2}\kappa_{s-1}k} = 0$. Then $f(\kappa_{s-1}, k) = 0$, so by (3.1e) we have $x_{\kappa_{s-1}\kappa_s k} = x_{\kappa_{s-2}\kappa_{s-1}k} = 0$. Thus, $x_{\kappa_{s-1}\kappa_s\kappa_t} = 1$ if and only if $s \le t$ as expected, and $x$ and $y$ represent the same TSP solution.

As a result, we see that if $[x\ y]^T$ satisfies the constraints (3.1b) to (3.1g), then $[x\ y]^T$ represents a TSP solution.

□

Now that we have shown our TSP formulation works as intended, we work on giving the recourse formulations.

## 3.2 Recourse Costs

For the TSP constraints, we did not deal with the demands at all. However, we need to be more concerned about the different scenarios when discussing the recourse costs. In a given scenario, we say the vehicle refills if it returns to the depot from its current vertex, increases its capacity to $C$, and returns to the same vertex from which it started. Such a refill is the recourse action that can be taken, and we say that a recourse action is taken at a vertex $k$ if the vehicle returns to the depot from vertex $k$ in its recourse action. As discussed in the problem definition, we ensure that the vehicle can only refill once along its route, and to this end we must ensure that any feasible solution for this IP represents a stochastic TSP path. This will be shown once the formulation is stated.

With the above notion of a refill, the variables defined for the recourse costs are as follows:

- $o_i$: contains the position of vertex $i$ along the TSP solution $P$ represented by $[x\,y]^T$.
- $w_{ks}$: indicates whether vertex $k$ is served at or after the vehicle is refilled in scenario $s$.
- $nns_s$: the number of customers served at or after the refill point along the TSP solution $P$ represented by $[x\,y]^T$
- $pays_{ks}$: indicates whether the vehicle goes from client $k$ to the depot and returns in scenario $s$. In other words, we have

$$pays_{ks} = \begin{cases} 1, & \text{if a recourse action is taken at } k \text{ in scenario } s \\ 0, & otherwise \end{cases}$$

With the variables defined as above, we have the following recourse LP:

$$\xi(x) = \begin{cases} \min & \sum_{s\in\mathcal{S}}\sum_{k=2}^{n} \mathbb{P}[s] pays_{ks}(c_{1k}+c_{k1}) & \text{(3.2a)} \\ \text{s.t.} & (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \le C-1,\ \ \forall k\in\bar{V}, s\in\mathcal{S} & \text{(3.2b)} \\ & (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \ge 0,\ \ \forall k\in\bar{V}, s\in\mathcal{S} & \text{(3.2c)} \\ & (\sum_{i=1}^{n}\sum_{j=2}^{n} x_{ijk}) - o_k = 0,\ \ \forall k\in\bar{V} & \text{(3.2d)} \\ & (\sum_{k=2}^{n} w_{ks}) - nns_s = 0,\ \ \forall s\in\mathcal{S} & \text{(3.2e)} \\ & o_k + nns_s + pays_{ks} \ge n\cdot w_{ks} + 1, \forall k\in\bar{V}, s\in\mathcal{S} & \text{(3.2f)} \\ & o_k, nns_s \in \mathbb{Z}, w_{ks}, pays_{ks} \in \{0,1\} & \text{(3.2g)} \end{cases}$$

Now we need to show that TSP solutions are feasible for (3.2) if and only if they are stochastic TSP solutions. To do this, we first present the following lemma.

**Lemma 3.2.** *Suppose that $P$ is a TSP solution represented by $[x\, y]^T$, and suppose that $P$ serves its customers in the order $1\kappa_1 \dots \kappa_{n-1}1$. Let $k = \kappa_t$. Given constants $\{a_j : j \in \bar{V}\}$, we have*

$$\sum_{l=1}^{t} a_j = \sum_{i=1}^{n}\sum_{j=2}^{n} a_j x_{ijk}$$

*Proof.* This follows easily from the reverse argument in Lemma 3.1, since it is shown there that $x_{ijk} = 1$ if and only if $ij = \kappa_{s-1}\kappa_s$ for some $1 \le s \le t$. Thus,

$$\begin{aligned} \sum_{i=1}^{n}\sum_{j=2}^{n} a_j x_{ijk} &= \sum_{l=1}^{t} a_{\kappa_l} x_{\kappa_{l-1}\kappa_l k} \\ &= \sum_{l=1}^{t} a_{\kappa_l} \end{aligned}$$

□

**Lemma 3.3.** *Given a TSP solution $P$ represented by $[x\ y]^T$, (3.2) with input $x$ is feasible if and only if $P$ is a stochastic TSP solution.*

*Proof.* ($\Longleftarrow$) Let $P$ serve the customers in the order $1\kappa_1 \ldots \kappa_{n-1}1$, and let $k = \kappa_t$. Since $P$ is a stochastic TSP solution, we have the variables set as follows:

$$x_{ijk} = \begin{cases} 1, & ij \in P, j \text{ comes before } k \text{ along } P \text{ or } j = k \\ 0, & otherwise \end{cases}$$

$$w_{ks} = \begin{cases} 1, & t \geq \alpha(P, s) \\ 0, & otherwise \end{cases}$$

$$pays_{ks} = \begin{cases} 1, & k = \gamma(P, s) \\ 0, & otherwise \end{cases}$$

and

$$nns_s = n - \alpha(P, s), o_k = t.$$

By the definition of $\gamma(P, s)$, we have by Lemma 3.2 that

$$0 \leq \sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ijk} = \sum_{l=1}^{t} d_{\kappa_l s} \leq C - 1$$

if $t < \alpha(P, s)$ and

$$C \leq \sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ijk} = \sum_{l=1}^{t} d_{\kappa_l s} \leq 2C - 1$$

if $t \geq \alpha(P, s)$, so constraints (3.2b) and (3.2c) are both satisfied by $P$. Clearly we have

$$\sum_{i=1}^{n} \sum_{j=2}^{n} x_{ijk} = t$$

by Lemma 3.2 and

$$\sum_{k=2}^{n} w_{ks} = (n-1) - (\alpha(P,s) - 1) = n - \alpha(P,s) = nns_s,$$

so constraints (3.2d) and (3.2e) are satisfied.

Finally we examine constraint (3.2f). We have $o_k + nns_s + pays_{ks} \geq o_k + nns_s \geq 1$, so if $w_{ks} = 0$ this constraint is satisfied. If $w_{ks} = 1$, then $t \geq \alpha(P,s)$. If $t \geq \alpha(P,s) + 1$, then $o_k + nns_s + pays_{ks} \geq o_k + nns_s \geq (\alpha(P,s) + 1) + (n - \alpha(P,s)) = n + 1$ and the constraint is satisfied. Finally, if $t = \alpha(P,s)$, we have $pays_{ks} = 1$ and $o_k + nns_s + pays_{ks} = \alpha(P,s) + (n - \alpha(P,s)) + 1 = n + 1$, so the constraint is satisfied. Since (3.2g) is clearly satisfied, (3.2) is feasible.

($\Longrightarrow$) Suppose $\xi(x,y)$ is feasible, and let $s \in \mathcal{S}$. Let $v \in \bar{V}$ be the last customer served along $P$. Then

$$\sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ijk} = \sum_{i \in \bar{V}} d_{is},$$

since

$$\sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ijk} = \sum_{i \in \bar{V}} d_{is} a(i,k)$$

where

$$a(i,k) = \begin{cases} 1, & i \text{ is served before } k \text{ along } P \\ 0, & \textit{otherwise} \end{cases}$$

and every vertex in $\bar{V}$ comes before $k$. From constraint (3.2g) we know that $w_{ks} \in \{0,1\}$, and from constraint (3.2b) we have either

$$\sum_{i \in \bar{V}} d_{is} \leq C - 1 < 2C - 1$$

if $w_{ks} = 0$ and

$$\sum_{i \in \bar{V}} d_{is} \leq 2C - 1$$

if $w_{ks} = 1$, so $\sum_{i \in \bar{V}} d_{is} \leq 2C - 1$. Thus, (2.5) is satisfied. Since $P$ is a TSP solution, (2.1) through (2.3) are satisfied, so $P$ is indeed a stochastic TSP solution.

□

Now that we have ensured that every stochastic TSP solution is feasible for (3.2), we must ensure that the recourse cost associated with a stochastic TSP solution $P$ is the appropriate recourse cost. Since the recourse cost is dependent only on the $pays_{ks}$ variables and input parameters, we need only ensure that the $pays_{ks}$ variables are set properly as defined in the problem definition.

**Lemma 3.4.** *Given a stochastic TSP solution $P$ represented by $[x\,y]^T$, an optimal solution to (3.2) with the input $x$ satisfies*

$$pays_{ks} = \begin{cases} 1, & k = \gamma(P, s) \\ 0, & otherwise \end{cases}$$

*for every demand scenario $s$.*

*Proof.* To start, we show that

$$pays_{ks} \geq \begin{cases} 1, & k = \gamma(P, s) \\ 0, & otherwise \end{cases}$$

for every $k \in \bar{V}$ and $s \in \mathcal{S}$. Let $s \in \mathcal{S}$. For simplicity, suppose that $P$ traverses the vertices in the order $1\kappa_1\kappa_2 \ldots \kappa_{n-2}\kappa_{n-1}1$.

First, we show that the $w_{ks}$ variables are set appropriately, since if they are it is clear that $nns_s$ is all set correctly. We should have that $w_{\kappa_i s} = 1$ if and only if $\sum_{j=1}^{i} d_{\kappa_j s} \geq C$. Since $\sum_{j=1}^{i} d_{\kappa_j s} = \sum_{l=1}^{n} \sum_{j=2}^{n} d_{js} x_{lj\kappa_i}$, this is true by constraints (3.2b) and (3.2c). Thus, the $w_{ks}$ variables are set appropriately, and as a result $nns_s$ really is the number of customers served after the refill. Note that $0 \leq nns_s \leq n - 1$.

Next, we note that $\sum_{l=1}^{n} \sum_{j=2}^{n} x_{lj\kappa_i} = i = o_{\kappa_i}$ by Lemma 3.2, so $o_k$ is set correctly as well for every vertex $k \in \bar{V}$. Note that $1 \leq o_k \leq n$.

Let $u = \kappa_a = \gamma(P, s)$ be the refill point, and suppose that $\gamma(P, s) \neq 1$. Since $\kappa_a$ is the refill point, we know that $a - 1$ customers are served before the refill. There are $n - 1$ customers in total, so we have $nns_s = (n - 1) - (a - 1) = n - a$. Thus, we have $n - a + 1 \leq o_v + nns_s \leq 2n - a$.

Consider a vertex $k = \kappa_b \neq 1$. If $b < a$, then $k$ is served before the refill point and we have $nw_{ks} + 1 = 1$. Since $o_k + nns_s \geq 1$, we have $o_k + nns_s \geq nw_{ks} + 1$ and $pays_{ks}$ can be 0. If $b = a$ then $k$ is the refill point. Since $w_{ks} = 1$ here, we have $nw_{ks} + 1 = n + 1$. We also have $o_k = b$, so $o_k + nns_s = b + (n - b) = n$. Thus, in order for the constraint $o_k + nns_s + pays_{ks} \geq nw_{ks} + 1$ to be satisfied, we must have $pays_{ks} = 1$. If $b > a$, then $k$ is served after the refill point and we have $nw_{ks} + 1 = n + 1$. As before, we have $o_k = b$, so $o_k + nns_s = b + (n - a) = n + (b - a)$. Since $b > a$, we have $o_k + nns_s \geq n + 1$, so $pays_{ks}$ can be 0. Thus, $pays_{ks}$ is forced to 0 if and only if $k = \gamma(P, v)$ is the refill point.

If $\gamma(P, s) = 1$, then there is no $1 \leq a \leq n - 1$ such that $\sum_{j=1}^{a} d_{\kappa_j s} \geq C$. By (3.2b) this means that $w_{ks} = 0$ for every $k \in \bar{V}$, and $nns_s = 0$. Then (3.2f) is $o_k + pays_{ks} \geq 1$ in this scenario. But $o_k \geq 1$ for every $k \in \bar{V}$, so (3.2f) is satisfied for both $pays_{ks} = 0$ and $pays_{ks} = 1$, independent of $k$. Thus, $pays_{ks}$ can be 0 for any $k$, which is as expected since $k \neq \gamma(P, s)$ for any $k \in \bar{V}$.

Finally, we show that if this inequality holds strictly for some vertex $k$ and some scenario $s$, then the solution is not optimal. Consider a solution $p_1$ for which $pays_{ks} = 1$ for some scenario $s \in \mathcal{S}$ and some $k \neq \gamma(P, s)$. Then create a new solution $p_2$ by setting $pays_{ks} = 0$ and leave all other variables the same, since we are required only that $pays_{ks} \geq 0$. But this new solution $p_2$ has cost less than that of $p_1$ by $\mathbb{P}[s](c_{1k} + c_{k1})$, so $p_1$ cannot be optimal. Thus, if $pays_{ks} = 1$ for some $k \neq \gamma(P, s)$ for some feasible solution to (3.2), then that solution is not optimal, so since

$$pays_{ks} \geq \begin{cases} 1, & k = \gamma(P, s) \\ 0, & otherwise \end{cases}$$

for any feasible solution for every $k \in \bar{V}$ and $s \in \mathcal{S}$, we have that at an optimal solution to (3.2)

$$pays_{ks} = \begin{cases} 1, & k = \gamma(P, s) \\ 0, & otherwise \end{cases}$$

as desired. □

This shows that the first recourse formulation gives the correct value for the recourse cost. Then the full formulation is

$$\left\{\begin{array}{ll}
\min & \sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}c_{ij} + \sum_{s\in\mathcal{S}}\sum_{k=2}^{n}\mathbb{P}[s]pays_{ks}(c_{1k}+c_{k1}) \\
\text{s.t.} & \sum_{j=1}^{n} y_{ij} = 1,\ \forall i \in V \\
& \sum_{i=1}^{n} y_{ij} = 1,\ \forall j \in V \\
& \sum_{j=1}^{n} x_{1jk} = 1,\ \forall k \in V \\
& \sum_{j=1}^{n} x_{ijk} - \sum_{j=1}^{n} x_{jik} = f(i,k)\ \forall i,k \in V \\
& (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \le C-1,\ \forall k \in \bar{V}, s \in \mathcal{S} \\
& (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \ge 0,\ \forall k \in \bar{V}, s \in \mathcal{S} \\
& (\sum_{i=1}^{n}\sum_{j=2}^{n} x_{ijk}) - o_k = 0,\ \forall k \in \bar{V} \\
& (\sum_{k=2}^{n} w_{ks}) - nns_s = 0,\ \forall s \in \mathcal{S} \\
& o_k + nns_s + pays_{ks} \ge n\cdot w_{ks} + 1, \forall k \in \bar{V}, s \in \mathcal{S} \\
& x_{ijk} - y_{ij} \le 0\ \forall i,j,k \in V \\
& y_{ij}, x_{ijk}, w_{ks}, pays_{ks} \in \{0,1\}\ \forall i,j,k \in V, s \in \mathcal{S} \\
& o_k, nns_s \in \mathbb{Z}
\end{array}\right. \qquad \text{FI-S}$$

## 3.3 An Alternate Formulation for Recourse Costs

In the formulation given in Section 3.2, the $pays_{ks}$ variables were used to determine if a recourse action should be taken at customer $k$ in scenario $s$. What this variable is really determining is the vertex at which $w_{ks}$ switches from 0 to 1 in scenario $s$. This is effectively measuring $w_{js} - w_{is}$ for some edge $ij \in E$, where $j$ is the vertex at which the recourse action is taken, so it is reasonable to assume that we can express $pays_{ks}$ as a product of $y_{ij}$ and $w_{js} - w_{is}$. Indeed, we have the following lemma.

