

UNIVERSIDAD DE LA REPÚBLICA
FACULTAD DE INGENIERÍA



FACULTAD DE
INGENIERÍA



UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

ASIGNACIÓN ÓPTIMA DE COSECHAS
DE GRANOS A PLANTAS DE SILOS
UN MODELO DE ASIGNACIÓN DE PROGRAMACIÓN LINEAL
ENTERA.

INFORME DE PROYECTO DE GRADO PRESENTADO POR

MARTÍN PLAZZOTTA

COMO REQUISITO DE GRADUACIÓN DE LA CARRERA DE INGENIERÍA EN
COMPUTACIÓN DE FACULTAD DE INGENIERÍA DE LA UNIVERSIDAD DE LA
REPÚBLICA

SUPERVISORES

MSc. ING. AGUSTÍN LÓPEZ DE LACALLE
DR. ING. FRANCO ROBLEDO

MONTVIDEO, 12 DE DICIEMBRE DE 2022

II

INTEGRANTES DEL TRIBUNAL DE DEFENSA DE TESIS DE GRADO

Prof. MSc. Ing. Omar Viera Zipitria
Dpto. de Investigación Operativa, InCo.

Dr. Elvio Accinelli
Dpto. de Investigación Operativa, InCo.

Dr. Ing. Rafael Sotelo
IIE, FIng.

Agradecimientos

Quiero dedicar este trabajo a mi madre, Ana, por su amor y apoyo constante. También a mi pareja, Rossana, por ser mi gran compañera. Por último, agradecerle a mis tutores, Franco y Agustín, por su tiempo y su guía durante este proyecto.

Resumen

El sector de producción de granos es un área de gran expansión, tanto a nivel nacional como mundial. El aumento en la producción se ve acompañado por el aumento de la demanda y los precios internacionales. En nuestro país, es uno de los sectores fuertes de la economía y encuentra año a año nuevos destinos para la producción nacional.

El conjunto de procesos y actividades que conforman esta cadena productiva ofrecen una oportunidad especial para la aplicación de la ciencia y tecnología, en pos de la mejora de los procesos y la obtención de un producto de alto valor y calidad.

En el presente trabajo se aborda la optimización de actividades fundamentales de la poscosecha. Se trata el problema del retiro, transporte, almacenamiento y distribución de las cosechas de los variados tipos de granos que se producen. Concretamente, en este proyecto se define un problema de optimización cuya solución realiza el retiro de la producción de los agricultores y su traslado hacia las plantas de acopio, aprovechando al máximo la capacidad e infraestructura del conjunto de las plantas y minimizando costos operativos y de transporte.

Se creó un Modelo de Programación Lineal Entera que resuelve dicho problema. Este es un modelo multi-período de asignación logística, distribución y transporte, que considera ventanas de tiempo, flota heterogénea y el manejo de los distintos tipos de granos. Los actores representados en el modelo incluyen a los productores, las plantas de acopio y eventuales consumidores.

El modelo se implementó y testeó en la plataforma AMPL/CPLEX. Se definió también un conjunto de instancias benchmark para estudiar la performance del modelo y su robustez ante instancias de gran tamaño. Se lograron resolver instancias de tamaño considerable, lo cual da un indicio de que el modelo puede ser de valor práctico para su aplicación a casos reales. El modelo creado no pretende ser una versión final y definitiva. En ese sentido, se proponen extensiones del mismo que puedan modelar otros aspectos de la realidad, como la formación de colas en las plantas.

Palabras clave: Poscosecha de granos, cadena de suministros de granos, opti-

VI

mización, silos

Índice general

1. Introducción	1
1.1. Definición del problema	1
1.2. Estado del arte y revisión bibliográfica	6
1.3. Descripción del documento	10
2. Primera aproximación al problema	13
2.1. Simplificación del problema	14
2.2. Modelado del Problema	15
2.2.1. Otros parámetros: tiempos y costos de los viajes	15
2.2.2. Primeras variables y función objetivo	16
2.3. Discretización de la dimensión tiempo	17
2.4. Actividad en las plantas	18
2.5. Modelo 1 completo	22
3. Incorporando los tipos de granos al modelo	23
3.1. Tipos de granos	23
3.2. Silos	24
3.3. Asignación de envíos a silos según sus tipos de granos	25
3.4. Modelo 2 completo	30
4. Modelo para flota heterogénea	33
4.1. Parámetros del problema	33
4.2. Primeras variables y función objetivo	34
4.3. Acopio en las plantas	35
4.4. Tipos de granos	36
4.5. Modelo 3 completo	38
4.5.1. Índices	38
4.5.2. Parámetros	38
4.5.3. Variables	39
4.5.4. Función objetivo y restricciones	39
4.5.5. Modelo 3b	41

5. Implementación del modelo y primeras pruebas	43
5.1. Descripción del entorno de ejecución	43
5.2. Descripción del directorio del proyecto	43
5.3. Datos provistos por el usuario	46
5.4. Datos preprocesados para AMPL	48
5.4.1. Argumentos para el preprocesamiento	48
5.4.2. IJKC.csv	49
5.4.3. consumidoresPRE.csv y productoresPRE.csv	49
5.4.4. tiempos1 y tiempos2	50
5.4.5. costos1_*.csv y costos2_*.csv	50
5.5. Modelo	51
5.5.1. Lectura de los parámetros desde los archivos	51
5.5.2. Código del modelo	53
5.5.3. Testeo unitario del modelo	55
6. Test de Performance	59
6.1. Observaciones teóricas sobre el tamaño del problema	59
6.2. Descripción general de los test	61
6.3. Test de performance 1: aumentando la cantidad de envíos I	62
6.3.1. Descripción de los datos	62
6.3.2. Casos del test	63
6.3.3. Resultados	64
6.4. Test de performance 2: aumentando el tamaño de los envíos p_i	66
6.4.1. Descripción de los datos	66
6.4.2. Casos del test	66
6.5. Test 3: Aumentando el tamaño de los envíos con flota homogénea. . . .	67
6.5.1. Resultados	68
6.6. Análisis de robustez bajo instancias más grandes	71
6.6.1. Test de performance 4: aumentando al límite el número de envíos	72
6.6.2. Resultados	72
6.6.3. Test de performance 5: aumentando el tamaño de los envíos p_i	72
7. Conclusiones	77
7.1. Trabajo realizado	77
7.2. Propuestas para trabajos a futuro	78
7.2.1. A modo de cierre	79
Referencias	81
Anexo: Tasa de llegada de camiones y formación de colas en las plantas	83
.1. Tiempo de descarga en planta y tasa de llegada de vehículos	84
.2. Caso fácil: $dt \geq \mu_\pi$	84
.3. Caso difícil: $dt < \mu_\pi$	85

<i>ÍNDICE GENERAL</i>	IX
.4. Largo de cola y tiempo de servicio	86
.4.1. Modelando el comportamiento de U^t	87
.4.2. Modelando el comportamiento de L^t	88
.5. Bosquejo de un modelo con colas en los silos	89
.6. Restricciones para un modelo con colas en las plantas	90
Anexo: código AMPL del modelo	91
Anexo: Complejidad del Problema	99
Anexo: Tarifa real	101

Capítulo 1

Introducción

1.1. Definición del problema

Las actividades y procesos que se realizan tras la cosecha de granos son de gran importancia a fin de asegurar la correcta conservación del mismo y evitar pérdidas cualitativas o cuantitativas. A su vez, un grano de mayor calidad tendrá mayor valor comercial.

El presente trabajo trata el problema de la asignación de las producciones de granos por parte de agricultores a las plantas de acopio, en el período de poscosecha. Se busca resolver de forma óptima la asignación de producciones a silos minimizando los costos de transporte y aprovechando al máximo la capacidad operativa y de almacenamiento de las plantas en su conjunto. Para ello hemos construido un Modelo de Programación Lineal Entera cuya solución indica qué viajes de camión se deben realizar entre productores y plantas. Los productores realizan producciones de distintos tipos de granos en épocas específicas del año. Dicho grano debe ser trasladado a planta para su acondicionamiento y acopio.

El retiro del grano desde el sitio del productor debe hacerse dentro de períodos muy específicos y cortos (usualmente de unos días a partir de la cosecha). Esto se debe a que los productores no suelen contar con las condiciones adecuadas para el almacenamiento de los granos. Por tanto, estos envíos de grano que los productores realizan consisten de cierta cantidad y cierto tipo de grano el cual debe retirarse dentro de cierta ventana de tiempo.

Cada planta cuenta con cierta cantidad de silos, cada uno con una capacidad dada. Claramente, distintos tipos de grano deben acopiarse en silos separados.

A su vez existen consumidores de granos. Estos consumidores pueden representar, por ejemplo, pedidos de exportación o requerimientos de fábricas. Al igual que con el retiro del grano desde los productores, la entrega a los con-

sumidores debe satisfacerse dentro de ciertos períodos de tiempo conocidos de antemano.

De este modo, la operativa en las plantas, que debe atender la llegada y salida de los distintos tipos de grano, debe realizarse dentro de un cronograma, más o menos ajustado, que cumpla con los requisitos de tiempo acordados y aprovechando al máximo la capacidad de las plantas.

El objetivo fundamental es definir un modelo de optimización cuya solución indique a qué planta (o plantas) se destinará cada envío, y qué planta proveerá a cada consumidor, buscando minimizar el costo total de transporte. El modelo deberá modelar distintos tipos de restricciones:

- Restricciones temporales: para cada envío de granos, el retiro desde el sitio del productor debe realizarse dentro de cierta ventana de tiempo. Análogamente, la entrega al consumidor de un pedido de granos debe realizarse dentro de cierta ventana de tiempo. No consideraremos ventanas de tiempo múltiples.
- Capacidad de las plantas: cada planta cuenta con varios silos, cada uno con cierta capacidad.
- Tipos de granos: los granos no pueden mezclarse; en un silo solo se puede acopiar un tipo de granos a la vez. Eventualmente un silo puede vaciarse antes de pasar a almacenar otro tipo de grano.

A modo de ejemplo puramente ilustrativo, en la Figura 1.1 presentamos una instancia de pequeñas dimensiones. En la misma están presentes dos tipos de granos:

- Trigo, cuya cosecha es en el mes de noviembre.
- Soja, cuya primera cosecha es en marzo y su segunda cosecha es en mayo.

A la izquierda podemos distinguir tres productores, representados por tractores. El primero de ellos tiene un envío de granos que realizar. Este consiste de quince toneladas de trigo, y debe recogerse entre el 15 y el 31 de noviembre.

Siguiendo con el ejemplo, el segundo productor tiene dos envíos, ambos del mismo tipo de grano: soja. El primer envío es de quince toneladas a recoger entre el 15 y el 31 de marzo; el segundo es de veinte toneladas a recoger entre el 15 y el 31 de mayo.

Por último, el tercer productor tiene un envío de quince toneladas de trigo a recoger en la segunda quincena de noviembre, y otro de quince toneladas de soja a recoger en la segunda quincena de marzo.

A los efectos de nuestro problema, los productores son solo relevantes en lo que respecta a su ubicación geográfica y su distancia hasta las plantas. Es decir, envíos pertenecientes a un mismo productor se retiran en el mismo sitio, y por tanto los eventuales viajes de transporte a las plantas tendrán las mismas

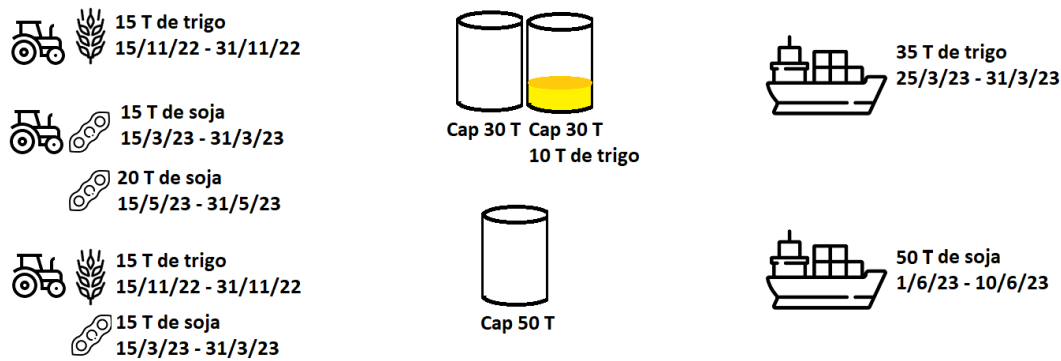


Figura 1.1: Ejemplo de pequeñas dimensiones

distancias y los mismos costos.