**Lemma 3.5.** *If for some stochastic TSP solution $P$ represented by $[x\ y]^T$,*

$$pays_{ks} = \begin{cases} 1, & k = \gamma(P, s) \\ 0, & otherwise \end{cases}$$

*and*

$$w_{ks} = \begin{cases} 1, & k \textit{ is served at or after } \gamma(P, s) \\ 0, & otherwise \end{cases}$$

*then*

$$\sum_{k=2}^{n} pays_{ks}(c_{1k} + c_{k1}) = \sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}(w_{js} - w_{is})(c_{1j} + c_{j1}).$$

*Proof.* Let $P$ serve the customers in the order $1\kappa_1 \ldots \kappa_{n-1}1$ with $\kappa_0 = \kappa_n = 1$. Since

$$y_{ij} = \begin{cases} 1, & ij = \kappa_{t-1}\kappa_t \text{ for some } 1 \leq t \leq n \\ 0, & otherwise \end{cases}$$

it is clear that

$$\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}(w_{js} - w_{is})(c_{1j} - c_{j1}) = \sum_{t=1}^{n}(w_{\kappa_t s} - w_{\kappa_{t-1} s})(c_{1\kappa_t} - c\kappa_t 1).$$

From the definition of $w_{ks}$, we have

$$w_{\kappa_t s} = \begin{cases} 1, & t \geq \alpha(P, s) \\ 0, & otherwise \end{cases}$$

so since the refill occurs at $\gamma(P, s)$,

$$w_{\kappa_t s} - w_{\kappa_{t-1} s} = \begin{cases} 1, & t = \alpha(P, s) \\ 0, & otherwise \end{cases}$$

Then from the definition of $pays_{ks}$, we have

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}(w_{js} - w_{is})(c_{1j} - c_{j1}) &= \sum_{t=1}^{n}(w_{\kappa_t s} - w_{\kappa_{t-1} s})(c_{1\kappa_t} - c\kappa_t 1) \\ &= c_{1\gamma(P,s)} + c_{\gamma(P,s)} \\ &= \sum_{k=2}^{n} pays_{ks}(c_{1k} + c_{k1}).\end{aligned}$$

□

Using the substitution from Lemma 3.5 allows us to avoid setting the $pays_{ks}$ variables entirely, which removes constraints (3.2d), (3.2e), and (3.2f) and variables $o_k$, $nns_s$ and $pays_{ks}$. However, substituting directly in (3.2a) leads to a non-linear objective function, so we need another way of expressing this alternate objective function. Fortunately, there is a simple method of linearizing a product of binary variables by replacing them with a single binary variable, as given in [16]. In our case, we'll replace the product $y_{ij}(w_{js} - w_{is})$ by $z_{ijs}$. However, we'll have to do a little bit of extra work because $w_{js} - w_{is}$ can take the value -1 and is not binary. As a logical statement, what we want is that the recourse cost $c_{1j} + c_{j1}$ should be included if

- The edge $ij$ is included in the path we are concerned with ($y_{ij} = 1$)
- The vertex $i$ is served before the recourse action is taken in scenario $s$ ($w_{is} = 0$)
- The vertex $j$ is served after the recourse action is taken in scenario $s$ ($w_{js} = 1$)

Noting that for a binary variable $a$, $1 - a = \bar{a}$, we want to show that

$$\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}(w_{js} - w_{is})(c_{1k} + c_{k1}) = \sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}w_{js}(1 - w_{is})(c_{1k} + c_{k1}).$$

**Lemma 3.6.** *If for some stochastic TSP solution $P$ represented by $[x\, y]^T$,*

$$w_{ks} = \begin{cases} 1, & k \textit{ is served at or after } \gamma(P, s) \\ 0, & \textit{otherwise} \end{cases}$$

*then*

$$\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}(w_{js} - w_{is})(c_{1k} + c_{k1}) = \sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}w_{js}(1 - w_{is})(c_{1k} + c_{k1}).$$

*Proof.* As noted in Lemma 3.5, we have

$$y_{ij} = \begin{cases} 1, & ij = \kappa_{t-1}\kappa_t \text{ for some } 1 \le t \le n \\ 0, & \textit{otherwise} \end{cases}$$

so

$$\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}w_{js}(1 - w_{is})(c_{1k} + c_{k1}) = \sum_{t=1}^{n} w_{\kappa_t s}(1 - w_{\kappa_{t-1} s})(c_{1k} + c_{k1}).$$

From the definition of $w_{ks}$, we have

$$w_{\kappa_t s}(1 - w_{\kappa_{t-1} s}) = \begin{cases} 1, & t = \alpha(P, s) \\ 0, & \textit{otherwise} \end{cases}$$

so

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}w_{js}(1-w_{is})(c_{1j}+c_{j1}) &= \sum_{t=1}^{n} w_{\kappa_t s}(1-w_{\kappa_{t-1}s})(c_{1\kappa_t}+c_{\kappa_t 1})\\ &= c_{1\gamma(P,s)} + c_{\gamma(P,s)1}\\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}(w_{js}-w_{is})(c_{1j}+c_{j1}).\end{aligned}$$

□

With this conversion, we now have a product of binary variables in the objective function. Using this, we have the following alternate IP to compute the recourse costs:

$$\varphi(x,y) = \begin{cases} \min & \sum_{s\in\mathcal{S}}\sum_{i=1}^{n}\sum_{j=1}^{n} \mathbb{P}[s] z_{ijs}(c_{1j}+c_{j1}) & \text{(3.3a)}\\ \text{s.t.} & (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \le C-1,\ \ \forall k\in\bar{V}, s\in\mathcal{S} & \text{(3.3b)}\\ & (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \ge 0,\ \ \forall k\in\bar{V}, s\in\mathcal{S} & \text{(3.3c)}\\ & z_{ijs} \le y_{ij},\ \ \forall i,j\in V, s\in\mathcal{S} & \text{(3.3d)}\\ & z_{ijs} \le w_{js},\ \ \forall i,j\in V, s\in\mathcal{S} & \text{(3.3e)}\\ & z_{ijs} \le 1-w_{is},\ \ \forall i,j\in V, s\in\mathcal{S} & \text{(3.3f)}\\ & z_{ijs} \ge y_{ij}+w_{js}-w_{is}-1,\ \ \forall i,j\in V, s\in\mathcal{S} & \text{(3.3g)}\\ & w_{ks}, z_{ijs}\in\{0,1\} & \text{(3.3h)}\end{cases}$$

It is quite easy to show that (3.3) is equivalent to (3.2).

**Lemma 3.7.** *(3.2) and (3.3) are equivalent.*

*Proof.* From lemmas 3.5 and 3.6, it is clear that as long as the $w_{ks}$ variables are set correctly and $z_{ijs} = y_{ij}w_{js}(1-w_{is})$, then the two formulations are equivalent. The $w_{ks}$ variables are correct by Lemma 3.4, and by [16], constraints (3.3d) through (3.3g) set $z_{ijs} = y_{ij}w_{js}(1-w_{is})$, so the formulations are equivalent. □

Since this gives a correct formulation, we give the full alternate IP formulation:

$$\begin{cases} \min & \sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}c_{ij} + \sum_{s\in\mathcal{S}}\sum_{i=1}^{n}\sum_{j=1}^{n} \mathbb{P}[s] z_{ijs}(c_{1j}+c_{j1}) \\ \text{s.t.} & \sum_{j=1}^{n} y_{ij} = 1,\ \ \forall i \in V \\ & \sum_{i=1}^{n} y_{ij} = 1,\ \ \forall j \in V \\ & \sum_{j=1}^{n} x_{1jk} = 1,\ \ \forall k \in V \\ & \sum_{j=1}^{n} x_{ijk} - \sum_{j=1}^{n} x_{jik} = f(i,k)\ \ \forall i,k \in V \\ & x_{ijk} - y_{ij} \le 0\ \ \forall i,j,k \in V \\ & (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \le C-1,\ \ \forall k \in \bar{V}, s\in\mathcal{S} \\ & (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \ge 0,\ \ \forall k \in \bar{V}, s\in\mathcal{S} \\ & z_{ijs} \le y_{ij},\ \ \forall i,j \in V, s\in\mathcal{S} \\ & z_{ijs} \le w_{js},\ \ \forall i,j \in V, s\in\mathcal{S} \\ & z_{ijs} \le 1 - w_{is},\ \ \forall i,j \in V, s\in\mathcal{S} \\ & z_{ijs} \ge y_{ij} + w_{js} - w_{is} - 1,\ \ \forall i,j \in V, s\in\mathcal{S} \\ & y_{ij}, x_{ijk}, w_{ks}, z_{ijs} \in \{0,1\}\ \ \forall i,j,k \in V, s\in\mathcal{S} \end{cases} \qquad \text{FA-S}$$

It is likely that (FA-S) will have a more useful LP relaxation in general. To see this, first note that the constraints in (FA-S) that are different from the ones given in (FI-S) are simply linearizing a product of binary variables. It can be shown that the linearization used in the final constraints of (FA-S) perform well in practice to approximate a product of binary variables [12]. Since these constraints work well in general, it is likely that they will work better than the constraints in (FI-S) which are designed to accomplish the same task. A computational comparison of the two formulations is given in Chapter 5.

# Chapter 4

# Multi-Vehicle Variant

For the multi-vehicle case, we base the recourse constraints more closely on the originals given in [15], and modify the TSP constraints much more. Unlike the single vehicle case, we do not concern ourselves too much with a range on the total demand. Since this formulation works for any number of vehicles, if the total demand is too high and the problem is infeasible, we can just increase the number of vehicles to compensate. Obviously for the recourse IP to be implemented in practice, a list of demand scenarios must be given, but for the following discussion we'll assume that there are enough vehicles that the problem is feasible.

## 4.1 TSP Formulation

The easiest way to extend the TSP constraints to the multi-vehicle variant of the problem is to simply change the right hand side of constraints (3.1b), (3.1c), and (3.1d) to $m$ when they refer to the depot to ensure that there are $m$ solutions starting and ending at the depot and that every customer is served exactly once by one vehicle, similar to MCF. Using the function

$$f(i,k) = \begin{cases} 1 & i = 1, k \neq 1 \\ -1 & i = k \neq 1 \\ 0 & otherwise \end{cases}$$

as defined in Chapter 3 and the new function

$$g(j,m) = \begin{cases} 1, & j \neq 1 \\ m, & j = 1 \end{cases},$$

the multi-TSP formulation is given by

$$\begin{cases} \min & \sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}c_{ij} & (4.1a) \\ \text{s.t.} & \sum_{j=1}^{n} y_{ij} = g(i,m), \ \ \forall i \in V & (4.1b) \\ & \sum_{i=1}^{n} y_{ij} = g(i,m), \ \ \forall j \in V & (4.1c) \\ & \sum_{j=1}^{n} x_{1jk} = g(i,m), \ \ \forall k \in V & (4.1d) \\ & \sum_{j=1}^{n} x_{ijk} - \sum_{j=1}^{n} x_{jik} = f(i,k) \ \ \forall i,k \in V & (4.1e) \\ & x_{ijk} - y_{ij} \leq 0 \ \ \forall i,j,k \in V & (4.1f) \\ & y_{ij}, x_{ijk} \in \{0,1\} \ \ \forall i,j,k \in V & (4.1g) \end{cases}$$

This formulation is almost the same as (3.1), but with $g(i,m)$ replacing 1 in the first three constraints to account for the $m$ vehicles leaving the depot.

## 4.2 TSP Formulation with Duplicated Variables

Unfortunately, (4.1) does not lend itself nicely to extending (3.2). The simplest way to give a proper set of recourse constraints is to simply make $m$ copies of each of the variables used in the recourse IP, where $m$ is the number of vehicles. Since the $x_{ijk}$ variables are included in the recourse IP, in order to replicate the constraints in the recourse IP we must also replicate the $x_{ijk}$ variables. If we replicate the $x_{ijk}$ variables without also duplicating the $y_{ij}$ variables, the linking between the two sets of variables doesn't work properly, so we also replicate the $y_{ij}$ variables. This gives the following variables for the multi-TSP IP:

- $y_{ijv}$ represents whether edge $ij \in P_v$.

- $x_{ijkv}$ represents whether edge $ij$ is on the path from 1 to $k$ along $P_v$. We define this to always be 0 if $k \notin P_v$.

The variables are replicated only so that the recourse costs can be computed similarly to the since vehicle case, not because it is necessary for the first stage IP. To avoid confusion, if $[x\ y]^T$ is the vector that represents an $m$-TSP solution with these replicated variables, we say that $[x\ y]^T$ represents $P$ multiply.

We also define the following function:

$$g(j,m) = \begin{cases} 1, & j \neq 1 \\ m, & j = 1 \end{cases},$$

not to be confused with

$$f(i,k) = \begin{cases} 1 & i = 1, k \neq 1 \\ -1 & i = k \neq 1 \\ 0 & \mathit{otherwise} \end{cases}$$

as defined in Chapter 3. With these replicated variables, we give the following multi-TSP problem IP formulation.

$$\begin{cases}
\min \quad \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{v \in \mathcal{V}} y_{ijv} c_{ij} & \text{(4.2a)} \\
\text{s.t.} \quad \sum_{v \in \mathcal{V}} \sum_{j=1}^{n} y_{ijv} = g(i,m), \ \ \forall i \in V & \text{(4.2b)} \\
\sum_{v \in \mathcal{V}} \sum_{i=1}^{n} y_{ijv} = g(j,m), \ \ \forall j \in V & \text{(4.2c)} \\
\sum_{j=1}^{n} y_{ijv} - \sum_{j=1}^{n} y_{jiv} = 0, \ \ \forall i \in V, v \in \mathcal{V} & \text{(4.2d)} \\
\sum_{v \in \mathcal{V}} \sum_{j=1}^{n} x_{1jkv} = g(k,m), \ \ \forall k \in V & \text{(4.2e)} \\
\sum_{j=1}^{n} x_{ijkv} - \sum_{j=1}^{n} x_{jikv} = f(i,k) \sum_{j=1}^{n} x_{1jkv} \ \ \forall i,k \in V, v \in \mathcal{V} & \text{(4.2f)} \\
\sum_{i=1}^{n} x_{1iiv} = 1, \ \ \forall v \in \mathcal{V} & \text{(4.2g)} \\
x_{ijkv} - y_{ijv} \leq 0 \ \ \forall i,j,k \in V & \text{(4.2h)} \\
y_{ijv}, x_{ijkv} \in \{0,1\} \ \ \forall i,j,k \in V & \text{(4.2i)}
\end{cases}$$

Note that this formulation is essentially MCF, but modified slightly for the replicated variables.

Now we show that these constraints really do correspond to the feasible solutions.

**Lemma 4.1.** *$[x\,y]^T$ corresponds to an $m$-TSP solution if and only if it satisfies constraints (4.2b) to (4.2i).*

*Proof.* ($\Longrightarrow$) Suppose that we have an $m$-TSP solution $P$, and let $[x\,y]^T$ represent $P$. We show that constraints (4.2b) to (4.2i) are satisfied by $[x\,y]^T$.

(4.2b): Each vertex $i \neq 1$ is part of exactly 1 route $P_v$ by constraints (2.2) and (2.3), so $\sum_{v \in \mathcal{V}} \sum_{j=1}^{n} y_{ijv} = 1$.

Vertex $1 \in V(P_v)\, \forall v \in \mathcal{V}$ by (2.3), and since these paths are edge-disjoint (also by (2.3)) we have $\sum_{v \in \mathcal{V}} \sum_{j=1}^{n} y_{1jv} = m$.

(4.2c): Follows by the same argument as for constraint (4.2b), with the sum over $y_{jiv}$ instead of $y_{ijv}$.

(4.2d): Let $i \in V$, $v \in \mathcal{V}$. If $i \in V(P_v)$ then there is an incoming and an outgoing edge at vertex $i$ along $P_v$ since $P_v$ is a cycle by (2.1). Then $\sum_{j=1}^n y_{ijv} = \sum_{j=1}^n y_{jiv} = 1$. If $i \notin V(P_v)$, then there are no incoming or outgoing edges at vertex $i$ along $P_v$, so $\sum_{j=1}^n y_{ijv} = \sum_{j=1}^n y_{jiv} = 0$. Thus, this constraint is satisfied.

(4.2e): Consider a vertex $k \neq 1$. Then $k \in V(P_v)$ for exactly one $v$ by (2.3). This path has exactly one outgoing edge $1j_v$ from vertex 1, so we have $x_{1j_vkv} = 1$. For every other choice of $j \neq j_v$ and $v' \neq v$, we have $x_{1jkv'} = 0$ since $1j \notin P_v$ and $k \notin V(P_{v'})$. Thus, $\sum_{v\in\mathcal{V}} \sum_{j=1}^n x_{1jkv} = 1$.

For $k = 1$, we have that $k \in V(P_v)\ \forall v \in \mathcal{V}$ by (2.3). For every route $P_v$, there is exactly one outgoing edge $1j_v$ from 1, so we have $x_{1j_v1v} = 1$ for all these choices of $j_v$ and $v$. Since for every choice of $v$, we have that $1j \notin P_v$ for $j \neq j_v$, we have $x_{1j1v} = 0$ for $j \neq j_v$. Thus, $\sum_{v\in\mathcal{V}} \sum_{j=1}^n x_{1j1v} = m$.

(4.2f): Fix vertices $i, k \in V$ and a vehicle $v \in \mathcal{V}$. First, if $k \notin V(P_v)$, then $x_{ijkv} = x_{jikv} = x_{1jkv} = 0$ for every $j \in V$, so both the left and right sides are 0 and the constraint is satisfied.

Now suppose that $k \in V(P_v)$. Then $\sum_{j=1}^n x_{1jkv} = 1$, so the right side is $f(i,k)$. If $i \notin V(P_v)$, then $x_{ijkv} = x_{jikv} = f(i,k) = 0$ for every $j \in V$, so the constraint is satisfied. The only remaining case is where $i, k \in V(P_v)$. We divide this case into further cases which are similar to the single vehicle variant.

- Let $i = 1$, $k \in \bar{V}$ so $f(i,k) = 1$. Then there is an edge $1j_1$ and an edge $j_21$ in $P_v$, but only $1j_1$ occurs before we reach $k$ along $P_v$. Since these are the only edges with vertex 1 as an endpoint in $P_v$, we have $\sum_{j=1}^n x_{1jkv} = 1$ and $\sum_{j=1}^n x_{j1kv} = 0$, so the constraint is satisfied.
- Let $i = k$, $k \in \bar{V}$ so $f(i,k) = -1$. Then there is an edge $ij_1$ and an edge $j_2i$ in $P_v$, but only $j_2i$ occurs before we reach $k$ along $P_v$. Since these are the only edges with vertex $i$ as an endpoint along $P_v$, we have $\sum_{j=1}^n x_{ijkv} = 0$ and $\sum_{j=1}^n x_{jikv} = 1$, so the constraint is satisfied.
- Let $i = k = 1$. Then $f(i,k) = 0$. There are $m$ incoming and $m$ outgoing edges from vertex 1 by (2.3). Since each of the routes $P_v$ start and end at vertex 1, all these edges are along the path from vertex 1 to vertex 1. This gives $\sum_{j=1}^n x_{1j1} = \sum_{j=1}^n x_{j11} = m$, so the constraint is satisfied.

- Let $i, k \in \bar{V}$, $i \neq k$. There is an edge $ij_1$ and an edge $j_2 i$ in $P_v$, and either both these edges occur before we reach $k$ along $P_v$ or neither of them do. Since these are the only edges with vertex i as an endpoint along $P_v$, we have either $\sum_{j=1}^{n} x_{1j1} = \sum_{j=1}^{n} x_{j11} = 1$ or $\sum_{j=1}^{n} x_{1j1} = \sum_{j=1}^{n} x_{j11} = 0$, so the constraint is satisfied.

(4.2g): Let $v \in \mathcal{V}$. By (2.1) there is exactly one edge $1j_v \in P_v$. Thus $x_{1j_vj_vv} = 1$ and $x_{1jjv} = 0$ for all $j \neq j_v$, so we have $\sum_{j=1}^{n} x_{1jjv} = 1$.

(4.2h): If $ij$ is on the path to $k$ served by vehicle $v$ for some $k \in V$ and $v \in \mathcal{V}$ (i.e. $x_{ijkv} = 1$), then $ij \in P_v \subset P$ and $y_{ijv} = 1$. Thus, this constraint is satisfied.

(4.2i): Clearly the variables are binary because there are indicators for the edges included in $P$.

($\Longleftarrow$) Now we need to show that if $[x\ y]^T$ satisfies these constraints, then it multiply represents an $m$-TSP solution $P$. For this to be true, we need to satisfy constraints (2.1) to (2.3). In addition, we need to verify that there is only one cycle corresponding to each vehicle and that the $x$ and $y$ variables define the same edge set $P$.

1. From (4.2b) and (4.2c), we have that for each $v$, the vector $\delta_v$ defined by $\delta_v(i,j) = \sum_{v \in \mathcal{V}} y_{ijv}$ describes an edge set made up only of cycles, and by the same argument as in the single vehicle variant we can conclude that $\delta_v$ describes exactly one cycle.

2. (4.2b) and (4.2c) ensure that for every $k \in V$, $k \in V(P_v)$ for some vehicle $v$, so $k \in V(P)$.

3. For $i \in \bar{V}$, (4.2b) says $\sum_{v \in \mathcal{V}} \sum_{j=1}^{n} y_{ijv} = 1$. Since the right hand side is 1, $i \in V(P_v)$ for at most one $v$, so $i \notin V(P_v) \cap V(P_w)$ for $v \neq w$.

   By (4.2g), we have $x_{1jjv} = 1$ for some $j$, and by (4.2h) we have $y_{1jv} = 1$ for this choice of $j$ and $v$. Thus, $1 \in V(P_v)$ for every $v \in \mathcal{V}$, so $1 \in V(P_v) \cap V(P_w)$ for every choice of $v, w \in \mathcal{V}$.

4. It is obvious that there is exactly one cycle corresponding to each vehicle, since (4.2g) ensures that there is exactly one outgoing edge from vertex 1 corresponding to each vehicle.

5. Suppose that for vehicle $v$, $y$ defines the path $1\kappa_1 \ldots \kappa_{r-1}1$, with $\kappa_0 = \kappa_r = 1$. If $x$ defines the same path as $y$ for this vehicle, we require that $x_{\kappa_{s-1}\kappa_s\kappa_t v} = 1$ for

$1 \leq s \leq t$, and $x_{ijkv} = 0$ for any other choice of $i$, $j$, and $k$. We show this in two parts. First, if $ij \neq \kappa_{s-1}\kappa_s$ for some $s$, then $y_{ijv} = 0$, which implies that $x_{ijkv} = 0$ for every $k$ by (4.2h). For $x$ variables of the form $x_{\kappa_{s-1}\kappa_s\kappa_t v}$, we first set $t$ and show that $x_{\kappa_{s-1}\kappa_s\kappa_t v} = 1$ if and only if $1 \leq s \leq t$. We do this by induction on $s$ using constraints (4.2e) and (4.2f). First, by (4.2e) and (4.2h) we have that $x_{\kappa_0\kappa_1\kappa_t v} = 1$. Now consider $1 < s \leq t$, and assume that $x_{\kappa_{s-2}\kappa_{s-1}\kappa_t v} = 1$. By (4.2f) with $i = \kappa_{s-1}$ and $k = \kappa_t$, we have that $\sum_{j=1}^{n} x_{\kappa_{s-1}j\kappa_t v} = 1$, and by (4.2h) this implies that $x_{\kappa_{s-1}\kappa_s\kappa_t v} = 1$. Next, consider $s = t+1$, and suppose that $x_{\kappa_{s-2}\kappa_{s-1}\kappa_t v} = 1$. By (4.2f), we have that $\sum_{j=1}^{n} x_{\kappa_{s-1}j\kappa_t v} = 0$, since $f(\kappa_t, \kappa_t) = -1$. Thus, $x_{\kappa_t\kappa_{t+1}\kappa_t v} = 0$. Finally, let $t+1 < s \leq r$, and suppose that $x_{\kappa_{s-2}\kappa_{s-1}\kappa_t v} = 0$. Then $\sum_{j=1}^{n} x_{j\kappa_{s-1}\kappa_t v} = 0$, so $\sum_{j=1}^{n} x_{\kappa_{s-1}j\kappa_t v} = 0$ by (4.2f). Thus, we have $x_{\kappa_{s-1}\kappa_s\kappa_t v} = 0$. From this analysis, we have that, given a vehicle $v$ that follows the path $1\kappa_1 \ldots \kappa_{r-1}1$ according to the $y$ vector, $x_{ijkv} = 1$ if and only if $i = \kappa_{s-1}$, $j = \kappa_s$, and $k = \kappa_t$ for $1 \leq s \leq t$.

Thus, we have satisfied the requirements for $[x\, y]^T$ to represent an $m$-TSP solution. $\square$

We require for the recourse IP that every $m$-TSP solution we pass in should also be a stochastic $m$-TSP solution, but this is dealt with in the recourse constraints.

## 4.3 Multi-Vehicle Recourse Constraints

To modify the original recourse constraints for the multi-vehicle variant of the problem, we largely follow the model in [15]. First, we replicate the variables used, similarly to the multi-TSP formulation. We also need to replace $n$ with the number of customers along the path for vehicle $v$, $n_v$. To do this we'll need another constraint to set $n_v$ for all $v \in \mathcal{V}$. This gives a list of variables as follows:

- $o_{iv}$: contains the position of vertex $i$ along $P_v$. We define this to be 0 if $i \notin V(P_v)$.
- $n_v$: $|P_v|$.
- $w_{ksv}$: indicates whether vertex $i$ is served at or after $\gamma(P_v, s)$ in scenario $s$ along $P_v$. We define this to be 0 if $k \notin V(P_v)$.
- $p^+_{ksv}$: a variable used to determine the number of vertices before $\gamma(P_v, s)$ after vertex $k$ in scenario $s$ along $P_v$. This is always non-negative, and only one of $p^+_{ksv}$ and $p^-_{ksv}$ can be non-zero.

- $p^-_{ksv}$: a variable used to determine the number of vertices after $\gamma(P_v, s)$ before vertex $k$ in scenario $s$ along $P_v$. This is always non-negative, and at most one of $p^+_{ksv}$ and $p^-_{ksv}$ can be non-zero.
- $sign_{ksv}$: a variable used to ensure that $p^+_{ksv}$ and $p^-_{ksv}$ are both non-negative.
- $pays_{ksv}$: indicates whether the vehicle goes from client $k$ to the depot and returns in scenario $s$ along the path covered by vehicle $v$. We have

$$pays_{ksv} = \begin{cases} 1, & \text{a recourse action is taken at vertex } k \text{ by vehicle } v \text{ in scenario } s \\ 0, & otherwise \end{cases}$$

Doing this allows us to simply take most of the original constraints from (3.2) and copy them so we have one of each constraint per vehicle. However, we need to be a little bit more clever in setting the $pays_{ks}$ variables because simply replacing $n$ with $n_v$ in (3.2f) makes the constraint non-linear. Instead, we'll use a slight variation on the idea in [15]. The full formulation is given as follows:

$$\Xi(x) = \begin{cases} \min & \sum_{s\in\mathcal{S}}\sum_{k=2}^{n}\sum_{v\in\mathcal{V}} \mathbb{P}[s]pays_{ksv}(c_{1k}+c_{k1}) & \text{(4.3a)} \\ \text{st} & (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijkv}) - C\cdot w_{ksv} \le C-1,\ \forall k\in\bar{V}, s\in\mathcal{S},\ \forall v\in\mathcal{V} & \text{(4.3b)} \\ & (\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijkv}) - C\cdot w_{ksv} \ge 0,\ \forall k\in\bar{V}, s\in\mathcal{S},\ \forall v\in\mathcal{V} & \text{(4.3c)} \\ & (\sum_{i=1}^{n}\sum_{j=2}^{n} x_{ijkv}) - o_{kv} = 0,\ \forall k\in\bar{V},\ \forall v\in\mathcal{V} & \text{(4.3d)} \\ & (\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij1v}) - n_v = 0,\ \forall v\in\mathcal{V} & \text{(4.3e)} \\ & (\sum_{k=2}^{n} w_{ksv}) - nns_{sv} = 0,\ \forall s\in\mathcal{S},\ \forall v\in\mathcal{V} & \text{(4.3f)} \\ & o_{kv} + nns_{sv} + p^{+}_{ksv} - p^{-}_{ksv} = n_v, \forall k\in\bar{V}, s\in\mathcal{S},\ \forall v\in\mathcal{V} & \text{(4.3g)} \\ & p^{+}_{ksv} - n\cdot sign_{ksv} \le 0,\ \forall k\in\bar{V}, s\in\mathcal{S},\ \forall v\in\mathcal{V} & \text{(4.3h)} \\ & p^{-}_{ksv} + n\cdot sign_{ksv} \le n,\ \forall k\in\bar{V}, s\in\mathcal{S},\ \forall v\in\mathcal{V} & \text{(4.3i)} \\ & pay_{ksv} + p^{+}_{ksv} + p^{-}_{ksv} \ge 1,\ \forall k\in\bar{V}, s\in\mathcal{S},\ \forall v\in\mathcal{V} & \text{(4.3j)} \\ & o_{kv}, n_v, p^{+}_{ksv}, p^{-}_{ksv} \in \mathbb{Z}\forall k\in\bar{V}, s\in\mathcal{S}, v\in\mathcal{V} & \text{(4.3k)} \\ & w_{ksv}, sign_{ksv}, pays_{ksv} \in \{0,1\}\forall k\in\bar{V}, s\in\mathcal{S}, v\in\mathcal{V} & \text{(4.3l)} \end{cases}$$

Before showing that only stochastic $m$-TSP solutions are feasible for (4.3), we show a useful lemma about the $x_{ijkv}$ variables corresponding to $m$-TSP solutions.

**Lemma 4.2.** *Suppose that $P$ is an $m$-TSP solution multiply represented by $[x\ y]^T$. Let $v \in \mathcal{V}$ and suppose that $P_v$ serves its customers in the order $1\kappa_1 \dots \kappa_{r-1}1$. Let $k = \kappa_t$. Given constants $\{a_j : j \in \bar{V}(P_v)\}$, we have*

$$\sum_{l=1}^{t} a_j = \sum_{i=1}^{n}\sum_{j=2}^{n} a_j x_{ijkv}$$

*Proof.* This follows easily from the reverse argument in Lemma 4.1, since it is shown there that $x_{ijkv} = 1$ if and only if $ij = \kappa_{s-1}\kappa_s$ for some $1 \le s \le t$. Thus,

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=2}^{n} a_j x_{ijkv} &= \sum_{l=1}^{t} a_{\kappa_l} x_{\kappa_{l-1}\kappa_l kv} \\ &= \sum_{l=1}^{t} a_{\kappa_l}\end{aligned}$$

□

Using this lemma, we can show easily that only stochastic $m$-TSP solutions can be feasible for (4.3).

**Lemma 4.3.** *Given an $m$-TSP solution $P$ multiply represented by $[x\, y]^T$, (4.3) with input $x$ is feasible if and only if $P$ is a stochastic $m$-TSP solution.*

*Proof.* ($\Longleftarrow$) Let $v \in \mathcal{V}$, let $P_v$ serve the customers in the order $1\kappa_1 \ldots \kappa_{r-1}1$, and let $k = \kappa_t \in V(P_v)$. From the definitions of each of the variables, we set them as follows:

$$x_{ijkv} = \begin{cases} 1, & \text{if } ij \in P_v, j \text{ comes before } k \text{ along } P \text{ or } j = k \\ 0, & otherwise \end{cases}$$

$$w_{ksv} = \begin{cases} 1, & \text{if } t \geq \alpha(P_v, s) \\ 0, & otherwise \end{cases}$$

$$pays_{ksv} = \begin{cases} 1, & \text{if } k = \gamma(P_v, s) \\ 0, & otherwise \end{cases}$$

$$p^+_{ksv} = \begin{cases} \alpha(P_v, s) - t, & t \leq \alpha(P_v, s) \\ 0, & otherwise \end{cases}$$

$$p^-_{ksv} = \begin{cases} t - \alpha(P_v, s), & t \geq \alpha(P_v, s) \\ 0, & otherwise \end{cases}$$

$$sign_{ksv} = \begin{cases} 1, & p^+_{ksv} > 0 \\ 0, & otherwise \end{cases}$$

and

$$n_v = r, nns_{sv} = n_v - \alpha(P_v, s), o_{kv} = t.$$

By the definition of $\gamma(P_v, s)$ and Lemma 4.2, we have that

$$0 \leq \sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ijkv} = \sum_{l=1}^{t} d_{\kappa_l s} \leq C - 1$$

if $t < \alpha(P_v, s)$ and

$$C \leq \sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ijkv} = \sum_{l=1}^{t} d_{\kappa_l s} \leq 2C - 1$$

if $t \geq \alpha(P_v, s)$, so constraints (4.3b) and (4.3c) are both satisfied by $P_v$. Clearly we have

$$\sum_{i=1}^{n} \sum_{j=2}^{n} x_{ijkv} = t,$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij1v} = r,$$

and

$$\sum_{k=2}^{n} w_{ks} = (r-1) - (\alpha(P_v, s) - 1) = r - \alpha(P_v, s) = nns_s,$$

so constraints (4.3d), (4.3e), and (4.3f) are satisfied.