En el lado derecho de la figura tenemos representados a los consumidores. Estos tienen pedidos de granos que deben ser satisfechos dentro de cierta ventana de tiempo. El primero de los consumidores pide treinta y cinco toneladas de trigo a entregar entre el 25 y el 31 de marzo. El segundo consumidor pide cincuenta toneladas de soja a entregar en la ventana de tiempo 1 al 10 de junio. Podemos pensar que los consumidores representan ventas de exportación o requerimientos de materia prima por parte de fábricas.

Por último, entre productores y consumidores tenemos a las plantas de silos. Cada silo cuenta con cierta capacidad de almacenamiento. Eventualmente, al inicio algunos de ellos pueden contener ya cierta cantidad de granos almacenada. Si bien los silos se agrupan en plantas, al igual que en el caso de los productores y los envíos de grano, qué silo pertenece a qué planta solo es relevante en lo que respecta a la ubicación del silo, es decir, en lo que respecta a sus distancias a los productores y consumidores.

El objetivo del problema es minimizar el costo de transportar el grano desde los productores a las plantas y de las plantas a los consumidores. Respecto a esto, son conocidas las distancias, y por tanto los costos de transporte, de los productores a las plantas y de estas a los consumidores.

Para el transporte de las cargas se cuenta con una flota de camiones, eventualmente heterogénea. En nuestro problema, los tipos de camión se refieren a camiones de distinta capacidad. Otras características distintivas de cada tipo de camión, como por ejemplo el tiempo de descarga, pueden ser incorporadas al modelo presentado en este trabajo.

Los viajes de camión se realizan mediante la modalidad Full Truck Load (FTL). Esto significa que el camión viaja lleno y que su carga pertenece a un

único productor. Y aunque permitamos que el camión no viaje lleno, el costo del viaje es calculado en función de la capacidad del camión y no de la carga real del viaje.

Como el viaje de camión se dedica a la carga de un único productor, el mismo viajará desde el productor directo a la planta, sin hacer escala en otros productores. También por este motivo, se sabe con certeza que, tras partir del sitio del productor, el camión llegará a planta en un tiempo conocido. El modelo que presentamos permite considerar, o no, este tiempo de viaje y otros tiempos, como el tiempo de carga en el productor o el tiempo requerido por otras actividades realizadas en planta desde que el camión llega a planta y hasta que su carga se ingresa al silo asignado¹. Llamaremos *tiempo de transporte* a la suma de estos tres tiempos.

Considerar el tiempo de transporte tiene sentido si se desea una planificación más granular en el tiempo. Por ejemplo, para un caso podría ser de interés una *planificación* con una granularidad de una hora; esto implica conocer **cada hora**:

- qué viajes de camión deben partir de productores o plantass
- qué viajes de camión ya han arribado a planta y han almacenado su carga
- qué viajes están partiendo de una planta a proveer a un consumidor
- qué viajes ya han entregado su carga al consumidor

En este caso, el modelo deberá considerar el tiempo de transporte de los viajes de camión. Este es el tiempo transcurrido entre que un camión inicia la carga en el productor y que su carga se computa como almacenada en el silo. Ya que los tiempos de transporte son del orden de algunas horas y dado que se busca una *planificación horaria*, el modelo debe conocer y considerar los tiempos de transporte entre cada productor y cada planta.

Sin embargo, podría no ser necesaria una planificación tan fina en el tiempo, y ser suficiente con que el modelo devuelva una *planificación diaria*. En este caso, los parámetros tiempo de transporte se pasan al modelo con valor nulo. Así, la carga de un viaje se computará como almacenada el mismo día en que partió del productor.

Volveremos sobre este tema en otros capítulos. Pero el lector debe saber desde ya que el modelo implementado en este proyecto permite, en su uso y al momento de resolver una instancia real del problema, elegir un nivel de granularidad en el tiempo (horas, días, ..). Dada la granularidad elegida, obtendrá

¹Estas actividades realizadas en planta tras la llegada de un camión incluyen descarga, recepción, almacenamiento en el silo, etc. Por simplicidad, en nuestro problema consideramos que el tiempo de estas actividades en planta es constante y que no depende de la cantidad de camiones que llegan a planta en un momento dado. En el anexo, se presentan un conjunto de restricciones que modelan la tasa de llegada de camiones a planta y la eventual formación de colas en las plantas.

como solución óptima una planificación de la granularidad elegida (planificación horaria, planificación diaria, ...).

En cualquier caso, debemos notar que el modelo creado considera la dimensión tiempo, en lo que respecta a varios aspectos: las ventanas de tiempo de los productores y consumidores, los momentos de llegada y salida de camiones, qué acopio tiene un silo en determinado momento, etc.

Por último, hagamos algunas consideraciones respecto al costo de los viajes. Se espera que la solución óptima hallada por el modelo use la menor cantidad de camiones posible. Por lo tanto, se debe exigir o bien que los camiones viajen llenos, o bien que la tarifa de transporte se aplique sobre la capacidad total del camión y no sobre la carga real del viaje. Además, en caso de contar con camiones de distinta capacidad, los costos de viaje deben favorecer usar menos camiones de más capacidad.

Supongamos una tarifa como la que se muestra en el Cuadro 1.1. Podemos considerar la tarifa como una función en dos variables, distancia y peso. Pero surge una interrogante: ¿la tarifa se debe aplicar a la carga transportada por el viaje o a la capacidad del camión?

Distancia (km)	Tarifa (U\$S)
0 - 50	11,95 por tonelada
51 - 100	17,33 por tonelada
101 - 200	25,95 por tonelada
≥ 201	0,078 por tonelada por km

Cuadro 1.1: Tarifa por distancia y cantidad de toneladas

Como ejemplo consideremos la instancia representada en la Figura 1.2. La misma consiste en un único envío de veinte toneladas y un único silo a cincuenta kilómetros de distancia. Supongamos que la flota cuenta con camiones de diez y de veinte toneladas. Supongamos que se permite que los camiones no viajen llenos, ya sea porque no es un requisito de la realidad o porque en el modelo no hay ninguna restricción que lo impida.

Si la tarifa anterior se aplicara a la carga transportada, en vez de a la capacidad de los camiones, entonces las tres soluciones indicadas en la figura serían óptimas e igual de buenas, pues todas tendrían un costo de U\$S 239.00 :

Solución A) cuatro camiones cargados con cinco toneladas

Solución B) dos camiones de capacidad diez toneladas llenos

Solución C) un camión de capacidad veinte toneladas lleno.

Si en cambio la tarifa se aplicara a la capacidad del camión, entonces la solución A ya no será óptima. Pero dado que la tarifa aún sigue siendo lineal

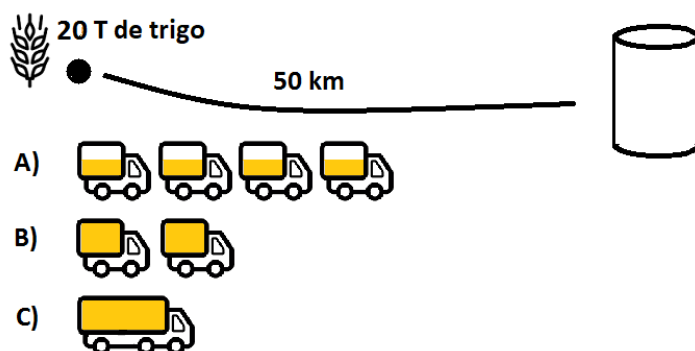


Figura 1.2: Si la tarifa se aplica a la carga y es lineal respecto a ella, entonces las soluciones A, B y C son óptimas

respecto a la variable peso, entonces la solución B es igual de costosa que la C. Como queremos que la solución no use viajes de más, precisamos de algún modo favorecer la solución C sobre la solución B.

Una posibilidad es que la tarifa no sea lineal respecto a la capacidad utilizada, sino que favorezca el uso de camiones de mayor capacidad. La siguiente tarifa (Cuadro 1.2) cumple con esto, ya que favorece utilizar un camión de veinte toneladas en oposición a usar dos camiones de diez toneladas :

Distancia (km)	Camión de 10 T (U\$S)	Camión de 20 T (U\$S)
0 - 50	119,5	215,0
51 - 100	173,3	310,0
101 - 200	259,5	470,0
≥ 201	0,78	1,45 por km

Cuadro 1.2: Tarifas por distancia y capacidad del camión

Bajo esta función de tarifa, las soluciones que usen menos camiones y menos capacidad libre, si se permite que viajen camiones con capacidad libre, serán favorecidas sobre las demás. De aquí en más consideraremos que el costo de los viajes se calcula sobre la capacidad del camión utilizado y que la tarifa favorece el uso de menos camiones de mayor capacidad.

1.2. Estado del arte y revisión bibliográfica

A nivel teórico podemos ver que nuestro problema de asignación de producciones de grano a silos tiene cierta similitud con el Problema General de Asignación (GAP)(Fisher, Jaikumar, y Van Wassenhove, 1986). El GAP puede enunciarse como el problema de asignar I tareas a J personas, de modo que a

cada tarea se le asigna una única persona. Asignar la tarea i a la persona j implica un costo c_{ij} y consumir una cantidad a_{ij} de cierto recurso. La persona j cuenta con una cantidad b_j de dicho recurso. El objetivo es minimizar el costo total de asignar todas las tareas, estando cada persona limitada por su cantidad de recurso b_j . Un modelo para el GAP es el siguiente:

$$\begin{aligned}
& \min_X \sum_{j \in J} \sum_{i \in I} c_{ij} x_{ij} \\
& \text{s.a.} \\
& \sum_{j \in J} x_{ij} = 1, \forall i \in I \\
& \sum_{i \in I} a_{ij} x_{ij} \leq b_j, \forall j \in J \\
& x_{ij} \in \{0, 1\}, \forall (i, j) \in I \times J
\end{aligned}$$

Este modelo es extraído de (Fisher y cols., 1986), donde también se puede encontrar una demostración de que el GAP es NP-duro. La prueba se basa en que el problema NP-Completo de la 2-Partición puede reducirse a él. Utilizando una prueba similar a esta demostraremos que el problema de la 2-Partición también se puede reducir a una instancia especial de nuestro problema de asignación de granos, y por tanto este último es NP-duro. La prueba se encuentra en el anexo y tiene como requisito leer el capítulo dos.

Más material sobre estos problemas se puede encontrar en (Percus, Istrate, y Cristopher, 2006), donde se discute la complejidad de los problemas de partición de números, y en (Kovalyov y Pesch, 2010) donde se provee un enfoque general para probar la pertenencia a NP-duro de este tipo de problemas.

En (Seda, 2022) puede encontrar una revisión del modelo de asignación y como con pequeñas variaciones puede aplicarse a distintos problemas de logística.

En lo que respecta a casos prácticos de optimización, tenemos el caso de (Asgari, Farahani, Rashidi-Bajgan, y Sajadieh, 2013) donde se crea y aplica un modelo de optimización al caso de producción y distribución de trigo entre provincias de India. Cada provincia tiene un consumo mensual fijo y una producción mensual variable. Las provincias con excedente de producción pueden distribuir trigo a provincias con déficit. Cada provincia cuenta con sus propios silos. La harina debe pasar un período de reposo de un mes antes de estar apta para su consumo. Se busca minimizar el costo de transporte y de almacenamiento durante un año calendario. Tiene similitud a nuestro problema en el sentido de que las provincias funcionan como productores, almacenadores y consumidores de grano. Además la producción y distribución se planifica a lo largo del tiempo. Nuestro modelo tiene el agregado de poder incorporar los tiempo de transporte.

La novedad que introduce nuestro problema es la existencia de distintos tipos de granos y como esto limita los posibles destinos de una carga particular (una carga se puede destinar a un silo que está vacío o que contiene el mismo tipo de grano de la carga.)

Otro problema de transporte y almacenamiento se presenta en (Mogale, Kumar, Márquez, y Tiwari, 2017). Similar a nuestro problema, la cadena cuenta con tres eslabones: productores, silos y consumidores. Se consideran dos medios de transporte: camión y tren. Se busca minimizar los costos totales de transporte y almacenamiento. A diferencia de nuestro trabajo, estos dos citados consideran un único tipo de grano: trigo.

Otro artículo relevante es el de (Rong, Akkerman, y Grunow, 2011). Allí se propone un modelo que trata la producción, distribución y almacenamiento de alimentos frescos que modela la degradación de la calidad de los estos en función de la conservación de la temperatura durante el almacenamiento y transporte. Costos de almacenamiento y transporte así como pérdidas por alimentos desechados por disminución en la calidad por debajo de cierto límite son considerados y se busca minimizar la suma de los costos. Si bien este caso trata con alimentos frescos, y no con granos como nuestro caso, el concepto de pérdida de calidad es interesante. Modelar la calidad del grano y controlarla según cierto umbral puede eventualmente ser incorporado a nuestro modelo.

En (Mogale, Kumar, Kumar, y Tiwari, 2018) se modela una cadena de suministros de cuatros eslabones (productores , silos base, silos de campo y consumidores). El modelo allí presentado considera varios aspectos como los tiempos y costos de distintas vías de transporte, y la disponibilidad de vehículos en un momento dado. El objetivo es hallar la ubicación óptima de nuevos silos a construir de entre un conjunto de ubicaciones potenciales. Se intenta minimizar el costo total de la red de suministros. Este costo incluye los costos de construcción de los silos, junto con los costos de transporte y de inventario de la red resultante.

Distintas etapas de la poscosecha de granos ofrecen oportunidades de investigación para procesos de optimización. Recientemente, en (Bayá, Canale, Nesmachnow, Robledo, y Sartor, 2022) se presenta la aplicación al caso real de una de las mayores compañías de acondicionamiento y acopio de granos de Uruguay. En dicho trabajo se optimiza la planificación de líneas de producción de modo de minimizar los costos y cumpliendo con restricciones de fechas de entrega de órdenes de productos, disponibilidad del personal y limpieza de las líneas de producción. El modelo obtenido aplicado a la planificación de la cosecha de soja de 2020, logró disminuir los costos operativos de las líneas de producción en un 10%.

En (Karande y cols., 2019) se estudia y compara la eficiencia de la operativa de varios depósitos de granos de la Corporación Alimentaria de India. Para evaluar la eficiencia de operaciones de cada depósito se aplica una metodología de

Análisis Envolvente de Datos, a partir de la cual se define una medida (DMU) a optimizar.

En nuestro problema, los productores cuentan con ventanas de tiempo en las que se debe retirar la carga. Estas ventanas son de algunos días de duración a lo sumo, debido a que los productores no cuentan con medios de almacenamiento adecuado. En (Lamsal, Jones, y Thomas, 2015) se plantea un problema de optimización en la cual varios granjeros realizan envíos a un molino. Estos envíos se cosechan, retiran y trasladan en el lapso de un día. No existe coordinación entre los productores; y cuando estos empiezan a cosechar, no se detienen hasta haber cosechado toda la producción planeada para el día. Estos productores no cuentan con ninguna forma de almacenamiento, debiendo cada carga que produzcan ser retirada apenas sea cosechada. El molino tiene cierta cantidad objetivo que pretende recibir y procesar; esta cantidad objetivo puede variar en distintos períodos del día (cada una hora por ejemplo). El problema se resuelve en dos etapas: en la primera se determina el momento del día en que cada granjero debe empezar a cosechar (estos momentos de inicio son las variables de decisión). En esta etapa se busca acompasar la tasa de recibo del molino con la tasa de llegada de cargas, buscando minimizar el tiempo en cola de los camiones y el tiempo de ocio del molino. En una segunda etapa, fijados las horas de inicio, se halla la mínima cantidad de camiones necesaria para cumplir con el retiro a tiempo de las cargas.

Vale la pena comparar el modelo de (Lamsal y cols., 2015) con el nuestro en lo que respecta a dos aspectos. Primero, en aquel modelo la planificación de los retiros se realiza en el lapso de un día. El modelo aquí desarrollado ofrece más libertad en su uso, permitiendo realizar planificación de distintos lapsos de tiempo, tan pequeños como un día o tan grandes como un año. Segundo, al realizar una planificación tan intensiva en el tiempo es realista considerar la formación de colas en el destino. El modelo implementado y testeado por nosotros no considera la formación de colas en las plantas, pero se puede extender fácilmente a un modelo que sí lo haga. El diseño teórico que modela la formación de colas en las plantas se provee en la sección de anexos.

En nuestro problema de asignación de producciones de grano a silos nos hemos centrado en las actividades de retiro, transporte y almacenamiento. Otras actividades de poscosecha no han sido consideradas, como la recepción, clasificación, limpieza, secado y acondicionamiento. Estas actividades, junto a aspectos como la humedad y la temperatura de almacenamiento, podrían ser incorporadas en futuros trabajos o investigaciones que se deriven del actual trabajo. En ese sentido, (Jaques y cols., 2022), (Jian y Jayas, 2021) y (Yousaf, Saleh, Ramazan, y Aftab, 2016) son trabajos muy recientes que puede representar una referencia de gran valor en lo que respecta a las actividades como manipulación, secado y almacenamiento, y en vistas de la conservación de la calidad del grano.

Finalmente, y dado que hemos construido un conjunto de instancias de

benchmark para nuestro modelo, corresponde mencionar la biblioteca MIPLIB. Esta es una biblioteca de instancias de benchmark de distintos problemas de optimización. La versión 2010 (*MIPLIB – the Mixed Integer Programming Library*, s.f.) contiene 361 instancias, las cuales se clasifican como fácil (se resuelve en menos de una hora), difícil (se resuelve en más de una hora) o sin resolver. En (Jablonsky, 2015) se compara la performance de tres solvers, entre ellos CPLEX, aplicados a un conjunto de instancias seleccionadas de la MIPLIB2010.

La versión actual de MIPLIB es la sexta (*MIPLIB – 2010*, s.f.), y según consta en su sitio web, el avance tecnológico en solvers y hardware llevó a que seis años después del lanzamiento de la versión 2010, todo el conjunto de instancias se considerara resuelto. Esto dió lugar a la necesidad de la nueva versión.

1.3. Descripción del documento

A continuación se describe la organización general del documento. Los siguientes tres capítulos muestran la evolución de modelo teórico, partiendo de una versión simple del problema hasta llegar al problema general y especificar la versión final del modelo.

En efecto, en el Capítulo 2 se desarrolla un modelo que considera un único tipo de granos y un único tipo de camiones. Allí se definen los elementos fundamentales del modelo: parámetros, índices y primeras variables. Se explica allí como la dimensión continua tiempo se modela mediante un conjunto discreto.

Luego, en el Capítulo 3 se toma en consideración la existencia de distintos tipos de granos y se extiende el modelo acordemente, agregando nuevas restricciones. Distintos tipos de grano deben almacenarse en silos distintos. Este es el aspecto fundamental de nuestro problema, y su correcto modelado es la novedad que este proyecto ofrece.

En el Capítulo 4, a partir de las ideas desarrolladas en los capítulos anteriores, se elabora el modelo final, el cual admite flota heterogénea y tiempos de transporte nulos.

El Capítulo 5 explica la implementación del modelo realizada sobre la plataforma AMPL/CPLEX. También se describen los test unitarios básicos realizados para verificar la implementación hecha.

En el Capítulo 6 se diseñan un conjunto de instancias benchmark, divididas en cinco flujos de test. Se investigó el tiempo de resolución mediante el solver CPLEX. Los primeros tres test buscan explorar como responde el modelo al uso de flota heterogénea y a variaciones en la cantidad de productores o en el tamaño de sus envíos. Los test cuarto y quinto buscan estudiar la robustez del modelo ante instancias de mayor tamaño.

A modo de cierre, en el Capítulo 7 se evalúan los resultados obtenidos y se realizan conclusiones. También se proponen trabajos a futuro, como la posibilidad de realizar extensiones del modelo para considerar la aparición de colas de camiones en las plantas por la llegada intensa de camiones. En el anexo se bosquejan un conjunto de restricciones para modelar dicha extensión.

Capítulo 2

Primera aproximación al problema

En este capítulo desarrollaremos un primer modelo el cual no distingue entre los distintos tipos de granos. Es decir, se tratarán todas las producciones de grano como si fueran del mismo tipo de grano. Además supondremos que la flota es homogénea y consta de un único tipo de camión de capacidad C . Este modelo solo considerará restricciones relativas a la capacidad de los silos y las ventanas de tiempo.

Podemos representar los envíos de granos de los productores como una lista de tuplas¹:

$$Envios = \{ \langle p_i, (a_i, b_i) \rangle \}_{i \in \mathbb{I}}$$

Con esto representamos que el i -ésimo envío es de p_i toneladas y se debe retirar en la ventana determinada por los instantes de tiempo a_i y b_i . Los valores a_i y b_i son enteros².

De modo similar, representamos los pedidos de los consumidores como una lista de K tuplas:

$$Pedidos = \{ \langle q_k, (e_k, l_k) \rangle \}_{k \in \mathbb{K}}$$

Es decir, el k -ésimo pedido es de q_k toneladas y se debe entregar en la ventana de tiempo (e_k, l_k) .

Respecto a los silos, consideramos que existen J silos, donde cada silo j tiene una capacidad de acopio $CapSilo_j$ y un acopio inicial $AcopioIni_j$, eventualmente nulo.

¹Puede contener tuplas repetidas

²Veremos en la Sección 2.3 como se obtienen a partir de las fecha y hora reales

Supondremos que siempre hay camiones disponibles para realizar un viaje. Por tanto, no debemos considerar el tamaño de la flota o si hay disponibilidad de camiones en un instante dado.

2.1. Simplificación del problema

Tal como está planteado el problema, cada envío de un productor y cada pedido de un consumidor puede requerir varios viajes. Si modelamos los envíos como nodos a visitar, el abordaje del problema será más sencillo si de cada nodo se saliera una única vez.

Podemos lograr esto simplificando el problema del siguiente modo. Para cada “nodo envío”, generaremos un “nodo carga” por cada camión que sea necesario para recoger el envío. Es decir, si el envío i es de una carga total p_i , este se representará con $\lceil \frac{p_i}{C} \rceil$ nodos carga.

A modo de ejemplo, consideremos un envío de 18 toneladas a recoger en la ventana de tiempo (5,8). Este envío se desagregará en dos nodos, representando cada uno un viaje de carga. El primero de ellos con capacidad colmada y el segundo con dos toneladas de capacidad libre, como se representa en la Figura 2.1.

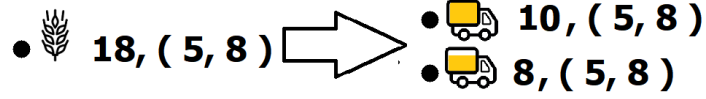


Figura 2.1: Cada nodo que represente un envío lo desagregamos en varios nodos, uno por cada viaje necesario. Cada uno de los nodos resultantes será visitado exactamente una vez.

Procedemos del mismo modo con los pedidos de los consumidores. Obtenemos así un problema más simple y equivalente al original, en donde cada nodo productor o consumidor será visitado exactamente una vez.

Cabe notar que tras esta simplificación el conjunto $\mathbb{I}$ ya no representa el conjunto de envíos, sino que representa viajes de carga, uno por cada camión necesario para transportar el envío completo. Del mismo modo podemos pensar a los elementos del conjunto $\mathbb{K}$ como los viajes de entrega de granos, más que los pedidos enteros.

Hecha la aclaración, eventualmente durante el texto usaremos el término “productor” para referirnos a elementos de $\mathbb{I}$, ya sea que este represente viajes de retiro de granos como en este capítulo, o represente envíos enteros como en el resto de los capítulos. Análogamente, a veces nos referimos a los elementos de $\mathbb{K}$ simplemente como “consumidores”. Finalmente $\mathbb{J}$ representa al conjunto

de los silos.

2.2. Modelado del Problema

2.2.1. Otros parámetros: tiempos y costos de los viajes

Los viajes considerados en nuestro problema son o bien desde productores a plantas, o bien desde plantas a consumidores. Si bien en la realidad pueden haber traslados de grano desde un productor directo a un consumidor, estos casos no son considerados. En nuestro modelo todo grano debe pasar por una planta de acopio en algún momento.

Se presentan a continuación algunos parámetros del problema. Para cada productor i y cada silo j sean:

T_{ij} : tiempo necesario para transportar la carga i al silo j
 c_{ij} : costo de transportar la carga i al silo j

El tiempo de transporte T_{ij} incluirá, además del tiempo de viaje, el tiempo necesario para cargar en el productor i y el tiempo de descarga en j . Del mismo para todo silo j y todo consumidor k están definidos los parámetros:

Γ_{jk} : tiempo de transporte para llevar la carga k desde el silo j
 d_{jk} : costo de transportar la carga k desde el silo j

La Figura 2.2 esquematiza el problema, mostrando los conjuntos de nodos $\mathbb{I}$, $\mathbb{J}$ y $\mathbb{K}$, junto con los parámetros asociados a cada conjunto de nodos. También se representan los parámetros de tiempo de transporte y costo asociados a los conjuntos de aristas $\mathbb{I} \times \mathbb{J}$ y $\mathbb{J} \times \mathbb{K}$.

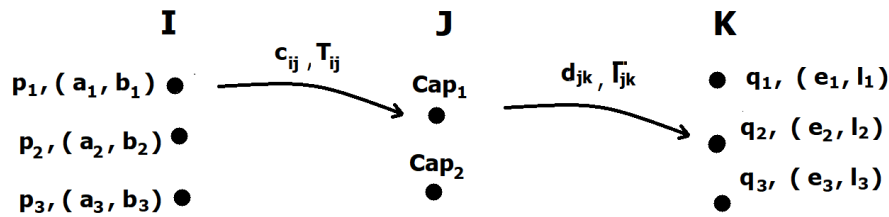


Figura 2.2: Conjuntos de nodos, aristas y sus parámetros asociados. No se dibujan todas las aristas para no sobrecargar el dibujo.