For $k \notin V(P_v)$, we let

$$x_{ijk} = o_{kv} = wksv = p^-_{ksv} = pays_{ksv} = 0, n_v = r, nns_s = n_v - \alpha(P_v, s), p^+_{ksv} = \alpha(P_v, s), sign_{ksv} = 1$$

Finally we examine constraints (4.3g) through (4.3j). If $t < \alpha(P_v, s)$, we have

$$o_{kv} + nns_{sv} + p^+_{ksv} - p^-_{ksv} = (t) + (r - \alpha(P_v, s)) + (\alpha(P_v, s) - t) - (0) = r = n_v,$$

$$p^+_{ksv} - n \cdot sign_{ksv} = (\alpha(P_v, s) - t) - n \cdot (1) = n + t - \alpha(P_v, s) \leq 0,$$

and

$$p^+_{ksv} + p^-_{ksv} + pays_{ksv} = (\alpha(P_v, s) - t) + (0) + (0) \geq 1,$$

so all the constraints are satisfied. Similarly, if $t > \alpha(P_v, s)$, we have

$$o_{kv} + nns_{sv} + p^+_{ksv} - p^-_{ksv} = (t) + (r - \alpha(P_v, s)) + (0) - (t - \alpha(P_v, s)) = r = n_v,$$

$$p^+_{ksv} - n \cdot sign_{ksv} = (0) - n \cdot (0) \leq 0,$$

$$p^-_{ksv} + n \cdot sign_{ksv} = (t - \alpha(P_v, s)) + n \cdot (0) = n + t - \alpha(P_v, s) \leq n,$$

and

$$p^+_{ksv} + p^-_{ksv} + pays_{ksv} = (0) + (t - \alpha(P_v, s)) + (0) \geq 1.$$

Finally, if $t = \alpha(P_v, s)$, we have

$$o_{kv} + nns_{sv} + p^+_{ksv} - p^-_{ksv} = (t) + (r - \alpha(P_v, s)) + (\alpha(P_v, s) - t) - (0) = r = n_v,$$

$$p^+_{ksv} - n \cdot sign_{ksv} = (\alpha(P_v, s) - t) - n \cdot (1) = n + t - \alpha(P_v, s) \leq 0,$$

$$p^-_{ksv} + n \cdot sign_{ksv} = (0) + n \cdot (1) \leq n,$$

and

$$p^+_{ksv} + p^-_{ksv} + pays_{ksv} = (\alpha(P_v, s) - t) + (0) + (1) = 1 \geq 1$$

and the constraints are still satisfied. Since (4.3k) is clearly satisfied, $\Xi(x)$ is feasible if $P$ is a stochastic $m$-TSP solution.

($\Longrightarrow$) Suppose that $P$ is not a stochastic $m$-TSP solution. Since $P$ is an $m$-TSP solution, it satisfies (2.1) to (2.3), so $P$ breaks constraint (2.5). This means that (2.5) is not satisfied for some vehicle $v \in \mathcal{V}$ and some scenario $s \in \mathcal{S}$. By definition, we have $\sum_{i \in \bar{V}(P_v)} d_{is} > 2C - 1$. Now let $P_v$ serve its customers in the order $1\kappa_1 \ldots \kappa_{r-1}1$, and let $k = \kappa_{r-1}$ be the last customer served by vehicle $v$. Then, by Lemma 4.2, we have

$$\sum_{i \in \bar{V}(P_v)} d_{is} = \sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ijkv} > 2C - 1,$$

so in order for (4.3b) to be satisfied for this choice of $k$, $s$, and $v$, we require that $w_{ksv} > 1$. However, this is not possible because we defined $w_{ksv}$ to be a binary variable in (4.3k), so $\Xi(x)$ is not feasible.

$\square$

Now that we have shown only stochastic $m$-TSP solutions are feasible for (4.3), we need to show that it actually gives the correct solution, which we show in the next lemma.

**Lemma 4.4.** *Given a solution $P$ multiply represented by $[x\ y]^T$, an optimal solution to (4.3) with input $x$ satisfies*

$$pays_{ksv} = \begin{cases} 1, & k = \gamma(P_v, s) \\ 0, & otherwise \end{cases}$$

*for every vehicle $v \in \mathcal{V}$, every demand scenario $s \in \mathcal{S}$, and every $k \in \bar{V}$.*

*Proof.* Let $P_v$ serve its customers in the order $1\kappa_1 \ldots \kappa_{r-1}1$, with $\kappa_0 = \kappa_r = 1$. Note that $r = |V(P_v)|$. As in the single variable case, we start by showing that

$$pays_{ksv} \geq \begin{cases} 1, & k = \gamma(P_v, s) \\ 0, & otherwise \end{cases}$$

We first ensure that $o_{kv}$, $nns_{sv}$, and $n_v$ are set properly, then show that the resulting $p^+_{ksv}$ and $p^-_{ksv}$ from (4.3g) set $pays_{ksv}$ correctly.

Consider a vertex $k \notin \bar{V}(P_v)$. Since $k \notin \bar{V}(P_v)$, we have $x_{ijkv} = 0$ for every edge $ij$, so by constraint (4.3d)

$$o_{kv} = \sum_{i=1}^{n} \sum_{j=2}^{n} x_{ijkv} = 0.$$

Now consider a vertex $k \in \bar{V}(P_v)$. Then $k = \kappa_t$ for some $1 \leq t \leq r - 1$. As shown in Lemma 4.3, we have $o_{kv} = t$, so $o_{kv}$ is set correctly. The correctness of $n_v$ follows immediately from the correctness of $o_{kv}$, since (4.3e) is simply (4.3d) with $k = 1$.

For $nns_{sv}$, we need to ensure that the $w_{ksv}$ variables are set correctly. We divide our analysis into three cases - $k \notin V(P_v)$, $k \in V(P_v)$ is before $\gamma(P_v, s)$, and $k \in V(P_v)$ is at or after $\gamma(P_v, s)$.

Suppose that $k \notin \bar{V}(P_v)$. As noted before, we have $x_{ijkv} = 0$ for every edge $ij$, so we have

$$\sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ijkv} = 0$$

and $w_{ksv} = 0$ from constraint (4.3c).

Now suppose that $u = \kappa_a = \gamma(P_v, s)$ is the refill point. Then we have

$$\sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ijuv} = \sum_{l=1}^{a} d_{\kappa_l s} \geq C$$

and

$$\sum_{i=1}^{n} \sum_{j=2}^{n} d_{js} x_{ij\kappa_{a-1}v} = \sum_{l=1}^{a-1} d_{\kappa_l s} \leq C - 1$$

by definition, unless $a = r$ in which case only the second inequality applies. Consider a vertex $k$ which is before the refill point. Then $k = \kappa_b$ with $1 \leq b \leq a - 1$, so we have

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijkv} &= \sum_{l=1}^{b} d_{\kappa_l s}\\ &\leq \sum_{l=1}^{a-1} d_{\kappa_l s}\\ &\leq C-1.\end{aligned}$$

Thus by constraint (4.3c) we have $w_{ksv} = 0$.

Conversely, consider a vertex $k$ which is at or after the refill point. Then $k = \kappa_b$ with $a \leq b \leq r-1$, so we have

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijkv} &= \sum_{l=1}^{b} d_{\kappa_l s}\\ &\geq \sum_{l=1}^{a} d_{\kappa_l s}\\ &\geq C.\end{aligned}$$

Thus by constraint (4.3b) we have $w_{ksv} = 1$, and $w_{ksv}$ is set correctly for every vertex $k \in \bar{V}$. Since $w_{ksv}$ is set correctly for every vertex $k \in \bar{V}$, it is clear that $nns_{sv}$ is set correctly.

Let the refill point be $u = \kappa_a = \gamma(P_v, s)$ with $1 \leq a \leq r$. Then $nns_s = (r-1)-(a-1) = r-a$, and $n_v = |V(P_v)| = r$. We consider constraints (4.3g) through (4.3j) in 4 cases - $k \notin \bar{V}(P_v)$, $k$ is served before the refill, $k$ is served after the refill, and $k = \gamma(P_v, s)$.

Suppose that $k \notin \bar{V}(P_v)$. Then $o_{kv} = 0$, so $o_{kv} + nns_{sv} = r - a$. Since, as noted in (4.3), at most one of $p^+_{ksv}$ and $p^-_{ksv}$ is positive, we have $p^+_{ksv} = a$ and $p^-_{ksv} = 0$ by constraints (4.3g) through (4.3i). As $p^+_{ksv} > 0$, $pays_{ksv}$ can be 0.

Now suppose that $k \in \bar{V}(P_v)$, and that $k$ is served before the refill. Then $k = \kappa_b$ for some $b < a$, and we have

$$o_{kv} + nns_{sv} = b + (r-a) = r + (b-a) < r = n_v$$

Thus, constraint (4.3g) sets $p^+_{ksv} = a - b$, so $p^+_{ksv} + p^-_{ksv} > 0$ and $pays_{ksv}$ can be 0.

Similarly, if $k$ is served after the refill we have $k = \kappa_b$ for some $b > a$, so

$$o_{kv} + nns_{sv} = b + (r - a) = r + (b - a) > r = n_v$$

Thus, constraint (4.3g) sets $p^-_{ksv} = b - a$, so $p^+_{ksv} + p^-_{ksv} > 0$ and $pays_{ksv}$ can be 0.

Finally, if $k$ is the refill point, we have $k = \kappa_a$, so

$$o_k + nns_{sv} = a + (r - a) = r = n_v$$

Thus, constraint (4.3g) sets $p^+_{ksv} = p^-_{ksv} = 0$, so $p^+_{ksv} + p^-_{ksv} = 0$ and $pays_{ksv}$ can be 0.

From this argument, we see that

$$pays_{ksv} \geq \begin{cases} 1, & k = \gamma(P_v, s) \\ 0, & otherwise \end{cases}$$

Finally, we show that if this inequality holds strictly for some vertex $k$ and some scenario $s$, then the solution is not optimal. Now consider a solution $p_1$ for which $pays_{ksv} = 1$ for some vehicle $v \in \mathcal{V}$, scenario $s \in \mathcal{S}$, and some $k \neq \gamma(P_v, s)$. Then we can create a new solution $p_2$ by setting $pays_{ksv} = 0$ and leave all other variables the same, since we are required only that $pays_{ksv} \geq 0$. But this new solution $p_2$ has cost less than that of $p_1$ by $\mathbb{P}[s](c_{1k} + c_{k1})$, so $p_1$ cannot be optimal. Thus, if $pays_{ksv} = 1$ for some $k \neq \gamma(P_v, s)$ for some feasible solution to (4.3), then that solution is not optimal, so since

$$pays_{ksv} \geq \begin{cases} 1, & k = \gamma(P_v, s) \\ 0, & otherwise \end{cases}$$

for any feasible solution for every $k \in \bar{V}$, $s \in \mathcal{S}$, and $\in \mathcal{V}$ we have that

$$pays_{ksv} = \begin{cases} 1, & k = \gamma(P_v, s) \\ 0, & otherwise \end{cases}$$

for the optimal solution. □

This shows that (4.3) gives the correct recourse cost given a stochastic $m$-TSP solution $P$ represented by $[x\, y]^T$. This means that a full formulation for the 2SCVRP can be given as follows:

$$
\begin{cases}
\min & \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{v\in\mathcal{V}} y_{ijv}c_{ij} + \sum_{s\in\mathcal{S}}\sum_{k=2}^{n}\sum_{v\in\mathcal{V}} \mathbb{P}[s]pays_{ksv}(c_{1k}+c_{k1}) \qquad \text{FI-M}\\
\text{s.t.} & \displaystyle\sum_{v\in\mathcal{V}}\sum_{j=1}^{n} y_{ijv} = g(i,m),\ \ \forall i\in V\\
& \displaystyle\sum_{v\in\mathcal{V}}\sum_{i=1}^{n} y_{ijv} = g(j,m),\ \ \forall j\in V\\
& \displaystyle\sum_{j=1}^{n} y_{ijv} - \sum_{j=1}^{n} y_{jiv} = 0,\ \ \forall i\in V, v\in\mathcal{V}\\
& \displaystyle\sum_{v\in\mathcal{V}}\sum_{j=1}^{n} x_{1jkv} = g(k,m),\ \ \forall k\in V\\
& \displaystyle\sum_{j=1}^{n} x_{ijkv} - \sum_{j=1}^{n} x_{jikv} = f(i,k)\sum_{j=1}^{n} x_{1jkv}\ \ \forall i,k\in V, v\in\mathcal{V}\\
& \displaystyle\sum_{i=1}^{n} x_{1iiv} = 1,\ \ \forall v\in\mathcal{V}\\
& \displaystyle(\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijkv}) - C\cdot w_{ksv} \le C-1,\ \ \forall k\in\bar{V}, s\in\mathcal{S},\ \ \forall v\in\mathcal{V}\\
& \displaystyle(\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijkv}) - C\cdot w_{ksv} \ge 0,\ \ \forall k\in\bar{V}, s\in\mathcal{S},\ \ \forall v\in\mathcal{V}\\
& \displaystyle(\sum_{i=1}^{n}\sum_{j=2}^{n} x_{ijkv}) - o_{kv} = 0,\ \ \forall k\in\bar{V},\ \ \forall v\in\mathcal{V}\\
& \displaystyle(\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij1v}) - n_v = 0,\ \ \forall v\in\mathcal{V}\\
& \displaystyle(\sum_{k=2}^{n} w_{ksv}) - nns_{sv} = 0,\ \ \forall s\in\mathcal{S},\ \ \forall v\in\mathcal{V}\\
& o_{kv} + nns_{sv} + p^{+}_{ksv} - p^{-}_{ksv} = n_v, \forall k\in\bar{V}, s\in\mathcal{S},\ \ \forall v\in\mathcal{V}\\
& p^{+}_{ksv} - n\cdot sign_{ksv} \le 0,\ \ \forall k\in\bar{V}, s\in\mathcal{S},\ \ \forall v\in\mathcal{V}\\
& p^{-}_{ksv} + n\cdot sign_{ksv} \le n,\ \ \forall k\in\bar{V}, s\in\mathcal{S},\ \ \forall v\in\mathcal{V}\\
& pay_{ksv} + p^{+}_{ksv} + p^{-}_{ksv} \ge 1,\ \ \forall k\in\bar{V}, s\in\mathcal{S},\ \ \forall v\in\mathcal{V}\\
& x_{ijkv} - y_{ijv} \le 0\ \ \forall i,j,k\in V \quad 42\\
& y_{ijv}, x_{ijkv}, w_{ksv}, sign_{ksv}, pays_{ksv} \in \{0,1\}\ \ \forall i,j,k\in V, s\in\mathcal{S}, v\in\mathcal{V}\\
& o_{kv}, n_v, p^{+}_{ksv}, p^{-}_{ksv} \in \mathbb{Z}\forall k\in\bar{V}, s\in\mathcal{S}, v\in\mathcal{V}
\end{cases}
$$

## 4.4 An Alternate Formulation for Multi Vehicle Recourse Costs

Similarly to the single vehicle case, we can express the multi vehicle recourse cost in an alternate way using a product of binary variables. Because there is nothing vehicle specific about determining the refill point in this way (unlike (4.3), which requires $n_v$), we can avoid having to copy the variables for each vehicle. Instead, we can just use the multi-TSP formulation given in (4.1). As a result, we have the following lemma.

**Lemma 4.5.** *If for some stochastic m-TSP solution $P$ is represented by $[x\ y]^T$,*

$$pays_{ks} = \begin{cases} 1, & k = \gamma(P_v, s) \\ 0, & otherwise \end{cases}$$

*and*

$$w_{ks} = \begin{cases} 1, & k \text{ is served by vehicle } v \text{ at or after } \gamma(P_v, s) \text{ for some } v \\ 0, & otherwise \end{cases}$$

*then*

$$\sum_{v \in \mathcal{V}} \sum_{k=2}^{n} pays_{ksv}(c_{1k} + c_{k1}) = \sum_{i=1}^{n} \sum_{j=1}^{n} y_{ij} w_{js}(1 - w_{is})(c_{1k} + c_{k1}).$$

*Proof.* This is clear from Lemmas 3.5 and 3.6, since

$$\sum_{v \in \mathcal{V}} pays_{ksv} = pays_{ks}$$

with $pays_{ks}$ is as defined in 3.5. □

From Lemma 4.5, we see that if $[xy]^T$ is an $m$-TSP solution without replicated variables, we can use (3.3) for the multi vehicle case as well.

**Lemma 4.6.** *Given some m-TSP solution $P$ represented by $[x\ y]^T$, (3.3) gives the correct recourse cost for $P$.*

*Proof.* Similarly to the single vehicle case, we need only show that the $w_{ks}$ variables are set correctly and that $z_{ijs} = y_{ij}w_{js}(1 - w_{is})$, since by Lemmas 4.5 and 4.4 (3.3a) is the correct objective function. We have a bit of extra work to do to show that the $w_{ks}$ variables are correct.

Let $v \in \bar{V}$, and suppose $P_v$ serves its customers in the order $1\kappa_1 \dots \kappa_{r-1}1$. We should have that $w_{ks} = 1$ if and only if it is served at or after $\gamma(P_v, s)$ for some vehicle $v$. Now let $k \in \bar{V}$, and suppose that $k$ is served by vehicle $v$ which serves its customers in the order $1\kappa_1 \dots \kappa_{r-1}1$. By the definition of $\gamma(P_v, s)$, we expect that $w_{ks} = 1$ if and only if $\sum_{j=1}^{i} d_{\kappa_j s} \geq C$. Since $\sum_{j=1}^{i} d_{\kappa_j s} = \sum_{l=1}^{n}\sum_{j=2}^{n} d_{js}x_{lj\kappa_i}$, as shown in Lemma 3.3, this is true by constraints (3.3b) and (3.3c). Thus, the $w_{ks}$ variables are set appropriately.