Observemos que se puede separar el problema en dos subproblemas análogos:

- el “lado izquierdo”, consistente en transportar las cargas desde los productores a los silos

- el “lado derecho”, consistente en proveer a los consumidores desde los silos

Como veremos, el tratamiento de ambos lados es similar, lo que nos facilitará el abordaje del problema global. Nos concentraremos en el lado izquierdo del problema, mencionando las analogías correspondientes al lado derecho cuando sea necesario.

2.2.2. Primeras variables y función objetivo

Las variables de decisión que definiremos indicarán a qué silo se destinará cada viaje de retiro desde los productores, y desde qué silo se realizará cada viaje de abastecimiento a un consumidor.

Definimos las siguientes variables binarias:

- Del lado izquierdo

$$x_{ij} = \begin{cases} 1, & \text{si la carga } i \text{ se envía al silo } j. \\ 0, & \text{sino.} \end{cases}$$

- Del lado derecho

$$y_{jk} = \begin{cases} 1, & \text{si desde el silo } j \text{ se provee al consumidor } k. \\ 0, & \text{sino.} \end{cases}$$

Ya estamos en condiciones de definir la función objetivo:

$$f(X, Y) = \sum_{j \in J} \left(\sum_{i \in I} c_{ij} x_{ij} + \sum_{k \in K} d_{jk} y_{jk} \right)$$

Y también las primeras restricciones:

De cada nodo productor i se sale exactamente una vez:

$$\sum_{j \in J} x_{ij} = 1, \quad \forall i \in I \tag{1}$$

$$x_{ij} \in \{0, 1\}, \quad \forall (i, j) \in I \times J \tag{2}$$

Del mismo modo, a cada nodo consumidor k se llega una vez:

$$\sum_{j \in J} y_{jk} = 1, \quad \forall k \in K \tag{3}$$

$$y_{jk} \in \{0, 1\}, \quad \forall (j, k) \in J \times K \tag{4}$$

2.3. Discretización de la dimensión tiempo

En esta sección explicamos como se modela la dimensión continua tiempo, representandola mediante un conjunto finito de marcas de tiempo.

Consideremos los eventos que nos interesan modelar: inicio de un viaje desde un productor³, fin de un viaje con destino a un silo⁴, partida de un viaje desde un silo y llegada de un viaje a un consumidor.

Cualquiera de estos eventos ocurrirá a lo sumo en el instante dado por

$$\max(\{b_i + T_{ij}\}_{(i,j) \in \mathbb{I} \times \mathbb{J}} \cup \{l_k\}_{k \in \mathbb{K}})$$

A su vez, el menor de los instantes en el que puede ocurrir la partida de un viaje desde un productor o desde un silo viene dado por:

$$\min(\{a_i\}_{i \in \mathbb{I}} \cup \{e_k - \Gamma_{jk}\}_{(j,k) \in \mathbb{J} \times \mathbb{K}})$$

Nos referiremos a estos valores como *inicio y fin del problema*, y al lapso de tiempo entre ellos como *duración del problema*.

Para representar la dimensión tiempo, dividimos la duración del problema en un conjunto de intervalos de igual duración, duración a la que llamaremos **dt**. El valor dt funciona como una unidad de tiempo. Definir la duración de los intervalos dt es un tema a considerar con cuidado, ya que define con cuanta granularidad tratamos la planificación de los viajes.

Definimos $\mathbb{T} = \{0, 1, 2, 3, \dots, T\}$ como el conjunto de instantes de tiempo t . El mismo resulta de dividir la duración del problema en intervalos de largo dt , siendo los instantes $t = 0$ y $t = T$, el inicio y fin del problema respectivamente.

Ahora, dada la ventana de tiempo de un productor i , sus fechas desde y hasta se mapean a valores enteros del conjunto $\mathbb{T}$, los cuales se pasarán al modelo como los parámetros a_i y b_i .

Recordemos el ejemplo de la Figura 1.1. Para este caso, el *inicio del problema* es 15/11/22 (a la hora 00:00 para ser específicos), pues es la menor fecha en la que podría ocurrir el inicio de un viaje.

Consideremos el envío 1, su ventana de tiempo es 15/11/22 al 31/11/22. Tenemos que el parámetro $a_1 = 0$ pues la fecha desde de la ventana coincide con el inicio del problema.

Si eligieramos $dt = 1$ día, ¿cuánto vale b_1 ? En ese caso la fecha 31/11/22 se corresponde con $t = 16 \in \mathbb{T}$. Por tanto $b_1 = 16$ y el envío 1 queda representado

³Con inicio de un viaje nos referimos al momento en que comienza la carga del camión en el sitio del productor.

⁴Por fin de un viaje nos referimos al momento en que el camión ha llegado y descargado en el silo. En ese momento el grano ya se considera almacenado en el silo.

por la tupla $\langle 15, (0, 16) \rangle$.

Observación: cuanto menor sea la duración elegida para la unidad dt , mayor resulta el tamaño del conjunto $\mathbb{T}$. Esto influye directamente sobre la cantidad de variables y de restricciones del modelo, ya que varias de ellas están indexadas por dicho conjunto. Esto puede afectar la performance al resolver la instancia del problema obtenida. Volveremos sobre este tema en capítulos posteriores. Por el momento vamos a suponer que se utiliza $dt = 1$ hora, y por tanto tiene sentido considerar los tiempos de transporte en el modelado.

Incorporaremos ahora las variables y restricciones que modelan las ventanas de tiempo. Consideremos el lado izquierdo del problema y modelemos la partida de vehículos desde los productores. Consideremos un envío i , el cual consiste de una carga p_i a retirarse en la ventana de tiempo (a_i, b_i) .

Definimos la variable auxiliar $r_i \in \mathbb{T}$ como el instante en que inicia el flete del envío i ⁵. Claramente se debe cumplir

$$a_i \leq r_i \leq b_i, \forall i \in I \quad (5)$$

$$r_i \in \{0, 1, 2, \dots, T\} \quad (6)$$

De modo similar, definimos la variable s_k como el instante en que se habrá llegado y entregado al consumidor la carga del pedido k . Tenemos:

$$e_k \leq s_k \leq l_k, \forall k \in K \quad (7)$$

$$s_k \in \{1, 2, \dots, T\} \quad (8)$$

Tras definir el conjunto de enteros $\{0, 1, \dots, T\}$ como representación de la dimensión tiempo, debemos exigir que los valores de los tiempos de transporte T_{ij} y Γ_{jk} sean redondeados a cantidades enteras de dt . Por ejemplo, si estamos usando $dt = 1$ hora y el tiempo de transporte real entre un productor i y un silo j es de 2 horas 40 minutos, entonces al modelo matemático recibe el parámetro $T_{ij} = 3$.

Lo que buscamos es que si t es un instante perteneciente a $\mathbb{T}$, entonces $t + T_{ij}$ también pertenece a $\mathbb{T}$. Esto será de gran utilidad como se verá en la próxima sección.

2.4. Actividad en las plantas

Deseamos modelar ahora la actividad de recepción y retiro de granos en las plantas.

⁵Esto es, el instante en que empieza a levantar la carga para su posterior transporte. Recordemos que T_{ij} es la suma de los tiempos de carga en el productor, de viaje en ruta y de descarga en la planta.

Para cada silo j , definamos una función acumulativa A_j^t que representa la cantidad de grano **ingresado** al silo a lo largo del tiempo. Es decir, A_j^t es la cantidad ingresada al silo j hasta el instante t inclusive. Representamos esta función mediante una sucesión de variables $A_j^1, A_j^2, A_j^3, \dots, A_j^T$. Claramente, esta función será creciente en el tiempo, pues aumenta cada vez que ingresa grano al silo:

$$A_j^1 \leq A_j^2 \leq \dots \leq A_j^T$$

Necesitamos ahora hallar una expresión para A_j^t . Definamos una variable binaria ξ_{ij}^t así:

$$\xi_{ij}^t = \begin{cases} 1, & \text{si } x_{ij} = 1 \text{ y en el instante } t \text{ la carga del envío } i \text{ ya se ingresó al silo } j. \\ 0, & \text{sino.} \end{cases}$$

Entonces podríamos expresar A_j^t como:

$$A_j^t = \text{AcopioIni}_j + p_1 \xi_{1j}^t + p_2 \xi_{2j}^t + \dots + p_I \xi_{Ij}^t \quad \forall t \in \{1, 2, \dots, T\}$$

donde AcopioIni_j es la cantidad de acopio inicial en el silo j , la cual es un parámetro del problema.

Tratemos de hallar una expresión para ξ_{ij}^t . Supongamos que el envío i consiste de una carga p_i , la cual el vehículo retira en el instante r_i . Supongamos que la carga se destinará al silo j ($x_{ij} = 1$). Entonces, en el instante $r_i + T_{ij}$ el camión ya habrá arribado y su carga ya estará almacenada en el silo. Por lo tanto tenemos:

$$\xi_{ij}^t = \begin{cases} 1, & \text{si } r_i + T_{ij} \leq t. \\ 0, & \text{si } t < r_i + T_{ij}. \end{cases}$$

El comportamiento buscado para ξ_{ij}^t , se obtiene con las siguientes restricciones (donde M es una constante muy grande)⁶:

$$t - (r_i + T_{ij}) < \xi_{ij}^t M, \quad \forall (t, i, j) \in \{1, 2, \dots, T\} \times \mathbb{I} \times \mathbb{J} \quad (9)$$

$$(r_i + T_{ij}) - t \leq (1 - \xi_{ij}^t) M, \quad \forall (t, i, j) \in \{1, 2, \dots, T\} \times \mathbb{I} \times \mathbb{J} \quad (10)$$

$$\xi_{ij}^t \in \{0, 1\}, \quad \forall (t, i, j) \in \{1, 2, \dots, T\} \times \mathbb{I} \times \mathbb{J} \quad (11)$$

Pero la variable ξ_{ij}^t así definida tiene sentido solo si $x_{ij} = 1$ (como habíamos supuesto). Por tanto, en la expresión para A_j^t debemos corregir los términos multiplicándolos por x_{ij} :

$$A_j^t = A_j^0 + p_1 \xi_{1j}^t x_{1j} + p_2 \xi_{2j}^t x_{2j} + \dots + p_I \xi_{Ij}^t x_{Ij} \quad \forall t \in \{1, 2, \dots, T\}$$

⁶En realidad es suficiente con usar $M = T$.

Esta restricción es de segundo grado en las variables ξ_{ij}^t y x_{ij} . Aplicaremos la siguiente equivalencia, de modo mantener solo restricciones lineales:

Sean x, y, z variables binarias.
Entonces $x \cdot y = z$ equivale al siguiente conjunto de restricciones:

$$\begin{cases} z \leq x \\ z \leq y \\ x + y - 1 \leq z \end{cases}$$

Así agregamos al problema las variables z_{ij}^t y las siguientes restricciones:

$$A_j^t = A_j^0 + \sum_{i \in I} p_i z_{ij}^t \quad \forall t \in \{1, 2, \dots, T\}, \forall j \in \mathbb{J} \quad (12)$$

$$z_{ij}^t \leq \xi_{ij}^t, \quad \forall (t, i, j) \in \{1, 2, \dots, T\} \times \mathbb{I} \times \mathbb{J} \quad (13)$$

$$z_{ij}^t \leq x_{ij}, \quad \forall (t, i, j) \in \{1, 2, \dots, T\} \times \mathbb{I} \times \mathbb{J} \quad (14)$$

$$\xi_{ij}^t + x_{ij} - 1 \leq z_{ij}^t, \quad \forall (t, i, j) \in \{1, 2, \dots, T\} \times \mathbb{I} \times \mathbb{J} \quad (15)$$

$$z_{ij}^t \in \{0, 1\}, \quad \forall (t, i, j) \in \{1, 2, \dots, T\} \times \mathbb{I} \times \mathbb{J} \quad (16)$$

Siguiendo la misma idea que para A_j^t , podemos definir una función de retiro de grano desde los silos, R_j^t :

R_j^t : cantidad de grano retirado del silo j hasta el instante t inclusive

Tenemos que el retiro en el silo j viene dado por :

$$R_j^t = q_1 \theta_{j1}^t y_{j1} + q_2 \theta_{j2}^t y_{j2} + \dots + q_K \theta_{jK}^t y_{jK} \quad \forall t \in \{1, 2, \dots, T\}$$

Donde θ_{jk}^t indica si en el instante t la carga con destino al consumidor k ya ha sido retirada del silo j (suponiendo que $y_{jk} = 1$). Esto es:

$$\theta_{jk}^t = \begin{cases} 1, & \text{si } s_k - \Gamma_{jk} \leq t \\ 0, & \text{si } t < s_k - \Gamma_{jk}. \end{cases}$$

El comportamiento buscado para θ_{jk}^t , se obtiene con las siguientes restricciones:

$$t - (s_k - \Gamma_{jk}) < \theta_{jk}^t M, \quad \forall (t, j, k) \in \{1, 2, \dots, T\} \times \mathbb{J} \times \mathbb{K} \quad (17)$$

$$(s_k - \Gamma_{jk}) - t \leq (1 - \theta_{jk}^t) M, \quad \forall (t, j, k) \in \{1, 2, \dots, T\} \times \mathbb{J} \times \mathbb{K} \quad (18)$$

$$\theta_{jk}^t \in \{0, 1\} \quad , \forall (t, j, k) \in \{1, 2, ..T\} \times \mathbb{J} \times \mathbb{K} \quad (19)$$

Finalmente, en la expresión anterior para R_j^t sustituimos el producto de θ_{jk}^t e y_{jk} por una variable binaria u_{jk}^t y agregamos las siguientes restricciones:

$$R_j^t = \sum_{k \in K} q_k u_{jk}^t \quad \forall t \in \{1, 2, ..T\} \quad , \forall j \in \mathbb{J} \quad (20)$$

$$u_{jk}^t \leq \theta_{jk}^t \quad , \forall (t, j, k) \in \{1, 2, ..T\} \times \mathbb{J} \times \mathbb{K} \quad (21)$$

$$u_{jk}^t \leq y_{jk} \quad , \forall (t, j, k) \in \{1, 2, ..T\} \times \mathbb{J} \times \mathbb{K} \quad (22)$$

$$\theta_{jk}^t + y_{jk} - 1 \leq u_{jk}^t \quad , \forall (t, j, k) \in \{1, 2, ..T\} \times \mathbb{J} \times \mathbb{K} \quad (23)$$

$$u_{jk}^t \in \{0, 1\} \quad , \forall (t, j, k) \in \{1, 2, ..T\} \times \mathbb{J} \times \mathbb{K} \quad (24)$$

Finalmente, agregamos la siguiente restricción para controlar que no se sobrepase la capacidad del silo y que no se realicen retiros en caso de no haber acopio suficiente:

$$0 \leq A_j^t - R_j^t \leq CapSilo_j \quad , \forall t \in \{1, 2, ..T\}, \forall j \in \mathbb{J} \quad (25)$$

2.5. Modelo 1 completo

A modo de síntesis, se presenta a continuación el modelo lineal completo hasta aquí desarrollado. Lo llamaremos Modelo 1.

$$\min_{X,Y} \sum_{j \in J} \left(\sum_{i \in I} c_{ij} x_{ij} + \sum_{k \in K} d_{jk} y_{jk} \right)$$

s.a.

$$\sum_{j \in J} x_{ij} = 1, \forall i \in I \quad (1)$$

$$x_{ij} \in \{0, 1\}, \forall (i, j) \in I \times J \quad (2)$$

$$\sum_{j \in J} y_{jk} = 1, \forall k \in K \quad (3)$$

$$y_{jk} \in \{0, 1\}, \forall (j, k) \in J \times K \quad (4)$$

$$a_i \leq r_i \leq b_i, \forall i \in I \quad (5)$$

$$r_i \in \{1, 2, \dots, T\}, \forall i \in I \quad (6)$$

$$e_k \leq s_k \leq l_k, \forall k \in K \quad (7)$$

$$s_k \in \{1, 2, \dots, T\}, \forall k \in K \quad (8)$$

$$t - (r_i + T_{ij}) < \delta_{ij}^t M, \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (9)$$

$$(r_i + T_{ij}) - t \leq (1 - \delta_{ij}^t) M, \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (10)$$

$$\delta_{ij}^t \in \{0, 1\}, \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (11)$$

$$A_j^t = A_j^0 + \sum_{i \in I} p_i z_{ij}^t, \forall t \in \{1, 2, \dots, T\}, \forall j \in J \quad (12)$$

$$z_{ij}^t \leq \delta_{ij}^t, \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (13)$$

$$z_{ij}^t \leq x_{ij}, \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (14)$$

$$\delta_{ij}^t + x_{ij} - 1 \leq z_{ij}^t, \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (15)$$

$$z_{ij}^t \in \{0, 1\}, \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (16)$$

$$t - (s_k - \Gamma_{jk}) < \theta_{jk}^t M, \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (17)$$

$$(s_k - \Gamma_{jk}) - t \leq (1 - \theta_{jk}^t) M, \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (18)$$

$$\theta_{jk}^t \in \{0, 1\}, \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (19)$$

$$R_j^t = \sum_{k \in K} q_k u_{jk}^t, \forall t \in \{1, 2, \dots, T\}, \forall j \in J \quad (20)$$

$$u_{jk}^t \leq \theta_{jk}^t, \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (21)$$

$$u_{jk}^t \leq y_{jk}, \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (22)$$

$$\theta_{jk}^t + y_{jk} - 1 \leq u_{jk}^t, \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (23)$$

$$u_{jk}^t \in \{0, 1\}, \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (24)$$

$$0 \leq A_j^t - R_j^t \leq CapSilo_j, \forall t \in \{1, 2, \dots, T\}, \forall j \in J \quad (25)$$

Capítulo 3

Incorporando los tipos de granos al modelo

3.1. Tipos de granos

Hasta este momento no hemos considerado la existencia de distintos tipos de granos. En este capítulo, agregaremos al Modelo 1 las variables y restricciones necesarias para así considerar los distintos tipos de granos y mantener a los mismos separados al momento de su acopio.

Sea $\{1, 2, \dots, G\}$ el conjunto de los tipos de granos. Ahora, las tuplas que representan los envíos deben contener un nuevo elemento g_i , que indica el tipo de grano del envío:

$$Envios = \{ \langle p_i, g_i, (a_i, b_i) \rangle \}_{i \in \mathbb{I}}$$

Una tupla de este conjunto representa que el envío i consiste de una cantidad p_i del tipo de grano g_i (con $g_i \in \{1, 2, \dots, G\}$), la cual se debe recoger en la ventana de tiempo (a_i, b_i) .

Análogamente, agregamos el parámetro tipo de grano h_k a las tuplas de pedidos de los consumidores:

$$Pedidos = \{ \langle q_k, h_k, (e_k, l_k) \rangle \}_{k \in \mathbb{K}}$$

Respecto a los silos, a los parámetros capacidad y acopio inicial agregamos un parámetro que indique el tipo de granos inicialmente almacenado en el silo. De este modo, los parámetros relativos a los silos vienen dados por la siguiente lista de ternas:

$$infoSilos = \{ \langle AcopioIni_j, GranoIni_j, CapSilo_j \rangle \}_{j \in \mathbb{J}}$$

Una terna de esta lista indica que el silo j es de capacidad $CapSilo_j$ y que inicialmente contiene una cantidad $AcopioIni_j$ del tipo de grano $GranoIni_j$. Si el silo se encuentra vacío al inicio entonces el valor de $GranoIni_j$ es indiferente; lo setearemos a cero en ese caso.

3.2. Silos

Al incorporar los distintos tipos de granos al modelo, debemos considerar una nuevo requerimiento: en un instante de tiempo dado, un silo solo puede contener un único tipo de grano. Es decir, los granos deben mantenerse separados en los silos.

Definamos para cada silo j y cada instante t , una variable $G_j^t \in \{1, \dots, G\}$ que indica qué tipo de grano contiene el silo en ese instante. Al considerar el valor de G_j^t también debemos considerar si el silo está vacío o no en el instante t .

Recordemos que la cantidad de grano en un instante t viene dado por $A_j^t - R_j^t$, y que la restricción 25 del Modelo 1 garantiza que dicha expresión será no negativa.

Ahora definamos una variable binaria δA_j^t , que indique si en el instante t hay grano en el silo j :

$$\delta A_j^t = \begin{cases} 1, & \text{si } A_j^t - R_j^t > 0. \\ 0, & \text{si } A_j^t - R_j^t = 0. \end{cases}$$

Dicha variable se puede modelar con las siguientes restricciones, donde M es un número constante muy grande:

$$\delta A_j^t \in \{0, 1\} \quad , \forall (t, j) \in \{1, 2, \dots, T\} \times \mathbb{J} \quad (26)$$

$$\delta A_j^t \leq (A_j^t - R_j^t) \cdot M \quad , \forall (t, j) \in \{1, 2, \dots, T\} \times \mathbb{J} \quad (27)$$

$$A_j^t - R_j^t \leq \delta A_j^t \cdot M \quad , \forall (t, j) \in \{1, 2, \dots, T\} \times \mathbb{J} \quad (28)$$

En realidad, como $A_j^t - R_j^t$ toma valores enteros únicamente, no es necesario multiplicarle por M en la restricción 27.

Además, alcanza con que M sea una cota superior de $A_j^t - R_j^t$; usaremos la constante $CapSilo_j$. Por tanto, las restricciones finales serán:

$$\delta A_j^t \in \{0, 1\} \quad , \forall (t, j) \in \{1, 2, \dots, T\} \times \mathbb{J} \quad (26)$$

$$\delta A_j^t \leq A_j^t - R_j^t \quad , \forall (t, j) \in \{1, 2, \dots, T\} \times \mathbb{J} \quad (27)$$

$$A_j^t - R_j^t \leq \delta A_j^t \cdot Cap_j \quad , \forall (t, j) \in \{1, 2, \dots, T\} \times \mathbb{J} \quad (28)$$

3.3. ASIGNACIÓN DE ENVÍOS A SILOS SEGÚN SUS TIPOS DE GRANOS²⁵

También agregaremos una restricción que asigne valor a la variable G_j^0 :

$$G_j^0 = \text{GranoIni}_j \quad , \forall j \in \mathbb{J} \quad (29)$$

Habiendo definido las variables δA_j^t y G_j^t , en la siguiente sección vamos a modelar la asignación de los distintos tipos de granos a los silos.

3.3. Asignación de envíos a silos según sus tipos de granos

A continuación enunciaremos cuatro condiciones, los cuales buscan modelar adecuadamente el comportamiento de la variable G_j^t a lo largo del tiempo. Cada condición va acompañada de la restricción matemática que la implementa.

Supongamos que la carga del envío i se asigna al silo j . Esta ingresará al silo en el instante $r_i + T_{ij}$ y su tipo de grano es g_i .

Condición Granos 1: “En el instante previo a la llegada de grano de tipo g_i , o bien el silo está vacío o bien el tipo de grano almacenado coincide con g_i .”

La siguiente restricción modela esto, donde $\wedge$ y $\vee$ simbolizan la conjunción y la disyunción lógicas (AND y OR):

$$x_{ij} = 1 \Rightarrow \delta A_j^{r_i+T_{ij}-1} = 0 \vee \delta A_j^{r_i+T_{ij}-1} = 1 \wedge G_j^{r_i+T_{ij}-1} = g_i \quad (30)$$

Es interesante notar que el estado del silo debe ser chequeado no en el instante que llega el grano, $r_i + T_{ij}$, sino en el instante previo. Veamos el porqué. Sea $t = r_i + T_{ij}$. No podemos asegurar que $\delta A^t = 0$ o que $\delta A^t = 1$. Cualquiera de los casos son factibles. Así lo ilustran los siguientes ejemplos.

Supongamos que el problema se inicia a las doce del mediodía. De este modo tenemos que $t = 1$ coincide con la 1 PM, $t = 2$ coincide con las 2 PM, etc. Supongamos que el camión sale de i a la 1 PM y su tiempo de viaje es de 2 horas 40 minutos. T_{ij} se redondea en 3 horas, y la llegada del camión se computa recién en el instante $t = 4$. Supongamos que el tipo de grano traído es $g_i = 3$ y que hasta su llegada no ha habido otras llegadas o retiros.

En el Cuadro 3.1 representaremos los ingresos y retiros de carga a lo largo del tiempo, junto con los valores de δA_t y G^t . Se dejan vacíos los instantes t en que G^t puede tomar cualquier valor, ya que este es indiferente. Recordar que A^t y R^t representan el acopio y retiro acumulado hasta el instante t inclusive, es decir que sus valores se incrementan con cada nueva llegada o retiro. Suponemos

26CAPÍTULO 3. INCORPORANDO LOS TIPOS DE GRANOS AL MODELO

que los camiones transportan de a 10 toneladas.

t	1	2	3	4	...
A	0	0	0	10	...
R	0	0	0	0	...
δA	0	0	0	1	...
G				3	...

Cuadro 3.1: Caso A

En este Caso A, en el instante de llegada de la carga i ($t = 4$), tenemos que $\delta A^t = 1$.