As in the single vehicle case, constraints (3.3d) through (3.3g) give $z_{ijs} = y_{ij}w_{js}(1 - w_{is})$ by [16], so (3.3) gives the correct recourse cost for $P$. □

As mentioned above, we can use the multi-TSP formulation given in (4.1). Using this TSP formulation, we can write the full two-stage IP as

$$
\left\{
\begin{array}{ll}
\min & \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij}c_{ij} + \sum_{s\in\mathcal{S}}\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbb{P}[s]z_{ijs}(c_{1j}+c_{j1}) \qquad\qquad \text{FA-M} \\
\text{s.t.} & \displaystyle\sum_{j=1}^{n} y_{ij} = g(i,m),\ \ \forall i \in V \\
& \displaystyle\sum_{i=1}^{n} y_{ij} = g(i,m),\ \ \forall j \in V \\
& \displaystyle\sum_{j=1}^{n} x_{1jk} = g(i,m),\ \ \forall k \in V \\
& \displaystyle\sum_{j=1}^{n} x_{ijk} - \sum_{j=1}^{n} x_{jik} = f(i,k)\ \ \forall i,k \in V \\
& x_{ijk} - y_{ij} \le 0\ \ \forall i,j,k \in V \\
& \displaystyle(\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \le C-1,\ \ \forall k \in \bar{V}, s\in\mathcal{S} \\
& \displaystyle(\sum_{i=1}^{n}\sum_{j=2}^{n} d_{js}x_{ijk}) - C\cdot w_{ks} \ge 0,\ \ \forall k \in \bar{V}, s\in\mathcal{S} \\
& z_{ijs} \le y_{ij},\ \ \forall i,j \in V, s\in\mathcal{S} \\
& z_{ijs} \le w_{js},\ \ \forall i,j \in V, s\in\mathcal{S} \\
& z_{ijs} \le 1 - w_{is},\ \ \forall i,j \in V, s\in\mathcal{S} \\
& z_{ijs} \ge y_{ij} + w_{js} - w_{is} - 1,\ \ \forall i,j \in V, s\in\mathcal{S} \\
& y_{ij}, x_{ijk}, w_{ks}, z_{ijs} \in \{0,1\}\ \ \forall i,j,k \in V, s\in\mathcal{S}
\end{array}
\right.
$$

# Chapter 5

# Computational Results

Since there are multiple formulations given to solve the problem, it is important to compare them empirically. In order to decide what should be compared between the two formulations, we should look at the steps an IP solver will take to solve each instance. First, we note that both formulations will be solved using the branch-and-cut method, since both are edge-based formulations. The first important thing to know is that the time required to solve an IP can be dependent on seemingly unrelated things, so comparing IP solve times is likely not the best comparison. Some causes of variability in IP solve times are given in [4]. To determine the next-best comparison, we note that branch-and-bound IP solvers solve successively more restrictive LP approximations of the given IP. Then the two things that are important are

1. The solve time of each of the LP approximations. Decreasing the time required for each iteration of the branch-and-bound process will decrease the overall time required to solve the IP if the number of LPs solved stays constant.

2. The difference between the optimal value of the LP approximations and the optimal value of the IP. Decreasing this difference will decrease the number of iterations of this branch-and-bound process that are required to solve the IP.

Since the LP approximations solved in the branch-and-bound method are similar to the LP relaxations of the original IPs, comparing the solve times and optimal values of the LP relaxations of the IPs should give a good idea of how their IP solve times will compare.

Before continuing, we give a note on comparing the gap between formulations. Normally to compare the optimal values two formulations, we would calculate the gap

$1 - \frac{\text{LP value}}{\text{MIP value}}$

for each individual formulation so the MIP value is used as the baseline for the comparison. In this case the gap is calculated for both formulations and the results are compared. However, since the MIP value is only given if it is solvable in under an hour some instances do not have a MIP cost, we'll do the comparison as

$\frac{FA{-}S/FA{-}M \text{ Value}}{FI{-}S/FI{-}M \text{ Value}} - 1$

so that the value of the FI-S/FI-M is used as the baseline for the comparison. We use these formulations as the base because for every instance, they give a lower objective value than the alternate formulation. For each instance, the gap will be given along with the two optimal values.

Each instance of the 2SCVRP is defined by a graph and a scenario set. The instances are not from a standard set of instances in literature because the standard instances have deterministic scenarios. They were designed with a couple of ideas in mind. The graphs with fewer than 10 vertices are crafted more carefully to ensure that the optimal solution avoids as many long edges as possible. They are generally a mix of edges that are less than 5 units long and ones around 10 units long. This is done to make sure that the vehicle tends towards the shorter edges in the first stage solution. The graphs with 10 vertices or more have edge lengths in a greater range, and are intended largely for testing the run time of the two formulations. The 10 vertex graph is taken from the CVRP library (instance E-n13-k4) [17], which is a Euclidean instance, and the larger graphs are expansions of the 10 vertex graph. A Euclidean instance is one generated by placing points on a grid and creating an adjacency matrix from the distances between them rounded down to the nearest integer, and graphs were expanded by adding customers in clusters near the existing customers. The graphs are divided into large and small in this way because the 7 vertex graph was the largest MIP that could be solved for every number of vehicles that was tested.

For each graph size, there are three different scenario sets. The first group of scenario sets is designed with small demands to allow for a small vehicle capacity. Specifically, for 6 vertices we include all the possible distributions which give a total demand of 4, and for larger graphs we give scenarios of a similar flavour to the sets used for the 6 vertex case so that pockets of customers have demands similar to the 6 vertex scenarios. There are not enough distributions in the 5 vertex case, so the scenarios for 5 vertex graphs are separate from the rest. Similar to the 6 vertex graphs, we simply include all possible distributions which give a total demand of 4. The second and third groups of scenario sets are designed to require a higher capacity. Each scenario is a vector which is first generated uniformly on the simplex $\{x \in \mathbb{R}^{n-1} : ||x||_1 = 99, x \geq 0\}$, then has each component rounded down to the nearest integer. The number 99 is used so that a capacity of 50 can be used in the tests.

The only difference between the second and third groups of scenario sets is the number of scenarios - 40 scenarios in the second set and 100 in the third set. For easy reference in the result tables, we denote the three groups of scenario sets by M-S, R-40, and R-100.

The reasoning for choosing these specific groups of scenario sets is to make two specific comparisons between the two formulations. The first is to make a comparison between the results with low capacities (M-S) and high capacities (R-40) with a similar number of scenarios. The second is to make a comparison between the results with the same capacities but a different number of scenarios (R-100).

The following sections give empirical results for both formulations in the single and multi-vehicle cases. The code for the tests is written in Julia [11], the LP solver is Gurobi 9.0.0 [10], and the tests were run on a Intel Core i7 CPU M 620 @ 2.67GHz quad-core processor with 7.8 GB of memory. For the sake of space, many of the column names are abbreviated. The abbreviations used are the following: nV represents the number of vertices in the graph, Cap represents the capacity, T(s) represents the time in seconds, and LP represents the optimal value of the corresponding LP. Also note that FI-S is the formulation based on the presentation in [15], FA-S is the formulation based on a product of binary variables, and FI-M and FA-M are their multi-vehicle counterparts. The value of the MIP will be denoted by simply "MIP" to avoid confusion.

## 5.1 Single Vehicle Results

The computational results for the single vehicle variant of the problem are summarized in tables 5.1 to 5.3. For both formulations of the problem, we give the amount of time taken to solve the LP and the optimal value of the LP. When the MIP is solvable in under an hour, the value of the MIP is given for comparison. The graphs and demand scenarios used are given in appendices A and B respectively, and the capacity required for a particular instance of the problem is also given when necessary.

| Graph | nV | Cap | FI-S T(s) | FA-S T(s) | FI-S LP | FA-S LP | Gap | MIP |
|---|---|---|---|---|---|---|---|---|
| A | 5 | 3 | 0.00 | 0.01 | 10.00 | 10.00 | 0.0% | 15.50 |
| B | 5 | 3 | 0.00 | 0.01 | 13.00 | 13.00 | 0.0% | 25.00 |
| C | 5 | 3 | 0.00 | 0.01 | 21.00 | 21.00 | 0.0% | 33.00 |
| D | 6 | 3 | 0.01 | 0.04 | 9.00 | 9.17 | 1.89% | 16.27 |
| E | 7 | 3 | 0.04 | 0.28 | 41.00 | 41.00 | 0.0% | 45.24 |
| F | 10 | 3 | 0.14 | 0.24 | 110.50 | 113.29 | 2.52% | 170.02 |
| G | 12 | 7 | 0.20 | 1.01 | 102.50 | 103.83 | 1.30% | N/A |
| H | 15 | 7 | 2.09 | 2.73 | 87.00 | 87.00 | 0.0% | N/A |
| I | 20 | 10 | 12.90 | 8.21 | 88.28 | 88.39 | 0.12% | N/A |
| J | 30 | 15 | 108.81 | 163.27 | 97.19 | 97.21 | 0.02% | N/A |
| K | 40 | 20 | 368.71 | 839.52 | 108.00 | 108.00 | 0.0% | N/A |

Table 5.1: Results for 1 Vehicle, M-S

| Graph | nV | FI-S T(s) | FA-S T(s) | FI-S LP | FA-S LP | Gap | MIP |
|---|---|---|---|---|---|---|---|
| A | 5 | 0.01 | 0.32 | 10.00 | 10.00 | 0.0% | 17.60 |
| B | 5 | 0.01 | 0.31 | 13.00 | 13.03 | 0.23% | 27.80 |
| C | 5 | 0.01 | 0.03 | 21.00 | 21.00 | 0.0% | 35.80 |
| D | 6 | 0.02 | 0.47 | 9.03 | 9.22 | 2.10% | 17.15 |
| E | 7 | 0.04 | 0.29 | 41.00 | 41.00 | 0.0% | 45.70 |
| F | 10 | 0.11 | 0.19 | 110.50 | 111.98 | 1.34% | 163.35 |
| G | 12 | 0.19 | 1.18 | 102.00 | 102.21 | 0.21% | 150.08 |
| H | 15 | 0.46 | 0.75 | 85.00 | 85.00 | 0.0% | N/A |
| I | 20 | 1.54 | 2.11 | 87.00 | 87.00 | 0.0% | N/A |
| J | 30 | 16.60 | 44.90 | 97.00 | 97.00 | 0.0% | N/A |
| K | 40 | 116.77 | 659.11 | 108.00 | 108.00 | 0.0% | N/A |

Table 5.2: Results for 1 Vehicle, R-40, Capacity 50

| Graph | nV | FI-S T(s) | FA-S T(s) | FI-S LP | FA-S LP | Gap | MIP |
|---|---|---|---|---|---|---|---|
| A | 5 | 0.07 | 0.09 | 10.00 | 10.00 | 0.0% | 16.60 |
| B | 5 | 0.04 | 0.09 | 13.00 | 13.08 | 0.62% | 25.72 |
| C | 5 | 0.04 | 0.08 | 21.00 | 21.04 | 0.19% | 33.72 |
| D | 6 | 0.04 | 0.15 | 9.00 | 9.23 | 2.56% | 16.78 |
| E | 7 | 0.15 | 0.29 | 41.00 | 41.00 | 0.0% | 45.16 |
| F | 10 | 0.29 | 0.52 | 110.50 | 111.78 | 1.16% | 163.79 |
| G | 12 | 0.53 | 1.92 | 102.00 | 102.25 | 0.25% | N/A |
| H | 15 | 1.13 | 1.76 | 85.00 | 85.00 | 0.0% | N/A |
| I | 20 | 3.07 | 3.57 | 87.00 | 87.00 | 0.0% | N/A |
| J | 30 | 33.34 | 41.26 | 97.00 | 97.00 | 0.0% | N/A |
| K | 40 | 176.36 | 230.86 | 108.00 | 108.00 | 0.0% | N/A |

Table 5.3: Results for 1 Vehicle, R-100, Capacity 50

Regarding the run time of the formulations, there isn't a definitive answer for which formulation is superior. However, there are a couple general trends that can be noted. The first is that the run time of (FI-S) scales better than the initial formulation in the size of the graph. FA-S is 50% more than FI-S on average for the 15 vertex instances, but the difference increases to 208% for the 40 vertex instance. Here the 15 vertex is used as the low vertex comparison because it is the smallest number of vertices for which both formulations have a run time of over 1 second for at least one choice of scenario set so that the results are more significant. Another trend for both formulations the impact of adding scenarios or increasing capacity seems dependent on the formulation. For FI-S, increasing the capacity decreases the solve time while increasing the number of scenarios increases the solve time. On the other hand, for FA-S increasing capacity and number of scenarios have erratic effects on the solve time that are dependent on the associated graph.

Regarding the gap, the comparisons we are interested in are between a low and high number of vertices and between the three groups of scenario sets (M-S, R-40, and R-100), and we'll compare the average gap across all instances in a given category. For example, when looking at the gap for scenarios M-S, we take the gap for each individual instance using a scenario from M-S and average them to get the average gap for scenario group M-S. Here a low number of vertices is less than 10 and a high number is greater than or equal to 10, the same as the split in the graph construction. The instances with a low number of vertices have a larger average gap than the instances with a high number of vertices, with the former having a gap of 0.51% and the latter a gap of 0.38%. Regarding the groups of scenario sets, the tests on the scenario sets M-S have the highest average gap, with a gap of 0.53% versus 0.35% for R-40 and 0.43% for R-100.

## 5.2 Multi Vehicle Results

Using the same graphs and demand distributions as in the single variable case, the results are summarized in the tables below. The general format of the tables is similar to the single vehicle case - for each instance, the values and solve times are given for both formulations, the capacity is given when necessary, and the MIP value is given when the solve time is under an hour. In addition, there are results for each instance for both 2 and 4 vehicles.

| Graph | nV | Cap | FI-M T(s) | FA-M T(s) | FI-M LP | FA-M LP | Gap | MIP |
|---|---|---|---|---|---|---|---|---|
| A | 5 | 3 | 0.01 | 0.01 | 12.00 | 12.00 | 0.0% | 16.88 |
| B | 5 | 3 | 0.01 | 0.01 | 14.00 | 14.00 | 0.0% | 18.38 |
| C | 5 | 3 | 0.01 | 0.01 | 22.00 | 22.00 | 0.0% | 26.25 |
| D | 6 | 3 | 0.09 | 0.03 | 11.00 | 11.67 | 6.09% | 16.27 |
| E | 7 | 3 | 0.11 | 0.30 | 41.00 | 41.00 | 0.0% | 44.51 |
| F | 10 | 3 | 0.30 | 0.28 | 136.00 | 140.47 | 3.29% | 186.88 |
| G | 12 | 5 | 0.71 | 1.01 | 121.00 | 125.13 | 3.41% | N/A |
| H | 15 | 5 | 1.16 | 2.41 | 104.00 | 105.74 | 1.67% | N/A |
| I | 20 | 5 | 5.90 | 9.91 | 106.00 | 106.95 | 0.90% | N/A |
| J | 30 | 8 | 64.88 | 101.57 | 115.00 | 115.39 | 0.34% | N/A |
| K | 40 | 10 | 266.05 | 526.45 | 126.00 | 126.00 | 0.0% | N/A |

Table 5.4: Results for 2 Vehicles, M-S

| Graph | nV | Cap | FI-M T(s) | FA-M T(s) | FI-M LP | FA-M LP | Gap | MIP |
|---|---|---|---|---|---|---|---|---|
| A | 5 | 3 | 0.02 | 0.00 | 26.00 | 26.00 | 0.0% | 26.00 |
| B | 5 | 3 | 0.02 | 0.00 | 40.00 | 40.00 | 0.0% | 40.00 |
| C | 5 | 3 | 0.02 | 0.00 | 40.00 | 40.00 | 0.0% | 40.00 |
| D | 6 | 3 | 0.40 | 0.02 | 23.00 | 23.00 | 0.0% | 23.88 |
| E | 7 | 3 | 0.14 | 0.04 | 41.00 | 41.00 | 0.0% | 41.80 |
| F | 10 | 3 | 0.42 | 0.11 | 219.00 | 222.38 | 1.54% | N/A |
| G | 12 | 3 | 1.14 | 0.21 | 196.00 | 199.73 | 1.90% | N/A |
| H | 15 | 3 | 2.87 | 0.78 | 162.00 | 164.87 | 1.77% | N/A |
| I | 20 | 3 | 8.28 | 6.96 | 146.00 | 146.50 | 0.34% | N/A |
| J | 30 | 5 | 67.98 | 41.83 | 153.00 | 153.40 | 0.26% | N/A |
| K | 40 | 5 | 404.58 | 471.60 | 163.00 | 163.00 | 0.0% | N/A |