Consideremos ahora el Caso B, en el que además ocurre un retiro en $t = 4$ (supongamos a las 3:50 PM). Obtenemos la tabla del Cuadro 3.2, en la cual para $t = 4$ tenemos $\delta A^t = 0$ en cambio.

t	1	2	3	4	...
A	0	0	0	10	...
R	0	0	0	10	...
δA	0	0	0	0	...
G				3	...

Cuadro 3.2: Caso B

Hemos dejado G^4 seteado en 3 a pesar de que $\delta A^4 = 0$, para indicar que a las 4 PM en punto el silo quedó vacío, pero que en algún momento durante la última hora (último período) hubo acopio del tipo de grano 3.

En conclusión, no debemos chequear si hay grano en el silo o si este es igual al tipo que va a llegar en el instante t de llegada del grano. Debemos, en cambio, chequear esa condición en el instante de tiempo previo $t - 1$.

Por ejemplo, supongamos que otra carga del tipo de granos 3 pretende arribar a planta en la siguiente hora. Su llegada se computará en $t = 5$. En el caso B, esto será posible pues el silo fue vaciado en el instante anterior $t = 4$ y a las 4 PM en punto el silo se encontraba vacío, esto es $\delta A^4 = 0$.

En el Caso A, también se aceptará la nueva carga entrante en $t = 5$ pues, si bien en $t = 4$ el silo no está vacío, el tipo de grano acopiado coincide con el de la carga a ingresar.

Por tanto, la condición Granos 1 debe cumplirse y la restricción dada por la Ecuación (30) la implementa. La restricción 30 puede escribirse de forma

3.3. ASIGNACIÓN DE ENVÍOS A SILOS SEGÚN SUS TIPOS DE GRANOS²⁷

t	1	2	3	4	5
A	0	0	0	10	20
R	0	0	0	10	10
δA	0	0	0	0	1
G				3	3

Cuadro 3.3: Caso B con segunda llegada de grano

t	1	2	3	4	5
A	0	0	0	10	20
R	0	0	0	0	0
δA	0	0	0	1	1
G				3	3

Cuadro 3.4: Caso A con segunda llegada de grano

más compacta, como se muestra a continuación. Esta última versión es la que incluiremos en el modelo:

$$x_{ij} = 1 \wedge \delta A_j^{r_i+T_{ij}-1} = 1 \Rightarrow G_j^{r_i+T_{ij}-1} = g_i \quad (30)$$

Condición Granos 2: “Al momento de llegada de la carga i , G_j se setea al valor del grano g_i .”

$$x_{ij} = 1 \Rightarrow G_j^{r_i+T_{ij}} = g_i \quad (31)$$

La condición Granos 1 se encarga de chequear que el silo tenga las condiciones previas para recibir la carga entrante i . Sin embargo, es esta segunda condición la que se encarga de setear adecuadamente el valor de G_j^t en el instante en que la carga ingresa al silo. Observar que esta condición también controla que si dos cargas llegan al silo al mismo tiempo entonces ambas traen el mismo tipo de grano.

Condición Granos 3: “Mientras el silo no se vacíe, G_j^t no cambia.”

$$\delta A_j^{t-1} = 1 \Rightarrow G_j^t = G_j^{t-1} \quad , \forall t \in \{1, 2, \dots, T\} \quad (32)$$

Esta condición se encarga de propagar en el tiempo el valor de G_j^t , suponiendo que no ocurran nuevas llegadas de grano. Si hubieran nuevas llegadas de grano, estas provocarían la asignación de G_j^t por la condición Granos 2, pero solo para ese período t puntualmente.

Por ejemplo, retomando el Caso A del Cuadro 3.1, teníamos que en el instante $t = 4$ llega el tipo de grano 3. La condición “Granos 2” provoca que $G^4 = 3$. Supongamos que en las siguientes dos horas no ocurrirán nuevas llegadas y que

28CAPÍTULO 3. INCORPORANDO LOS TIPOS DE GRANOS AL MODELO

el silo no se vaciará. La condición “Granos 3” es necesaria pues fuerza a que el valor de G^t se mantenga en el tiempo, mientras el silo no se vacíe. En el Cuadro 3.5 los valores de G^5 y G^6 se setean gracias a la dicha condición.

t	1	2	3	4	5	6	7
A	0	0	0	10	10	10	..
R	0	0	0	0	0	0	..
δA	0	0	0	1	1	1	..
G				3	3	3	..

Cuadro 3.5: Caso A con propagación en el tiempo del valor de G^t

Supongamos que momentos antes de la hora 7 PM ha habido un retiro, el cual vació el silo. Por la condición “Granos 3”, G^7 también se setea adecuadamente al tipo de grano 3, a pesar de que $\delta A^7 = 0$.

t	1	2	3	4	5	6	7
A	0	0	0	10	10	10	10
R	0	0	0	0	0	0	10
δA	0	0	0	1	1	1	0
G				3	3	3	3

Cuadro 3.6: El valor de G^t se propaga hasta el instante en que el silo se vacía incluído.

Por último, vamos a considerar la salida de granos hacia los consumidores. Consideremos un pedido de un consumidor k , por el cual se solicita el tipo de grano h_k .

Supongamos que el consumidor k va a ser abastecido por el silo j ($y_{jk} = 1$) y el camión llegará en el instante s_k . El camión habrá salido de planta en $t = s_k - \Gamma_{jk}$.

Sabemos que en el momento de salida de planta habrá grano disponible en el silo j y que este será suficiente. Esto se debe a la restricción 25, que controla que la cantidad de grano almacenado en el silo nunca sea negativa.

Falta modelar que el tipo de grano en el silo al momento del retiro sea h_k . A continuación enunciamos y modelamos esta última condición.

Condición Granos 4: “Si una carga llega al consumidor k proveniente del silo j entonces en el instante $s_k - \Gamma_{jk}$ el grano en el silo es de tipo h_k .”

$$y_{jk} = 1 \Rightarrow G_j^{s_k - \Gamma_{jk}} = h_k \quad (33)$$

3.3. ASIGNACIÓN DE ENVÍOS A SILOS SEGÚN SUS TIPOS DE GRANOS 29

Por ejemplo, en el caso ilustrado en el Cuadro 3.6, el retiro en $t = 7$ es posible ya que $G^7 = 3$, más allá de que δA haya pasado a cero en $t = 7$.

En conclusión, las condiciones Granos 1 a Granos 3 modelan correctamente la asignación de envíos a los silos teniendo en cuenta el tipo de grano de los envíos. Estas garantizan que tipos de granos distintos se almacenarán en silos separados. La restricción Granos 4 garantiza que al momento de retirarse grano de un silo con el fin de proveer a un consumidor, dicho grano es del tipo solicitado.

Las restricciones 26 a 33 se agregan al modelo para que el mismo asigne correctamente los distintos tipos de granos a los silos. El diseño de estas restricciones constituye la principal característica del modelo y un resultado novedoso del presente trabajo. Sin embargo, caben un par de observaciones.

Primero, hay que notar que las variables δA_j^t y G_j^t han sido indexadas en función de otras variables. Por ejemplo, la expresión $G_j^{r_i+T_{ij}}$ en la restricción 31, muestra la variable G_j^t indexada por $r_i + T_{ij}$. Afortunadamente, las herramientas que hemos usado para la implementación permiten indexar variables de decisión de esta forma.

Segundo, las restricciones 30 a 33 son de tipo condicional. Estas son fácilmente transformables a restricciones puramente lineales, usando la siguiente propiedad:

Sea x una variable binaria. La restricción condicional: $\text{si } x = 1 \Rightarrow A = B$ es equivalente al siguiente par de restricciones, donde M es un número constante grande: $A \leq B + (1 - x) * M$ $B \leq A + (1 - x) * M$

Nuevamente, la herramienta con la cual implementamos el modelo soporta restricciones condicionales. Preferimos estas por sobre sus formas lineales, pues son más compactas y de más fácil lectura.

3.4. Modelo 2 completo

A modo de síntesis se presenta a continuación el Modelo 2 completo, el cual modela el manejo de los distintos tipos de granos.

$$\begin{aligned} \min_{X,Y} \quad & \sum_{j \in J} \left(\sum_{i \in I} c_{ij} x_{ij} + \sum_{k \in K} d_{jk} y_{jk} \right) \\ \text{s.a.} \quad & \end{aligned}$$

$$\sum_{j \in J} x_{ij} = 1 \quad , \forall i \in I \quad (1)$$

$$x_{ij} \in \{0, 1\} \quad , \forall (i, j) \in I \times J \quad (2)$$

$$\sum_{j \in J} y_{jk} = 1 \quad , \forall k \in K \quad (3)$$

$$y_{jk} \in \{0, 1\} \quad , \forall (j, k) \in J \times K \quad (4)$$

$$a_i \leq r_i \leq b_i \quad , \forall i \in I \quad (5)$$

$$r_i \in \{1, 2, \dots, T\} \quad , \forall i \in I \quad (6)$$

$$e_k \leq s_k \leq l_k \quad , \forall k \in K \quad (7)$$

$$s_k \in \{1, 2, \dots, T\} \quad , \forall k \in K \quad (8)$$

$$t - (r_i + T_{ij}) < \delta_{ij}^t M \quad , \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (9)$$

$$(r_i + T_{ij}) - t \leq (1 - \delta_{ij}^t) M \quad , \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (10)$$

$$\delta_{ij}^t \in \{0, 1\} \quad , \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (11)$$

$$A_j^t = A_{\text{copio}} I n_{ij} + \sum_{i \in I} p_i z_{ij}^t \quad \forall t \in \{1, 2, \dots, T\} \quad , \forall j \in J \quad (12)$$

$$z_{ij}^t \leq \delta_{ij}^t \quad , \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (13)$$

$$z_{ij}^t \leq x_{ij} \quad , \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (14)$$

$$\delta_{ij}^t + x_{ij} - 1 \leq z_{ij}^t \quad , \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (15)$$

$$z_{ij}^t \in \{0, 1\} \quad , \forall (t, i, j) \in \{1, 2, \dots, T\} \times I \times J \quad (16)$$

$$t - (s_k - \Gamma_{jk}) < \theta_{jk}^t M \quad , \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (17)$$

$$(s_k - \Gamma_{jk}) - t \leq (1 - \theta_{jk}^t) M \quad , \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (18)$$

$$\theta_{jk}^t \in \{0, 1\} \quad , \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (19)$$

$$R_j^t = \sum_{k \in K} q_k u_{jk}^t \quad \forall t \in \{1, 2, \dots, T\} \quad , \forall j \in J \quad (20)$$

$$u_{jk}^t \leq \theta_{jk}^t \quad , \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (21)$$

$$u_{jk}^t \leq y_{jk} \quad , \forall (t, j, k) \in \{1, 2, \dots, T\} \times J \times K \quad (22)$$

$$\theta_{jk}^t + y_{jk} - 1 \leq u_{jk}^t, \forall (t, j, k) \in \{1, 2, \dots, T\} \times \mathbb{J} \times \mathbb{K} \quad (23)$$

$$u_{jk}^t \in \{0, 1\}, \forall (t, j, k) \in \{1, 2, \dots, T\} \times \mathbb{J} \times \mathbb{K} \quad (24)$$

$$0 \leq A_j^t - R_j^t \leq CapSilo_j, \forall t \in \{1, 2, \dots, T\}, \forall j \in \mathbb{J} \quad (25)$$

$$\delta A_j^t \in \{0, 1\}, \forall (t, j) \in \{1, 2, \dots, T\} \times \mathbb{J} \quad (26)$$

$$\delta A_j^t \leq A_j^t - R_j^t, \forall (t, j) \in \{1, 2, \dots, T\} \times \mathbb{J} \quad (27)$$

$$A_j^t - R_j^t \leq \delta A_j^t \cdot CapSilo_j, \forall (t, j) \in \{1, 2, \dots, T\} \times \mathbb{J} \quad (28)$$

$$G_j^0 = GranoIni_j, \forall j \in \mathbb{J} \quad (29)$$

$$x_{ij} = 1 \wedge \delta A_j^{r_i + T_{ij} - 1} = 1 \Rightarrow G_j^{r_i + T_{ij} - 1} = g_i, \forall (i, j) \in \mathbb{I} \times \mathbb{J} \quad (30)$$

$$x_{ij} = 1 \Rightarrow G_j^{r_i + T_{ij}} = g_i, \forall (i, j) \in \mathbb{I} \times \mathbb{J} \quad (31)$$

$$\delta A_j^{t-1} = 1 \Rightarrow G_j^t = G_j^{t-1}, \forall t \in \{1, 2, \dots, T\}, \forall j \in \mathbb{J} \quad (32)$$

$$y_{jk} = 1 \Rightarrow G_j^{s_k - \Gamma_{jk}} = h_k, \forall (j, k) \in \mathbb{J} \times \mathbb{K} \quad (33)$$

Capítulo 4

Modelo para flota heterogénea

El modelo que presentaremos en este capítulo incorpora el caso de flota heterogénea: distintos tipos de camiones, de distinta capacidad y distinto costo por kilómetro eventualmente. El mismo lo obtendremos a partir del Modelo 2 y realizándole las modificaciones necesarias.

Además, en este modelo se trabaja directamente con la lista de envíos. Ya no habrá necesidad de realizar la simplificación previa para obtener una lista de viajes de carga, la cual se presentó en el Capítulo 2.

De este modo, en este capítulo cada nodo de $\mathbb{I}$ representa un envío. Por tanto, de cada nodo productor partirán uno o varios camiones. A su vez, los elementos del conjunto $\mathbb{K}$ ya no representan descargas, sino que ahora representan los pedidos completos. Por tanto, a cada nodo consumidor llegarán eventualmente varios camiones.

4.1. Parámetros del problema

Sean:

- $\mathbb{I}$: el conjunto de los envíos
- $\mathbb{J}$: el conjunto de silos
- $\mathbb{K}$: el conjunto de pedidos
- $\mathbb{C} = \{1, \dots, C\}$: el conjunto de los tipos de camión, cada uno con distinta capacidad y costo por kilómetro

Ahora el costo de un viaje productor-planta depende, no solo de la distancia entre ellos, sino también del tipo de camión usado. Para cada par (i, j) y para cada tipo de camión c , definimos el parámetro costo χ_{ijc} como el costo del viaje de i a j de un camión de tipo c . Nuevamente, T_{ij} representa el tiempo de transporte desde i a j . Recordamos que por tiempo de transporte nos referimos a la suma del tiempo de carga en i , el tiempo de viaje de i a j y el tiempo de descarga en planta y almacenamiento en el silo j . También para cada par silo j , consumidor k y para cada tipo de camión c consideramos el costo de viaje d_{jkc} y el tiempo de transporte Γ_{jk} .

Nuevamente consideramos la lista de envíos, aunque en este caso no la desagregaremos en una lista de cargas, como habíamos hecho en la Sección 2.1.

$$Envios = \{ \langle p_i, g_i, (a_i, b_i) \rangle \}_{i \in \mathbb{I}}$$

$$Pedidos = \{ \langle q_k, h_k, (e_k, l_k) \rangle \}_{k \in \mathbb{K}}$$

Otra vez consideramos para cada silo la terna acopio inicial, tipo de grano inicial y capacidad:

$$infoSilos = \{ \langle AcopioIni_j, GranoIni_j, CapSilo_j \rangle \}_{j \in \mathbb{J}}$$

Un nuevo parámetro del problema será la capacidad de cada tipo de camión c :

$$\{ CapCamion_c \}_{c \in \mathbb{C}}$$

4.2. Primeras variables y función objetivo

Consideremos el lado izquierdo del problema y definamos la variable entera no negativa x_{ijtc} :

$$\begin{aligned} x_{ijtc} = & \text{cantidad de camiones de tipo } c \\ & \text{que salen de } i \text{ rumbo a } j \\ & \text{en el instante } t \end{aligned}$$

Entonces el costo de mover las cosechas desde los productores a los silos será:

$$\sum_{i \in \mathbb{I}} \sum_{j \in \mathbb{J}} \left(\sum_{c \in \mathbb{C}} \left(\sum_{t \in \mathbb{T}} x_{ijtc} \cdot \chi_{ijc} \right) \right)$$

Aplicando distributiva

$$\sum_{i \in I} \sum_{j \in J} \sum_{c \in C} \sum_{t \in T} x_{ijtc} \cdot \chi_{ijc}$$

Esta última expresión es uno de los grandes términos de la función objetivo.

Consideremos la variable entera x_{ijtc} . Esta es equivalente a la variable binaria x_{ij} del Modelo 2. En aquel modelo, de cada nodo i salía exactamente un camión. Esto se traducía en la siguiente restricción:

$$\sum_{j \in J} x_{ij} = 1, \forall i \in I \quad (1)$$

Como ahora ya no estamos atendiendo el problema simplificado ¹, de los nodos productores parten varios camiones y por tanto debemos usar una variable entera en vez de una binaria.

Como suponemos que los camiones viajan llenos, sabemos que para cada nodo productor i , la capacidad sumada por todos los camiones que salen de él es igual a la carga total del envío i . Suponer que los camiones viajen llenos es algo que se apeg a la realidad del transporte de carga. Además, permitir que haya camiones que no viajen llenos implica varios detalles que habría que considerar (como ya se discutió en la Introducción). Por estos motivos y en favor de mantener la sencillez del modelo, se tomó dicha decisión de diseño: los camiones viajan llenos. Obviamente, esto requerirá que el tamaño de cada envío i se pueda expresar como la suma de capacidades de camiones de la flota.

Usando la nueva variable x_{ijtc} podemos expresar la restricción de las ventanillas de tiempo en los productores así:

$$t < a_i \vee b_i < t \Rightarrow x_{ijtc} = 0$$

4.3. Acopio en las plantas

Consideremos ahora un silo j y un instante t . Definamos la variable α_j^t como la cantidad de grano que ingresó al silo en el período t . ²

Conservaremos la variable A_j^t , que representa el acopio acumulado hasta el instante t inclusive. En el modelo que estamos construyendo tendremos la restricción:

$$A_j^t = \text{AcopioIni}_j + \alpha_j^1 + \alpha_j^2 + \alpha_j^3 + \dots + \alpha_j^t$$

¹Recordar la Sección 2.1

²Es decir, en el intervalo semiabierto $(t-1, t]$

Necesitamos hallar una expresión para α_j^t . Definamos la variable ρ_{ijt} como la cantidad de grano que sale de i rumbo a j en el instante t . Como los camiones viajan llenos, tenemos que ρ_{ijt} es la capacidad sumada de todos los camiones que salen de i a j en el instante t :

$$\rho_{ijt} = \sum_{c \in \mathbb{C}} (x_{ijtc} \cdot \text{CapCamion}_c)$$

Además tenemos que para todo envío i (cuyo tamaño es p_i) se cumple :

$$\sum_{j \in \mathbb{J}} \sum_{t \in \mathbb{T}} \rho_{ijt} = p_i$$

Ahora podemos calcular α_j^t en función de las variables ρ_{ijt} . Primero debemos observar que el grano que ingresa al silo j en el instante t proveniente de i habrá partido de i en el instante $t - T_{ij}$. Por tanto, tenemos que:

$$\alpha_j^t = \sum_{i \in \mathbb{I}, t \geq T_{ij}} \rho_{i,j,t-T_{ij}}$$

Como es costumbre, al final del capítulo se incluye el modelo completo, incluyendo sus variables y restricciones.

4.4. Tipos de granos

Respecto a los tipos de granos y en comparación con el Modelo 2:

- Se mantiene la variable binaria δA_j^t y las restricciones 26 a 28 del Modelo 2 que modelan su comportamiento.
- En las restricciones 30 a 33 del Modelo 2, alcanza con sustituir la expresión $x_{ij} = 1$ por $x_{ijtc} > 0$.

Por ejemplo, la restricción 30 del Modelo 2 :

$$x_{ij} = 1 \wedge \delta A_j^{r_i+T_{ij}-1} = 1 \Rightarrow G_j^{r_i+T_{ij}-1} = g_i, \quad \forall (i, j) \in \mathbb{I} \times \mathbb{J}$$

se transforma en:

$$x_{ijtc} > 0 \wedge \delta A_j^{t+T_{ij}-1} = 1 \Rightarrow G_j^{t+T_{ij}-1} = g_i, \quad \forall (i, j, t, c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C}$$

O mejor aún, podemos sustituir $x_{ijtc} > 0$ por $\rho_{ijt} > 0$, con lo cual evitamos el índice c y la cantidad de restricciones resultantes será menor. Con esto, la restricción quedaría formulada como:

$$\rho_{ijt} > 0 \wedge \delta A_j^{t+T_{ij}-1} = 1 \Rightarrow G_j^{t+T_{ij}-1} = g_i, \quad \forall (i, j, t) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T}$$

Hagamos una observación respecto a esta última expresión. Debemos tener la precaución de que el valor $t + T_{ij} - 1$ sea un índice válido, es decir, mayor o igual a cero. Esto no ocurrirá para $t = 0$ y si al tiempo de transporte T_{ij} se le asigna valor nulo.

Como ya se explicó en otros capítulos, tiene sentido asignar valor cero a los tiempos de transporte T_{ij} en caso de seleccionar un valor de dt mucho mayor a una hora; por ejemplo, dt igual a una semana. Como vimos, los tiempos reales de transporte deben ser redondeados a valores enteros de dt , de modo que se cumpla que si $t \in \mathbb{T}$ entonces $t + T_{ij} \in \mathbb{T}$. Por tanto, ante la elección de un valor de dt muy grande, corresponde setear los parámetros T_{ij} a cero. De lo contrario, los tiempos de transporte, que en la realidad son de algunas horas como máximo, serían redondeados a una semana, es decir, un dt .

Para permitir que los tiempos de transporte sean nulos y evitar índices inválidos, como en el caso que se explicó, la restricción 30 se puede indexar para los valores de t positivos y agregar una restricción para el caso $t = 0$. Obtenemos así las siguientes ecuaciones que sustituyen a la restricción 30 del Modelo 2:

$$\rho_{ijt} > 0 \wedge t + T_{ij} > 0 \wedge \delta A_j^{t+T_{ij}-1} = 1 \Rightarrow G_j^{t+T_{ij}-1} = g_i, \forall (i, j, t) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \quad (30b)$$

$$\rho_{ij0} > 0 \wedge T_{ij} = 0 \wedge \delta A_j^0 = 1 \Rightarrow G_j^0 = g_i, \forall (i, j) \in \mathbb{I} \times \mathbb{J} \quad (30c)$$

A continuación se presenta el Modelo 3 completo. Este funciona con tiempos de transporte T_{ij} no nulos. Se indican también cambios en las restricciones, como el mostrado recién, necesarios para que el modelo acepte parámetros T_{ij} nulos. A este modelo resultante lo nombramos Modelo 3b.

4.5. Modelo 3 completo

A continuación se presenta el Modelo 3, el cual soporta flota heterogénea. Se exige que los camiones viajen con capacidad completa. Además los parámetros T_{ij} , tiempos de transportes entre productores y silos, deben ser positivo.

4.5.1. Índices

- $\mathbb{I} = 1..I$ es el índice sobre la lista de envíos de los productores.
- $\mathbb{J} = 1..J$ es el índice sobre la lista de los silos.
- $\mathbb{K} = 1..K$ es el índice sobre la lista de pedidos de los consumidores.
- $\mathbb{C} = 1..C$ es el índice sobre los tipos de camiones.
- $\mathbb{T} = 0..T_{max}$ es el conjunto de los instantes de tiempo.

4.5.2. Parámetros

- $Envios = \{ \langle p_i, g_i, (a_i, b_i) \rangle \}_{i \in \mathbb{I}}$
Es la lista de los envíos de los productores, indexada sobre el conjunto $\mathbb{I}$. Para la tupla i -ésima: p_i es el tamaño del envío, g_i el tipo de grano y (a_i, b_i) es la ventana de tiempo en la cual se debe retirar.
- $Pedidos = \{ \langle q_k, h_k, (e_k, l_k) \rangle \}_{k \in \mathbb{K}}$
Similar a los envíos, es la lista de los pedidos de los consumidores. Para la tupla k -ésima: q_k es el tamaño del pedido, h_k el tipo de grano solicitado y (e_k, l_k) es la ventana de tiempo en la cual debe entregarse.
- $infoSilos = \{ \langle AcopioIni_j, GranoIni_j, CapSilo_j \rangle \}_{j \in \mathbb{J}}$
Para cada silo j , $CapSilo_j$ es la capacidad del silo, $AcopioIni_j$ y $GranoIni_j$ son la cantidad y tipo de grano presentes en el silo al inicio.
- $CapCamion_c$: capacidad del tipo de camión c , con $c \in \mathbb{C}$.
- T_{ij} : tiempo de transporte del envío i al silo j . Incluye el tiempo de carga en el productor, el tiempo de viaje hasta la planta y el tiempo necesario en planta entre que el viaje llega y la carga se recibe y almacena en el silo.
- Γ_{jk} : tiempo de transporte del pedido k , desde el silo j hasta el consumidor. Incluye los tiempos necesarios para retirar la carga del silo, viajar hasta el sitio del consumidor y descargar en el lugar del consumidor.
- χ_{ijc} : costo del flete de un camión de tipo c desde el productor i a la planta del silo j
- d_{jkc} : costo del flete de un camión de tipo c desde la planta del silo j al consumidor k .