Table 5.5: Results for 4 Vehicles, M-S

| Graph | nV | FI-M T(s) | FA-M T(s) | FI-M LP | FA-M LP | Gap | MIP |
|---|---|---|---|---|---|---|---|
| A | 5 | 0.02 | 0.02 | 12.00 | 12.16 | 1.33% | 19.50 |
| B | 5 | 0.02 | 0.02 | 14.00 | 14.16 | 1.14% | 24.30 |
| C | 5 | 0.03 | 0.02 | 22.00 | 22.00 | 0.0% | 32.30 |
| D | 6 | 0.05 | 0.03 | 11.00 | 11.24 | 2.18% | 18.35 |
| E | 7 | 0.11 | 0.28 | 41.00 | 41.00 | 0.0% | 45.20 |
| F | 10 | 0.29 | 0.17 | 136.00 | 138.51 | 1.85% | 179.23 |
| G | 12 | 0.56 | 1.06 | 121.00 | 121.60 | 0.50% | 163.20 |
| H | 15 | 0.75 | 0.50 | 104.00 | 104.00 | 0.0% | 104.19 |
| I | 20 | 5.21 | 1.97 | 106.00 | 106.00 | 0.0% | N/A |
| J | 30 | 53.27 | 23.97 | 115.00 | 115.00 | 0.0% | N/A |
| H | 40 | 251.96 | 424.67 | 126.00 | 126.00 | 0.0% | N/A |

Table 5.6: Results for 2 Vehicles, R-40, Capacity 50

| Graph | nV | FI-M T(s) | FA-M T(s) | FI-M LP | FA-M LP | Gap | MIP |
|---|---|---|---|---|---|---|---|
| A | 5 | 0.05 | 0.01 | 26.00 | 27.05 | 4.04% | 29.10 |
| B | 5 | 0.07 | 0.01 | 40.00 | 41.67 | 4.18% | 44.80 |
| C | 5 | 0.05 | 0.01 | 40.00 | 41.67 | 4.18% | 44.80 |
| D | 6 | 0.10 | 0.03 | 23.00 | 23.09 | 0.39% | 26.25 |
| E | 7 | 0.15 | 0.08 | 41.00 | 41.04 | 0.10% | 43.05 |
| F | 10 | 0.38 | 0.17 | 219.00 | 220.65 | 0.75% | 239.18 |
| G | 12 | 0.74 | 0.26 | 196.00 | 196.14 | 0.07% | 204.80 |
| H | 15 | 1.39 | 0.39 | 162.00 | 162.00 | 0.0% | 166.33 |
| I | 20 | 9.59 | 1.78 | 146.00 | 146.00 | 0.0% | 146.00 |
| J | 30 | 85.98 | 17.80 | 153.00 | 153.00 | 0.0% | 153.00 |
| K | 40 | 507.52 | 197.20 | 163.00 | 163.00 | 0.0% | 163.00 |

Table 5.7: Results for 4 Vehicles, R-40, Capacity 50

| Graph | nV | FI-M T(s) | FA-M T(s) | FI-M LP | FA-M LP | Gap | MIP |
|---|---|---|---|---|---|---|---|
| A | 5 | 0.07 | 0.05 | 12.00 | 12.22 | 1.83% | 18.62 |
| B | 5 | 0.08 | 0.06 | 14.00 | 14.20 | 1.43% | 22.52 |
| C | 5 | 0.07 | 0.07 | 22.00 | 22.07 | 0.32% | 30.52 |
| D | 6 | 0.16 | 0.12 | 11.00 | 11.25 | 2.27% | 17.88 |
| E | 7 | 0.32 | 0.42 | 41.00 | 41.00 | 0.0% | 44.94 |
| F | 10 | 0.93 | 0.54 | 136.00 | 138.09 | 1.54% | 178.69 |
| G | 12 | 1.04 | 1.36 | 121.00 | 121.64 | 0.53% | 178.69 |
| H | 15 | 3.19 | 1.33 | 104.00 | 104.00 | 0.0% | 121.20 |
| I | 20 | 15.30 | 5.33 | 106.00 | 106.00 | 0.0% | N/A |
| J | 30 | 73.63 | 27.12 | 115.00 | 115.00 | 0.0% | N/A |
| K | 40 | 519.98 | 182.32 | 126.00 | 126.00 | 0.0% | N/A |

Table 5.8: Results for 2 Vehicles, R-100, Capacity 50

| Graph | nV | FI-M T(s) | FA-M T(s) | FI-M LP | FA-M LP | Gap | MIP |
|---|---|---|---|---|---|---|---|
| A | 5 | 0.14 | 0.04 | 26.00 | 26.85 | 3.27% | 29.26 |
| B | 5 | 0.14 | 0.04 | 40.00 | 41.18 | 2.95% | 44.76 |
| C | 5 | 0.15 | 0.04 | 40.00 | 41.18 | 2.95% | 44.76 |
| D | 6 | 0.28 | 0.08 | 23.00 | 23.14 | 0.61% | 25.48 |
| E | 7 | 0.36 | 0.20 | 41.00 | 41.01 | 0.02% | 42.30 |
| F | 10 | 0.93 | 0.35 | 219.00 | 220.22 | 0.56% | 236.55 |
| G | 12 | 1.61 | 0.61 | 196.00 | 196.27 | 0.14% | 204.95 |
| H | 15 | 4.20 | 1.00 | 162.00 | 162.02 | 0.01% | 164.58 |
| I | 20 | 19.22 | 2.60 | 146.00 | 146.00 | 0.0% | 146.46 |
| J | 30 | 105.87 | 23.75 | 153.00 | 153.00 | 0.0% | 153.00 |
| K | 40 | 1550.70 | 155.58 | 163.00 | 163.00 | 0.0% | N/A |

Table 5.9: Results for 4 Vehicles, R-100, Capacity 50

With results on the same instances and different vehicle counts, we can make some inferences about general trends. As with the single vehicle case, however, we cannot treat these as absolute trends, but rather issues to be investigated. The main trend that we see from the multi vehicle case is that FA-M scales better with the number of vehicles than FI-M - in the single vehicle case FI-M is solved faster in 32 of 33 instances, for 2 vehicles FA-M is solved faster in most instances (20 of 33), and for 4 vehicles FA-M is solved faster in 30 of 33 instances. One particularly interesting thing to note is that FA-M seems to reach an optimal solution faster when the number of vehicles is increased - in almost every instance, the 2 vehicle LP is solved faster than the 1 vehicle and the 4 vehicle is solved faster than the 2 vehicle. For example, for graph $K$ with scenario set R-100, the 1 vehicle instance is solved in 230.86 seconds, the 2 vehicle in 182.32 seconds, and the 4 vehicle in 155.58 seconds. It would be interesting in the future to investigate why this is the case, and whether it holds for more general instances and could be exploited in any way.

For the gap, we compare the results by number of vehicles, scenario group (M-S, R-40, or R-100), and low number of vertices vs. high (where low number of vertices is less than 10 and high is greater than or equal to 10). In all comparisons, as in the single vehicle case, the gap given is an average over all tests in a given category. First, the gap in the 2 vehicle tests is roughly equal to the gap in the 4 vehicle tests, with gaps of 0.93% and 0.91% respectively. Both are more than double the gap of 0.44% in the single vehicle tests. Given this result, it might be interesting to see if this is a feature of the formulations themselves instead of a result of increasing the number of vehicles, since the 2 and 4 vehicle tests use a different formulation than the 1 vehicle tests. The difference in gaps of the low vertex tests and high vertex tests increases drastically with the number of vehicles. In the 2 vehicle case, the average gap for low vertices is 1.11% and the gap for high vertices is 0.78%, while in the 4 vehicle case the gaps are 1.51% and 0.41%, respectively. This indicates that FA-M gives better approximations as the number of vehicles is increased, although more tests should be done to see if this is the case in general. Finally, as with the single variable case, the scenario group M-S has the largest gap, with a value of 0.83% across all instances (including the single vehicle ones) versus 0.74% for R-40 and 0.71% for R-100. The range across the three values is a little bit smaller than for the single vehicle case, even though the values themselves are higher. It might be interesting to look into whether the gap is generally larger when the capacities are smaller, and if the gap would decrease with an even larger capacity.

From the results of the tests, we can conclude that, at least in these instances, formulation (FA-S) is the more promising single vehicle formulation and (FA-M) is the more promising multi vehicle formulation. This conclusion is made based on the results for both the run time and optimal value for both formulations. The reasons for this are the following. Regarding run time, (FA-S) and (FA-M) scale better with run time than their counterparts in both number of customers and number of scenarios. Formulations (FI-S) and (FI-M) may be better for a smaller number of customers, since there is a larger disparity in the run times for a smaller number of vertices, but in general (FA-S) and (FA-M) gives a solution more quickly. Secondly, these formulation always give an equal or better approximation to the IP values than their counterparts, which should decrease the number of iterations required to solve its corresponding IP. Reducing the number of iterations required to solve the IP reduces the overall solve time of the IP, which is the ultimate goal.

# Chapter 6

# Conclusion

The 2SCVRP is a version of the vehicle routing problem given in a stochastic setting which is concerned with serving a group of customers while minimizing travel costs. The customers are served by vehicles with a given capacity which most start and end their routes at their depot, and are allowed to refill at their depot once in the middle of their route. In this paper, we started by giving a formal definition of the 2SCVRP. With this definition, we were able to give two formulations each for the single and multi vehicles versions of the problem, differing in their method of determining whether a refill is needed for a given vehicle and the cost of that refill if necessary. These formulations of the problem were compared empirically, with the single vehicle formulations ((FI-S) and (FA-S)) being compared to each other separately from the multi vehicle formulations ((FI-M) and (FA-M)). After comparing the formulations empirically, it was concluded that (FA-S) and (FA-M) show more promise of being used in practice for the single and multi vehicle problems respectively. Future work on the problem should include further verification that (FA-S) and (FA-M) are really the formulations that should be pursued for practical purposes, as well as improving both the formulations themselves and their implementation in code. One particular line of research is looking into implementing the formulations in code using capacity separation constraints to reduce the number of invalid routes verified by the TSP constraints.

# References

[1] A. Letchford, J. Salazar-Gonzalez. Stronger multi-commodity flow formulations of the capacitated vehicle routing problem. *European Journal of Operations Research*, 244:730–738, 2015.

[2] B. Gavish, S. Graves. The traveling salesman problem and related problems. *Working Paper OR-078-78, Operations Research Center, MIT, Cambridge, MA*, 1978. https://dspace.mit.edu/bitstream/handle/1721.1/5363/OR-078-78.pdf?sequence=1&isAllowed=y.

[3] C. Cetinkaya, I. Karaoglan, H. Gokcen. Two-stage vehicle routing problem with arc time windows: A mixed integer programming formulation and a heuristic approach. *European Journal of Operational Research*, 230(3):539–550, 2013.

[4] E. Danna. Performance variability in mixed integer programming. Presentation given at the Workshop on Mixed Integer Programming at Columbia University. https://coral.ise.lehigh.edu/mip-2008/talks/danna.pdf, August 2008.

[5] G. Clarke, J. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4):568–581, 1964.

[6] G. Dantzig, J. Ramser. The truck dispatching problem. *Management Science*, 6(1):80–91, 1959.

[7] G. Dantzig, R. Fulkerson, S. Johnson. Solution of a large-scale traveling-salesman problem. *Operations Research*, 2(4):393–410, 1954.

[8] G. Laporte, Y. Nobert. Exact algorithms for the vehicle routing problem. *Ann. Discrete Math*, 31:147–184, 1987.

[9] G. Laporte, Y. Nobert, M. Desrochers. Optimal routing under capacity and distance restrictions. *Operations Research*, 33(5):1050–1073, 1985.

[10] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2020. http://www.gurobi.com.

[11] J. Bezanson, A. Edelman, S. Karpinski, V. B. Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[12] J. Luedtke, M. Namazifar, J. Linderoth. Some results on the strength of relaxations of multilinear functions. *Mathematical Programming*, 136:325–351, 2012.

[13] L. Hvattum, A. Lokketangen, G. Laporte. Solving a dynamic and stochasic vehicle routing problem with a sample scenario hedging heuristic. *Transportation Science*, 40(4):421–438, 2006.

[14] M. Noorizadegan, B. Chen. Vehicle routing with probabilistic capacity constraints. *European Journal of Operational Research*, 270(2):544–555, 2018.

[15] M. Poggi, G. Spyrides. Routing with stochastic demands: A scenario approach. Presentation given at the International Conference on Stochastic Programming, July 2019.

[16] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I - convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.

[17] N. Christofides, S. Eilon. An algorithm for the vehicle-dispatching problem. *Operations Research*, 20(3):309–318, 1969.

[18] P. Toth, D. Vigo. *The vehicle routing problem*. SIAM, Philadelphia, PA, 2001.

[19] R. Baldacci, E. Hadjiconstantinou, A. Mingozzi. An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation. *Operations Research*, 52(5):723–738, 2004.

[20] S. Ghosal, W. Wiesemann. The distributionally robust chance constrainted vehicle routing problem. *Operations Research*, 68(3):716–732, 2019.

[21] S. Venkatachalam, K. Sundar, S. Rathinam. Two-stage stochastic programming model for routing multiple drones with fuel constraints. *Sensors*, 18(11), 2018.

[22] T. Dinh, R. Fukasawa, J. Luedtke. Exact algorithms for the chance-constrained vehicle routing problem. *Mathematical Programming*, 172(1-2):105–138, 2018.

# APPENDICES

## Appendix A - Graphs

Graph A: $\begin{bmatrix} 0 & 3 & 4 & 5 & 1 \\ 3 & 0 & 5 & 1 & 2 \\ 4 & 5 & 0 & 2 & 3 \\ 5 & 1 & 2 & 0 & 4 \\ 1 & 2 & 3 & 4 & 0 \end{bmatrix}$

Graph B: $\begin{bmatrix} 0 & 1 & 9 & 9 & 1 \\ 1 & 0 & 1 & 9 & 9 \\ 9 & 1 & 0 & 1 & 9 \\ 9 & 9 & 1 & 0 & 9 \\ 1 & 9 & 9 & 9 & 0 \end{bmatrix}$

Graph C: $\begin{bmatrix} 0 & 1 & 9 & 9 & 1 \\ 1 & 0 & 9 & 9 & 9 \\ 9 & 9 & 0 & 1 & 9 \\ 9 & 9 & 1 & 0 & 9 \\ 1 & 9 & 9 & 9 & 0 \end{bmatrix}$

Graph D: $\begin{bmatrix} 0 & 3 & 4 & 5 & 1 & 2 \\ 3 & 0 & 5 & 1 & 2 & 4 \\ 4 & 5 & 0 & 2 & 3 & 1 \\ 5 & 1 & 2 & 0 & 4 & 3 \\ 1 & 2 & 3 & 4 & 0 & 5 \\ 2 & 1 & 5 & 4 & 3 & 0 \end{bmatrix}$

$$\text{Graph E: } \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 3 & 4 & 5 & 6 & 7 \\ 2 & 2 & 0 & 5 & 6 & 7 & 8 \\ 3 & 4 & 5 & 0 & 7 & 8 & 9 \\ 4 & 5 & 6 & 7 & 0 & 9 & 10 \\ 5 & 6 & 7 & 8 & 9 & 0 & 11 \\ 6 & 7 & 8 & 9 & 10 & 11 & 0 \end{bmatrix}$$

$$\text{Graph F: } \begin{bmatrix} 0 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 \\ 50 & 0 & 5 & 12 & 22 & 21 & 24 & 31 & 35 & 37 \\ 41 & 49 & 0 & 7 & 17 & 16 & 23 & 26 & 30 & 36 \\ 36 & 44 & 46 & 0 & 21 & 30 & 27 & 37 & 43 & 31 \\ 37 & 39 & 19 & 28 & 0 & 35 & 41 & 29 & 31 & 29 \\ 9 & 10 & 16 & 22 & 20 & 0 & 30 & 7 & 11 & 13 \\ 17 & 25 & 27 & 10 & 16 & 10 & 0 & 20 & 6 & 6 \\ 14 & 19 & 12 & 12 & 20 & 8 & 10 & 0 & 6 & 20 \\ 24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 0 & 20 \\ 4 & 16 & 10 & 22 & 10 & 28 & 16 & 12 & 3 & 0 \end{bmatrix}$$

$$\text{Graph G: } \begin{bmatrix} 0 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 13 & 3 \\ 50 & 0 & 5 & 12 & 22 & 21 & 24 & 31 & 35 & 37 & 3 & 17 \\ 41 & 49 & 0 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 5 \\ 36 & 44 & 46 & 0 & 21 & 30 & 27 & 37 & 43 & 31 & 5 & 1 \\ 37 & 39 & 19 & 28 & 0 & 35 & 41 & 29 & 31 & 29 & 1 & 2 \\ 9 & 10 & 16 & 22 & 20 & 0 & 30 & 7 & 11 & 13 & 18 & 3 \\ 17 & 25 & 27 & 10 & 16 & 10 & 0 & 20 & 6 & 6 & 3 & 15 \\ 14 & 19 & 12 & 12 & 20 & 8 & 10 & 0 & 6 & 20 & 4 & 5 \\ 24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 0 & 20 & 5 & 1 \\ 4 & 16 & 10 & 22 & 10 & 28 & 16 & 12 & 3 & 0 & 1 & 2 \\ 19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 0 & 3 \\ 41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 0 \end{bmatrix}$$

$$\text{Graph H:}\quad \begin{bmatrix}
0 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 13 & 3 & 4 & 5 & 1 \\
50 & 0 & 5 & 12 & 22 & 21 & 24 & 31 & 35 & 37 & 3 & 17 & 5 & 1 & 2 \\
41 & 49 & 0 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 5 & 11 & 2 & 3 \\
36 & 44 & 46 & 0 & 21 & 30 & 27 & 37 & 43 & 31 & 5 & 1 & 2 & 10 & 4 \\
37 & 39 & 19 & 28 & 0 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 3 & 4 & 12 \\
9 & 10 & 16 & 22 & 20 & 0 & 30 & 7 & 11 & 13 & 18 & 3 & 4 & 5 & 1 \\
17 & 25 & 27 & 10 & 16 & 10 & 0 & 20 & 6 & 6 & 3 & 15 & 5 & 1 & 2 \\
14 & 19 & 12 & 12 & 20 & 8 & 10 & 0 & 6 & 20 & 4 & 5 & 14 & 2 & 3 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 0 & 20 & 5 & 1 & 2 & 17 & 4 \\
4 & 16 & 10 & 22 & 10 & 28 & 16 & 12 & 3 & 0 & 1 & 2 & 3 & 4 & 16 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 0 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 0 & 5 & 2 & 3 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 0 & 3 & 4 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 3 & 5 & 1 & 0 & 2 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 5 & 1 & 2 & 4 & 0
\end{bmatrix}$$

$$\text{Graph I:}\quad \begin{bmatrix}
0 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 \\
50 & 0 & 5 & 12 & 22 & 21 & 24 & 31 & 35 & 37 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 \\
41 & 49 & 0 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 \\
36 & 44 & 46 & 0 & 21 & 30 & 27 & 37 & 43 & 31 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 \\
37 & 39 & 19 & 28 & 0 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 \\
9 & 10 & 16 & 22 & 20 & 0 & 30 & 7 & 11 & 13 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 \\
17 & 25 & 27 & 10 & 16 & 10 & 0 & 20 & 6 & 6 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 \\
14 & 19 & 12 & 12 & 20 & 8 & 10 & 0 & 6 & 20 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 0 & 20 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 \\
4 & 16 & 10 & 22 & 10 & 28 & 16 & 12 & 3 & 0 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 0 & 3 & 4 & 5 & 1 & 20 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 0 & 5 & 2 & 3 & 10 & 20 & 5 & 2 & 3 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 0 & 3 & 4 & 12 & 7 & 6 & 3 & 4 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 3 & 5 & 1 & 0 & 2 & 1 & 12 & 3 & 13 & 2 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 5 & 1 & 2 & 4 & 0 & 2 & 4 & 1 & 14 & 10 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 3 & 4 & 5 & 1 & 20 & 0 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 5 & 2 & 3 & 10 & 20 & 0 & 5 & 2 & 3 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 3 & 4 & 12 & 7 & 6 & 0 & 3 & 4 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 3 & 5 & 1 & 2 & 1 & 12 & 3 & 13 & 0 & 2 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 5 & 1 & 2 & 4 & 2 & 4 & 1 & 14 & 10 & 0
\end{bmatrix}$$

$$\text{Graph J:}\quad \begin{bmatrix}
0 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 \\
50 & 0 & 5 & 12 & 22 & 21 & 24 & 31 & 35 & 37 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 \\
41 & 49 & 0 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 \\
36 & 44 & 46 & 0 & 21 & 30 & 27 & 37 & 43 & 31 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 \\
37 & 39 & 19 & 28 & 0 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 \\
9 & 10 & 16 & 22 & 20 & 0 & 30 & 7 & 11 & 13 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 \\
17 & 25 & 27 & 10 & 16 & 10 & 0 & 20 & 6 & 6 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 \\
14 & 19 & 12 & 12 & 20 & 8 & 10 & 0 & 6 & 20 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 0 & 20 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 \\
4 & 16 & 10 & 22 & 10 & 28 & 16 & 12 & 3 & 0 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 0 & 3 & 4 & 5 & 1 & 20 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 0 & 5 & 2 & 3 & 10 & 20 & 5 & 2 & 3 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 0 & 3 & 4 & 12 & 7 & 6 & 3 & 4 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 3 & 5 & 1 & 0 & 2 & 1 & 12 & 3 & 13 & 2 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 5 & 1 & 2 & 4 & 0 & 2 & 4 & 1 & 14 & 10 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 3 & 4 & 5 & 1 & 20 & 0 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 5 & 2 & 3 & 10 & 20 & 0 & 5 & 2 & 3 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 3 & 4 & 12 & 7 & 6 & 0 & 3 & 4 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 3 & 5 & 1 & 2 & 1 & 12 & 3 & 13 & 0 & 2 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 5 & 1 & 2 & 4 & 2 & 4 & 1 & 14 & 10 & 0 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 0 & 3 & 4 & 5 & 1 & 20 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 4 & 0 & 5 & 2 & 3 & 10 & 20 & 5 & 2 & 3 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 1 & 2 & 0 & 3 & 4 & 12 & 7 & 6 & 3 & 4 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 3 & 5 & 1 & 0 & 2 & 1 & 12 & 3 & 13 & 2 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 5 & 1 & 2 & 4 & 0 & 2 & 4 & 1 & 14 & 10 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 3 & 4 & 5 & 1 & 20 & 0 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 4 & 5 & 2 & 3 & 10 & 20 & 0 & 5 & 2 & 3 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 1 & 2 & 3 & 4 & 12 & 7 & 6 & 0 & 3 & 4 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 3 & 5 & 1 & 2 & 1 & 12 & 3 & 13 & 0 & 2 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 5 & 1 & 2 & 4 & 2 & 4 & 1 & 14 & 10 & 0
\end{bmatrix}$$

$$\text{Graph K:}\quad \begin{bmatrix}
0 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 \\
50 & 0 & 5 & 12 & 22 & 21 & 24 & 31 & 35 & 37 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 \\
41 & 49 & 0 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 \\
36 & 44 & 46 & 0 & 21 & 30 & 27 & 37 & 43 & 31 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 \\
37 & 39 & 19 & 28 & 0 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 \\
9 & 10 & 16 & 22 & 20 & 0 & 30 & 7 & 11 & 13 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 \\
17 & 25 & 27 & 10 & 16 & 10 & 0 & 20 & 6 & 6 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 \\
14 & 19 & 12 & 12 & 20 & 8 & 10 & 0 & 6 & 20 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 0 & 20 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 \\
4 & 16 & 10 & 22 & 10 & 28 & 16 & 12 & 3 & 0 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 0 & 3 & 4 & 5 & 1 & 20 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 0 & 5 & 2 & 3 & 10 & 20 & 5 & 2 & 3 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 0 & 3 & 4 & 12 & 7 & 6 & 3 & 4 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 3 & 5 & 1 & 0 & 2 & 1 & 12 & 3 & 13 & 2 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 5 & 1 & 2 & 4 & 0 & 2 & 4 & 1 & 14 & 10 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 3 & 4 & 5 & 1 & 20 & 0 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 4 & 5 & 2 & 3 & 10 & 20 & 0 & 5 & 2 & 3 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 1 & 2 & 3 & 4 & 12 & 7 & 6 & 0 & 3 & 4 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 3 & 5 & 1 & 2 & 1 & 12 & 3 & 13 & 0 & 2 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 5 & 1 & 2 & 4 & 2 & 4 & 1 & 14 & 10 & 0 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 0 & 3 & 4 & 5 & 1 & 20 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 4 & 0 & 5 & 2 & 3 & 10 & 20 & 5 & 2 & 3 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 1 & 2 & 0 & 3 & 4 & 12 & 7 & 6 & 3 & 4 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 3 & 5 & 1 & 0 & 2 & 1 & 12 & 3 & 13 & 2 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 5 & 1 & 2 & 4 & 0 & 2 & 4 & 1 & 14 & 10 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 3 & 4 & 5 & 1 & 20 & 0 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 4 & 5 & 2 & 3 & 10 & 20 & 0 & 5 & 2 & 3 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 1 & 2 & 3 & 4 & 12 & 7 & 6 & 0 & 3 & 4 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 3 & 5 & 1 & 2 & 1 & 12 & 3 & 13 & 0 & 2 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 0 & 16 & 5 & 1 & 2 & 4 & 2 & 4 & 1 & 14 & 10 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 13 & 3 & 4 & 5 & 1 & 0 & 3 & 4 & 5 & 1 & 20 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 3 & 17 & 5 & 1 & 2 & 4 & 0 & 5 & 2 & 3 & 10 & 20 & 5 & 2 & 3 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 4 & 5 & 11 & 2 & 3 & 1 & 2 & 0 & 3 & 4 & 12 & 7 & 6 & 3 & 4 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 5 & 1 & 2 & 10 & 4 & 3 & 5 & 1 & 0 & 2 & 1 & 12 & 3 & 13 & 2 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 1 & 2 & 3 & 4 & 12 & 5 & 1 & 2 & 4 & 0 & 2 & 4 & 1 & 14 & 10 \\
19 & 14 & 21 & 23 & 22 & 25 & 32 & 36 & 38 & 42 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 18 & 3 & 4 & 5 & 1 & 3 & 4 & 5 & 1 & 20 & 0 & 3 & 4 & 5 & 1 \\
41 & 49 & 15 & 7 & 17 & 16 & 23 & 26 & 30 & 36 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 3 & 15 & 5 & 1 & 2 & 4 & 5 & 2 & 3 & 10 & 20 & 0 & 5 & 2 & 3 \\
37 & 39 & 19 & 28 & 25 & 35 & 41 & 29 & 31 & 29 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 4 & 5 & 14 & 2 & 3 & 1 & 2 & 3 & 4 & 12 & 7 & 6 & 0 & 3 & 4 \\
17 & 25 & 27 & 10 & 16 & 10 & 18 & 20 & 6 & 6 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 5 & 1 & 2 & 17 & 4 & 3 & 5 & 1 & 2 & 1 & 12 & 3 & 13 & 0 & 2 \\
24 & 6 & 16 & 14 & 21 & 7 & 13 & 20 & 16 & 20 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 1 & 2 & 3 & 4 & 16 & 5 & 1 & 2 & 4 & 2 & 4 & 1 & 14 & 10 & 0
\end{bmatrix}$$

# Appendix B - Demand Scenarios

The sets of demand scenarios are given as matrices. They are formatted as follows: given a matrix $A = a_{is}$, the entry $a_{is}$ is the demand for customer $i$ in scenario $s$. Note that vertex 1 always has demand 0, since it is the depot.

$$\text{5 Vertex: } \begin{bmatrix} 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 1 & 1 & 2 \\ 0 & 0 & 1 & 2 & 1 \\ 0 & 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 0 & 2 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 2 & 0 \\ 0 & 1 & 2 & 0 & 1 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 2 & 0 & 0 & 2 \\ 0 & 2 & 0 & 1 & 1 \\ 0 & 2 & 0 & 2 & 0 \\ 0 & 2 & 1 & 0 & 1 \\ 0 & 2 & 1 & 1 & 0 \\ 0 & 2 & 2 & 0 & 0 \end{bmatrix}$$

$$\text{6 Vertex:}\quad \begin{bmatrix}
0 & 0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 1 & 1 & 2 \\
0 & 0 & 0 & 1 & 2 & 1 \\
0 & 0 & 1 & 0 & 1 & 2 \\
0 & 0 & 1 & 0 & 2 & 1 \\
0 & 0 & 1 & 1 & 0 & 2 \\
0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 2 & 0 \\
0 & 0 & 1 & 2 & 0 & 1 \\
0 & 0 & 1 & 2 & 1 & 0 \\
0 & 0 & 2 & 0 & 0 & 2 \\
0 & 0 & 2 & 0 & 1 & 1 \\
0 & 0 & 2 & 0 & 2 & 0 \\
0 & 0 & 2 & 1 & 0 & 1 \\
0 & 0 & 2 & 1 & 1 & 0 \\
0 & 0 & 2 & 2 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 2 \\
0 & 1 & 0 & 0 & 2 & 1 \\
0 & 1 & 0 & 1 & 0 & 2 \\
0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 2 & 0 \\
0 & 1 & 0 & 2 & 0 & 1 \\
0 & 1 & 0 & 2 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 2 \\
0 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 1 & 0 & 2 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 2 & 0 & 0 \\
0 & 1 & 2 & 0 & 0 & 1 \\
0 & 1 & 2 & 0 & 1 & 0 \\
0 & 1 & 2 & 1 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 2 \\
0 & 2 & 0 & 0 & 1 & 1 \\
0 & 2 & 0 & 0 & 2 & 0 \\
0 & 2 & 0 & 1 & 0 & 1 \\
0 & 2 & 0 & 1 & 1 & 0 \\
0 & 2 & 1 & 0 & 0 & 1 \\
0 & 2 & 1 & 0 & 1 & 0 \\
0 & 2 & 1 & 1 & 0 & 0 \\
0 & 2 & 2 & 0 & 0 & 0
\end{bmatrix}$$

$$\text{7 Vertex: } \begin{bmatrix}
0 & 0 & 0 & 0 & 2 & 2 & 1 \\
0 & 0 & 0 & 1 & 1 & 2 & 0 \\
0 & 0 & 0 & 1 & 2 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 2 & 0 \\
0 & 0 & 1 & 0 & 2 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 2 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 2 & 0 & 0 \\
0 & 0 & 1 & 2 & 0 & 1 & 1 \\
0 & 0 & 1 & 2 & 1 & 0 & 0 \\
0 & 0 & 2 & 0 & 0 & 2 & 1 \\
0 & 0 & 2 & 0 & 1 & 1 & 0 \\
0 & 0 & 2 & 0 & 2 & 0 & 1 \\
0 & 0 & 2 & 1 & 0 & 1 & 0 \\
0 & 0 & 2 & 1 & 1 & 0 & 1 \\
0 & 0 & 2 & 2 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 2 & 1 \\
0 & 1 & 0 & 0 & 2 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 2 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 2 & 0 & 1 \\
0 & 1 & 0 & 2 & 0 & 1 & 0 \\
0 & 1 & 0 & 2 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 2 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 1 & 0 & 2 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 2 & 0 & 0 & 1 \\
0 & 1 & 2 & 0 & 0 & 1 & 0 \\
0 & 1 & 2 & 0 & 1 & 0 & 1 \\
0 & 1 & 2 & 1 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 2 & 1 \\
0 & 2 & 0 & 0 & 1 & 1 & 0 \\
0 & 2 & 0 & 0 & 2 & 0 & 1 \\
0 & 2 & 0 & 1 & 0 & 1 & 0 \\
0 & 2 & 0 & 1 & 1 & 0 & 1 \\
0 & 2 & 1 & 0 & 0 & 1 & 0 \\
0 & 2 & 1 & 0 & 1 & 0 & 1 \\
0 & 2 & 1 & 1 & 0 & 0 & 0 \\
0 & 2 & 2 & 0 & 0 & 0 & 1
\end{bmatrix}$$

$$\text{10 Vertex: } \begin{bmatrix}
0 & 0 & 0 & 0 & 2 & 2 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 2 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 & 2 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 2 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 2 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 2 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 2 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 1 & 0 \\
0 & 0 & 2 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 2 & 0 & 2 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 2 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 2 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 2 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 2 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 2 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 2 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 2 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 1 & 2 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 2 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
0 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 \\
0 & 2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 2 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 2 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 2 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{bmatrix}$$

$$
\text{12 Vertex:}\quad
\begin{bmatrix}
0&0&0&0&2&2&1&0&0&0&0&0&2&2&2\\
0&0&0&1&1&2&0&1&0&0&0&1&1&2&2\\
0&0&0&1&2&1&0&0&1&0&0&1&2&1&1\\
0&0&1&0&1&2&0&0&0&1&1&0&1&2&2\\
0&0&1&0&2&1&1&0&0&0&1&0&2&1&1\\
0&0&1&1&0&2&0&1&0&0&1&1&0&2&2\\
0&0&1&1&1&1&0&0&1&0&1&1&1&1&1\\
0&0&1&1&2&0&0&0&0&1&1&1&2&0&0\\
0&0&1&2&0&1&1&0&0&0&1&2&0&1&1\\
0&0&1&2&1&0&0&1&0&0&1&2&1&0&0\\
0&0&2&0&0&2&0&0&1&0&2&0&0&2&2\\
0&0&2&0&1&1&0&0&0&1&2&0&1&1&1\\
0&0&2&0&2&0&1&0&0&0&2&0&2&0&0\\
0&0&2&1&0&1&0&1&0&0&2&1&0&1&1\\
0&0&2&1&1&0&0&0&1&0&2&1&1&0&0\\
0&0&2&2&0&0&0&0&0&1&2&2&0&0&0\\
0&1&0&0&1&2&1&0&0&0&0&0&1&2&2\\
0&1&0&0&2&1&0&1&0&0&0&0&2&1&1\\
0&1&0&1&0&2&0&0&1&0&0&1&0&2&2\\
0&1&0&1&1&1&0&0&0&1&0&1&1&1&1\\
0&1&0&1&2&0&1&0&0&0&0&1&2&0&0\\
0&1&0&2&0&1&0&1&0&0&0&2&0&1&1\\
0&1&0&2&1&0&0&0&1&0&0&2&1&0&0\\
0&1&1&0&0&2&0&0&0&1&1&0&0&2&2\\
0&1&1&0&1&1&1&0&0&0&1&0&1&1&1\\
0&1&1&0&2&0&0&1&0&0&1&0&2&0&0\\
0&1&1&1&0&1&0&0&1&0&1&1&0&1&1\\
0&1&1&1&1&0&0&0&0&1&1&1&1&0&0\\
0&1&1&2&0&0&1&0&0&0&1&2&0&0&0\\
0&1&2&0&0&1&0&1&0&0&2&0&0&1&1\\
0&1&2&0&1&0&0&0&1&0&2&0&1&0&0\\
0&1&2&1&0&0&0&0&0&1&2&1&0&0&0\\
0&2&0&0&0&2&1&0&0&0&0&0&0&2&2\\
0&2&0&0&1&1&0&1&0&0&0&0&1&1&1\\
0&2&0&0&2&0&0&0&1&0&0&0&2&0&0\\
0&2&0&1&0&1&0&0&0&1&0&1&0&1&1\\
0&2&0&1&1&0&1&0&0&0&0&1&1&0&0\\
0&2&1&0&0&1&0&1&0&0&1&0&0&1&1\\
0&2&1&0&1&0&0&0&1&0&1&0&1&0&0\\
0&2&1&1&0&0&0&0&0&1&1&1&0&0&0\\
0&2&2&0&0&0&1&0&0&0&2&0&0&0&0
\end{bmatrix}
$$

$$
\text{15 Vertex:}\quad
\begin{bmatrix}
0 & 0 & 0 & 0 & 2 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 \\
0 & 0 & 0 & 1 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 2 & 2 \\
0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 2 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 2 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 2 & 2 \\
0 & 0 & 1 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 2 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 2 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 2 & 2 \\
0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 \\
0 & 0 & 1 & 2 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 2 & 0 & 1 & 1 \\
0 & 0 & 1 & 2 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 2 & 1 & 0 & 0 \\
0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 1 & 0 & 2 & 0 & 0 & 2 & 2 \\
0 & 0 & 2 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 2 & 0 & 1 & 1 & 1 \\
0 & 0 & 2 & 0 & 2 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 2 & 0 & 0 \\
0 & 0 & 2 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 2 & 1 & 0 & 1 & 1 \\
0 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 2 & 1 & 1 & 0 & 0 \\
0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 \\
0 & 1 & 0 & 0 & 2 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 2 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 2 & 2 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 \\
0 & 1 & 0 & 2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 1 & 1 \\
0 & 1 & 0 & 2 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 2 & 2 \\
0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 1 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 2 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 \\
0 & 1 & 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 1 & 1 \\
0 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 0 \\
0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 \\
0 & 2 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 \\
0 & 2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 2 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 2 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 2 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

$$\text{20 Vertex: }\begin{bmatrix}
0&0&0&0&2&2&1&0&0&0&0&0&2&2&2&2&2&1&0&0\\
0&0&0&1&1&2&0&1&0&0&0&1&1&2&2&1&2&0&1&0\\
0&0&0&1&2&1&0&0&1&0&0&1&2&1&1&2&1&0&0&1\\
0&0&1&0&1&2&0&0&0&1&1&0&1&2&2&1&2&0&0&0\\
0&0&1&0&2&1&1&0&0&0&1&0&2&1&1&2&1&1&0&0\\
0&0&1&1&0&2&0&1&0&0&1&1&0&2&2&0&2&0&1&0\\
0&0&1&1&1&1&0&0&1&0&1&1&1&1&1&1&1&0&0&1\\
0&0&1&1&2&0&0&0&0&1&1&1&2&0&0&2&0&0&0&0\\
0&0&1&2&0&1&1&0&0&0&1&2&0&1&1&0&1&1&0&0\\
0&0&1&2&1&0&0&1&0&0&1&2&1&0&0&1&0&0&1&0\\
0&0&2&0&0&2&0&0&1&0&2&0&0&2&2&0&2&0&0&1\\
0&0&2&0&1&1&0&0&0&1&2&0&1&1&1&1&1&0&0&0\\
0&0&2&0&2&0&1&0&0&0&2&0&2&0&0&2&0&1&0&0\\
0&0&2&1&0&1&0&1&0&0&2&1&0&1&1&0&1&0&1&0\\
0&0&2&1&1&0&0&0&1&0&2&1&1&0&0&1&0&0&0&1\\
0&0&2&2&0&0&0&0&0&1&2&2&0&0&0&0&0&0&0&0\\
0&1&0&0&1&2&1&0&0&0&0&0&1&2&2&1&2&1&0&0\\
0&1&0&0&2&1&0&1&0&0&0&0&2&1&1&2&1&0&1&0\\
0&1&0&1&0&2&0&0&1&0&0&1&0&2&2&0&2&0&0&1\\
0&1&0&1&1&1&0&0&0&1&0&1&1&1&1&1&1&0&0&0\\
0&1&0&1&2&0&1&0&0&0&0&1&2&0&0&2&0&1&0&0\\
0&1&0&2&0&1&0&1&0&0&0&2&0&1&1&0&1&0&1&0\\
0&1&0&2&1&0&0&0&1&0&0&2&1&0&0&1&0&0&0&1\\
0&1&1&0&0&2&0&0&0&1&1&0&0&2&2&0&2&0&0&0\\
0&1&1&0&1&1&1&0&0&0&1&0&1&1&1&1&1&1&0&0\\
0&1&1&0&2&0&0&1&0&0&1&0&2&0&0&2&0&0&1&0\\
0&1&1&1&0&1&0&0&1&0&1&1&0&1&1&0&1&0&0&1\\
0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&1&0&0&0&0\\
0&1&1&2&0&0&1&0&0&0&1&2&0&0&0&0&0&1&0&0\\
0&1&2&0&0&1&0&1&0&0&2&0&0&1&1&0&1&0&1&0\\
0&1&2&0&1&0&0&0&1&0&2&0&1&0&0&1&0&0&0&1\\
0&1&2&1&0&0&0&0&0&1&2&1&0&0&0&0&0&0&0&0\\
0&2&0&0&0&2&1&0&0&0&0&0&0&2&2&0&2&1&0&0\\
0&2&0&0&1&1&0&1&0&0&0&0&1&1&1&1&1&0&1&0\\
0&2&0&0&2&0&0&0&1&0&0&0&2&0&0&2&0&0&0&1\\
0&2&0&1&0&1&0&0&0&1&0&1&0&1&1&0&1&0&0&0\\
0&2&0&1&1&0&1&0&0&0&0&1&1&0&0&1&0&1&0&0\\
0&2&1&0&0&1&0&1&0&0&1&0&0&1&1&0&1&0&1&0\\
0&2&1&0&1&0&0&0&1&0&1&0&1&0&0&1&0&0&0&1\\
0&2&1&1&0&0&0&0&0&1&1&1&0&0&0&0&0&0&0&0\\
0&2&2&0&0&0&1&0&0&0&2&0&0&0&0&0&0&1&0&0
\end{bmatrix}$$

$$\text{30 Vertex:}\quad \begin{bmatrix}
0 & 0 & 0 & 0 & 2 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 1 & 0 & 2 & 2 & 1 & 0 & 0 & 0 & 2 & 1 & 0 & 2 & 2 \\
0 & 0 & 0 & 1 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 2 & 2 & 1 & 2 & 0 & 1 & 1 & 2 & 0 & 1 & 0 & 0 & 2 & 0 & 1 & 1 & 2 \\
0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 2 & 1 & 1 & 2 & 1 & 0 & 0 & 2 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 2 & 1 \\
0 & 0 & 1 & 0 & 1 & 2 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 2 & 2 & 1 & 2 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 2 \\
0 & 0 & 1 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 2 & 1 & 1 & 2 & 1 & 1 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 2 & 1 \\
0 & 0 & 1 & 1 & 0 & 2 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 2 & 2 & 0 & 2 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 0 & 2 & 0 & 1 & 0 & 2 \\
0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 2 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 2 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 2 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 1 & 0 & 2 & 0 & 0 & 2 & 2 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 1 & 2 & 0 & 0 & 0 & 2 \\
0 & 0 & 2 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 2 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 2 & 0 & 2 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 2 & 0 & 0 & 2 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 0 \\
0 & 0 & 2 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 2 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 2 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 1 & 2 & 1 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & 2 & 1 & 0 & 1 & 2 \\
0 & 1 & 0 & 0 & 2 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 2 & 1 & 1 & 2 & 1 & 0 & 1 & 2 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 2 & 1 \\
0 & 1 & 0 & 1 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 2 & 2 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 1 & 2 & 0 & 0 & 0 & 2 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 2 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 0 \\
0 & 1 & 0 & 2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 2 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 2 & 2 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 \\
0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 1 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 1 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 0 & 2 & 1 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 2 \\
0 & 2 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 2 & 0 \\
0 & 2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 2 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 2 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 2 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{bmatrix}$$

$$
40 \text{ Vertex: } \begin{bmatrix}
0&0&0&0&2&2&1&0&0&0&0&0&2&2&2&2&2&1&0&2&2&1&0&0&0&2&1&0&2&2&2&1&0&0&0&0&0&2&2&2\\
0&0&0&1&1&2&0&1&0&0&0&1&1&2&2&1&2&0&1&1&2&0&1&0&0&2&0&1&1&2&2&0&1&0&0&0&1&1&2&2\\
0&0&0&1&2&1&0&0&1&0&0&1&2&1&1&2&1&0&0&2&1&0&0&1&1&1&0&0&2&1&1&0&0&1&0&0&1&2&1&1\\
0&0&1&0&1&2&0&0&0&1&1&0&1&2&2&1&2&0&0&1&2&0&0&0&0&2&0&0&1&2&2&0&0&0&1&1&0&1&2&2\\
0&0&1&0&2&1&1&0&0&0&1&0&2&1&1&2&1&1&0&2&1&1&0&0&0&1&1&0&2&1&1&1&0&0&0&1&0&2&1&1\\
0&0&1&1&0&2&0&1&0&0&1&1&0&2&2&0&2&0&1&0&2&0&1&0&0&2&0&1&0&2&2&0&1&0&0&1&1&0&2&2\\
0&0&1&1&1&1&0&0&1&0&1&1&1&1&1&1&1&0&0&1&1&0&0&1&1&1&0&0&1&1&1&0&0&1&0&1&1&1&1&1\\
0&0&1&1&2&0&0&0&0&1&1&1&2&0&0&2&0&0&0&2&0&0&0&0&0&0&0&0&2&0&0&0&0&0&1&1&1&2&0&0\\
0&0&1&2&0&1&1&0&0&0&1&2&0&1&1&0&1&1&0&0&1&1&0&0&0&1&1&0&0&1&1&1&0&0&0&1&2&0&1&1\\
0&0&1&2&1&0&0&1&0&0&1&2&1&0&0&1&0&0&1&1&0&0&1&0&0&0&0&1&1&0&0&0&1&0&0&1&2&1&0&0\\
0&0&2&0&0&2&0&0&1&0&2&0&0&2&2&0&2&0&0&0&2&0&0&1&1&2&0&0&0&2&2&0&0&1&0&2&0&0&2&2\\
0&0&2&0&1&1&0&0&0&1&2&0&1&1&1&1&1&0&0&1&1&0&0&0&0&1&0&0&1&1&1&0&0&0&1&2&0&1&1&1\\
0&0&2&0&2&0&1&0&0&0&2&0&2&0&0&2&0&1&0&2&0&1&0&0&0&0&1&0&2&0&0&1&0&0&0&2&0&2&0&0\\
0&0&2&1&0&1&0&1&0&0&2&1&0&1&1&0&1&0&1&0&1&0&1&0&0&1&0&1&0&1&1&0&1&0&0&2&1&0&1&1\\
0&0&2&1&1&0&0&0&1&0&2&1&1&0&0&1&0&0&0&1&0&0&0&1&1&0&0&0&1&0&0&0&0&1&0&2&1&1&0&0\\
0&0&2&2&0&0&0&0&0&1&2&2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&2&2&0&0&0\\
0&1&0&0&1&2&1&0&0&0&0&0&1&2&2&1&2&1&0&1&2&1&0&0&0&2&1&0&1&2&2&1&0&0&0&0&0&1&2&2\\
0&1&0&0&2&1&0&1&0&0&0&0&2&1&1&2&1&0&1&2&1&0&1&0&0&1&0&1&2&1&1&0&1&0&0&0&0&2&1&1\\
0&1&0&1&0&2&0&0&1&0&0&1&0&2&2&0&2&0&0&0&2&0&0&1&1&2&0&0&0&2&2&0&0&1&0&0&1&0&2&2\\
0&1&0&1&1&1&0&0&0&1&0&1&1&1&1&1&1&0&0&1&1&0&0&0&0&1&0&0&1&1&1&0&0&0&1&0&1&1&1&1\\
0&1&0&1&2&0&1&0&0&0&0&1&2&0&0&2&0&1&0&2&0&1&0&0&0&0&1&0&2&0&0&1&0&0&0&0&1&2&0&0\\
0&1&0&2&0&1&0&1&0&0&0&2&0&1&1&0&1&0&1&0&1&0&1&0&0&1&0&1&0&1&1&0&1&0&0&0&2&0&1&1\\
0&1&0&2&1&0&0&0&1&0&0&2&1&0&0&1&0&0&0&1&0&0&0&1&1&0&0&0&1&0&0&0&0&1&0&0&2&1&0&0\\
0&1&1&0&0&2&0&0&0&1&1&0&0&2&2&0&2&0&0&0&2&0&0&0&0&2&0&0&0&2&2&0&0&0&1&1&0&0&2&2\\
0&1&1&0&1&1&1&0&0&0&1&0&1&1&1&1&1&1&0&1&1&1&0&0&0&1&1&0&1&1&1&1&0&0&0&1&0&1&1&1\\
0&1&1&0&2&0&0&1&0&0&1&0&2&0&0&2&0&0&1&2&0&0&1&0&0&0&0&1&2&0&0&0&1&0&0&1&0&2&0&0\\
0&1&1&1&0&1&0&0&1&0&1&1&0&1&1&0&1&0&0&0&1&0&0&1&1&1&0&0&0&1&1&0&0&1&0&1&1&0&1&1\\
0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&1&1&0&0\\
0&1&1&2&0&0&1&0&0&0&1&2&0&0&0&0&0&1&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&1&2&0&0&0\\
0&1&2&0&0&1&0&1&0&0&2&0&0&1&1&0&1&0&1&0&1&0&1&0&0&1&0&1&0&1&1&0&1&0&0&2&0&0&1&1\\
0&1&2&0&1&0&0&0&1&0&2&0&1&0&0&1&0&0&0&1&0&0&0&1&1&0&0&0&1&0&0&0&0&1&0&2&0&1&0&0\\
0&1&2&1&0&0&0&0&0&1&2&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&2&1&0&0&0\\
0&2&0&0&0&2&1&0&0&0&0&0&0&2&2&0&2&1&0&0&2&1&0&0&0&2&1&0&0&2&2&1&0&0&0&0&0&0&2&2\\
0&2&0&0&1&1&0&1&0&0&0&0&1&1&1&1&1&0&1&1&1&0&1&0&0&1&0&1&1&1&1&0&1&0&0&0&0&1&1&1\\
0&2&0&0&2&0&0&0&1&0&0&0&2&0&0&2&0&0&0&2&0&0&0&1&1&0&0&0&2&0&0&0&0&1&0&0&0&2&0&0\\
0&2&0&1&0&1&0&0&0&1&0&1&0&1&1&0&1&0&0&0&1&0&0&0&0&1&0&0&0&1&1&0&0&0&1&0&1&0&1&1\\
0&2&0&1&1&0&1&0&0&0&0&1&1&0&0&1&0&1&0&1&0&1&0&0&0&0&1&0&1&0&0&1&0&0&0&0&1&1&0&0\\
0&2&1&0&0&1&0&1&0&0&1&0&0&1&1&0&1&0&1&0&1&0&1&0&0&1&0&1&0&1&1&0&1&0&0&1&0&0&1&1\\
0&2&1&0&1&0&0&0&1&0&1&0&1&0&0&1&0&0&0&1&0&0&0&1&1&0&0&0&1&0&0&0&0&1&0&1&0&1&0&0\\
0&2&1&1&0&0&0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&0&0&0\\
0&2&2&0&0&0&1&0&0&0&2&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&2&0&0&0&0
\end{bmatrix}
$$