4.5.3. Variables

- x_{ijtc} , $\forall(i, j, t, c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C}$: cantidad de vehículos de tipo c que salen de i en el instante t y cuya carga se asigna al silo j .
- ρ_{ijt} , $\forall(i, j, t) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T}$: cantidad de grano que sale de i en el instante t a depositarse en el silo j .
- y_{jktc} , $\forall(j, k, t, c) \in \mathbb{J} \times \mathbb{K} \times \mathbb{T} \times \mathbb{C}$: cantidad de vehículos de tipo c que salen de j hacia k en el instante t .
- ϕ_{jkt} , $\forall(i, j, t) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T}$: cantidad de carga que se retira del silo j en el instante t con destino al consumidor k .
- α_j^t , $\forall(j, t) \in \mathbb{J} \times \mathbb{T}^+$: cantidad de grano ingresado al silo j en el período t , es decir en el intervalo de tiempo $(t-1, t]$.
- A_j^t , $\forall(j, t) \in \mathbb{J} \times \mathbb{T}$: cantidad de grano ingresado en el silo j hasta el instante t inclusive.
- r_j^t , $\forall(j, t) \in \mathbb{J} \times \mathbb{T}$: cantidad de grano retirado del silo j en el período t , similar a α_j^t .
- R_j^t , $\forall(j, t) \in \mathbb{J} \times \mathbb{T}$: cantidad de grano retirado del silo j hasta el instante t inclusive.
- δA_j^t , $\forall(j, t) \in \mathbb{J} \times \mathbb{T}$: variable binaria que indica si en el instante t el silo j está vacío, aunque haya estado ocupado en algún momento del período $(t-1, t]$.
- G_j^t , $\forall(j, t) \in \mathbb{J} \times \mathbb{T}$: tipo de grano que se guardó en el silo j durante el período $(t-1, t]$, aunque en el instante t el silo haya quedado vacío.

4.5.4. Función objetivo y restricciones

$$\min_{X, Y} \sum_{i \in \mathbb{I}} \sum_{j \in \mathbb{J}} \sum_{c \in \mathbb{C}} \sum_{t \in \mathbb{T}} (\chi_{ijc} \cdot x_{ijtc}) + \sum_{k \in \mathbb{K}} \sum_{j \in \mathbb{J}} \sum_{c \in \mathbb{C}} \sum_{t \in \mathbb{T}} (d_{jkc} \cdot y_{jktc})$$

s.t.

$$x_{ijtc} \in \mathbb{N}, \forall(i, j, t, c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C} \quad (1)$$

$$t < a_i \vee b_i < t \Rightarrow \rho_{ijt} = 0, \forall(i, j, t) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \quad (2)$$

$$\sum_{c \in \mathbb{C}} (x_{ijtc} \cdot \text{CapCamion}_c) = \rho_{ijt}, \forall(i, j, t) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \quad (3)$$

$$\sum_{j \in \mathbb{J}} \sum_{t \in \mathbb{T}} \rho_{ijt} = p_i, \forall i \in \mathbb{I} \quad (4)$$

$$y_{jktc} \in \mathbb{N}, \forall(j, k, t, c) \in \mathbb{J} \times \mathbb{K} \times \mathbb{T} \times \mathbb{C} \quad (5)$$

$$t + \Gamma_{jk} < e_k \vee l_k < t + \Gamma_{jk} \Rightarrow \phi_{jkt} = 0 \quad , \forall (j, k, t) \in \mathbb{J} \times \mathbb{K} \times \mathbb{T} \quad (6)$$

$$\sum_{c \in C} (y_{jktc} \cdot CapCamion_c) = \phi_{jkt} \quad , \forall (j, k, t) \in \mathbb{J} \times \mathbb{K} \times \mathbb{T} \quad (7)$$

$$\sum_{j \in J} \sum_{t \in T} \phi_{jkt} = q_k \quad , \forall k \in \mathbb{K} \quad (8)$$

Funciones de acopio y retiro

$$\alpha_j^t = \sum_{i \in \mathbb{I}, t \geq T_{ij}} \rho_{i,j,t-T_{ij}} \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T}^+ \quad (9)$$

$$A_j^t = AcopioIni_j + \sum_{\tau=1}^{\tau=t} \alpha_j^\tau \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T}^+ \quad (10)$$

$$r_j^t = \sum_{k \in \mathbb{K}} \phi_{jkt} \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T} \quad (11)$$

$$R_j^t = \sum_{\tau=0}^{\tau=t} r_j^\tau \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T} \quad (12)$$

$$0 \leq A_j^t - R_j^t \leq CapSilo_j \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T} \quad (13)$$

Tipos de granos

$$\delta A_j^t \in \{0, 1\} \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T} \quad (14)$$

$$\delta A_j^t \leq (A_j^t - R_j^t) \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T} \quad (15)$$

$$A_j^t - R_j^t \leq \delta A_j^t \cdot CapSilo_j \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T} \quad (16)$$

$$G_j^0 = GranoIni_j \quad , \forall j \in \mathbb{J} \quad (17)$$

$$\rho_{ijt} > 0 \wedge \delta A_j^{t+T_{ij}-1} = 1 \Rightarrow G_j^{t+T_{ij}-1} = g_i \quad , \forall (i, j, t, c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C} \quad (18)$$

$$\rho_{ijt} > 0 \Rightarrow G_j^{t+T_{ij}} = g_i \quad , (i, j, t, c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C} \quad (19)$$

$$\delta A_j^{t-1} = 1 \Rightarrow G_j^t = G_j^{t-1} \quad , \forall t \in \mathbb{T}^+, \forall j \in \mathbb{J} \quad (20)$$

$$\phi_{jkt} > 0 \Rightarrow G_j^t = h_k \quad , \forall (j, k, t) \in \mathbb{J} \times \mathbb{K} \times \mathbb{T} \quad (21)$$

4.5.5. Modelo 3b

Para que el modelo acepte parámetros T_{ij} y Γ_{jk} nulos se deben modificar levemente algunas restricciones.

La restricción 17 se debe sustituir por:

$$\delta A_j^0 = 1 \Rightarrow G_j^0 = GranoIni_j \quad , \forall j \in \mathbb{J} \quad (17)$$

Como ya se observó en la Sección 4.4, la restricción 18 se debe sustituir por el siguiente par de restricciones:

$$\rho_{ijt} > 0 \wedge t + T_{ij} > 0 \wedge \delta A_j^{t+T_{ij}-1} = 1 \Rightarrow G_j^{t+T_{ij}-1} = g_i \quad , \forall (i, j, t) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \quad (18a)$$

$$\rho_{ij0} > 0 \wedge T_{ij} = 0 \wedge \delta A^0 = 1 \Rightarrow G_j^0 = g_i \quad , \forall (i, j) \in \mathbb{I} \times \mathbb{J} \quad (18b)$$

Por último, las restricciones 9 y 10 que computan α_j^t y A_j^t se sustituyen por:

$$\alpha_j^t = \sum_{i \in \mathbb{I}, t \geq T_{ij}} \rho_{i,j,t-T_{ij}} \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T} \quad (9)$$

$$A_j^t = AcopioIni_j + \sum_{\tau=0}^{\tau=t} \alpha_j^\tau \quad , \forall (j, t) \in \mathbb{J} \times \mathbb{T} \quad (10)$$

Capítulo 5

Implementación del modelo y primeras pruebas

En los capítulos anteriores hemos realizado el desarrollo teórico de un modelo de programación lineal entera que resuelve el problema planteado. El Modelo 3 y su versión 3b fueron implementados y testeados usando la plataforma AMPL junto al solver CPLEX.

En esta sección veremos algunos detalles de la implementación y testeo del modelo teórico. También veremos un mecanismo por el cual un usuario puede proveer datos para el modelo en un formato amigable. Luego, estos datos son procesados y pasados al modelo como una instancia.

5.1. Descripción del entorno de ejecución

AMPL es una herramienta que ofrece un lenguaje de modelado para problemas de programación matemática y optimización. Este lenguaje es muy similar a la notación algebraica de los modelos matemáticos. AMPL también integra distintos solvers, como CPLEX, el cual hemos usado para este proyecto.

Como ambiente de ejecución se utilizó el servidor Dantzig de Facultad de Ingeniería. El mismo está dedicado a tareas de optimización. En el Cuadro 5.1 se muestran las especificaciones del entorno de ejecución.

Como entorno de desarrollo local se utilizó el entorno de desarrollo integrado Visual Studio Code, sincronizado con el servidor Dantzig mediante conexión ssh.

5.2. Descripción del directorio del proyecto

Junto con el presente documento se entrega el directorio del proyecto, “Proy_Poscosecha”. Parte del mismo se muestra en la Figura 5.1. Allí se encuentra el código fuente

```

AMPL version 20220119
CPLEX version 20.1.0.0
Servidor :
Arquitectura: x86_64
Sistema Operativo: Linux
Hostname dantzig.fing.edu.uy
Procesador Intel Core i9-9900K 3.60 GHz 16 cores 32 hilos
RAM 64 GiB
L1d caché 32K
L1i caché 32K
L2 caché 256K
L3 caché 16384K

```

Cuadro 5.1: Especificaciones del entorno de ejecución

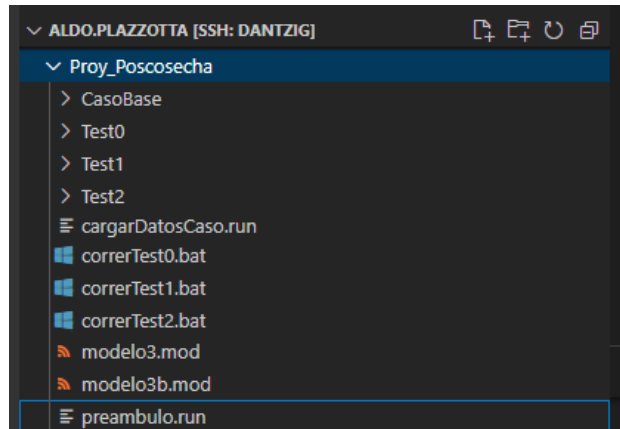


Figura 5.1: Directorio del proyecto ubicado en el servidor AMPL

del modelo implementado. Se han implementado los modelos 3 y 3b. Su código completo se muestra en el anexo.

A modo de verificación y validación del modelo se diseñaron flujos de casos de test. Estos flujos están contenidos en las subcarpetas Test0, Test1 y Test2. Cada flujo consiste de una secuencia de instancias, algunas factibles y algunas no factibles.

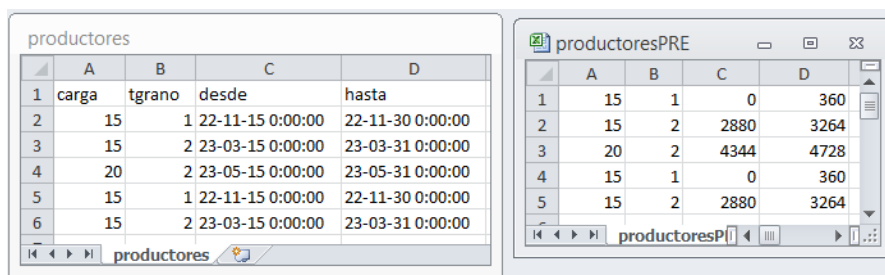
Para cada flujo de test, se provee un script que lo implementa. El script se encarga de cargar los datos de la instancia inicial. Luego, este va modificando los parámetros y se resuelve cada instancia resultante. En cada resolución se chequea que la solución y valor óptimos hallados sean los esperados. A fin de reproducir las pruebas aquí descriptas se debe ubicar el directorio “Proy_Poscosecha”

en el servidor AMPL.

Los scripts `preambulo.run` y `cargarDatosCaso.run` leen los parámetros de una instancia a ejecutar. Dichos parámetros se leen desde archivos de valores separados por comas (CSV). Por ejemplo, la carpeta “CasoBase” contiene los archivos con los parámetros del caso presentado en la introducción (Figura 1.1).

Estos archivos contienen los datos de la instancia como parámetros del problema. Por ejemplo, el archivo “`productoresPre.csv`” contiene los parámetros para el modelo relativos a las producciones de grano. En él, los extremos de las ventanas de tiempo están expresados como enteros, tal como el modelo lo requiere.

Dichos valores se han obtenido a partir de las fechas “reales” de las ventanas de tiempo y expresandolas en base a $dt = 1$ hora, tal como se explicó en la Sección 2.3. La Figura 5.2 muestra el contenido de ambos archivos. El primero de ellos tiene un formato de datos “amigable con el usuario”, mientras que el segundo está destinado a pasar valores de parámetros al modelo AMPL.



	A	B	C	D
1	carga	tgrano	desde	hasta
2	15	1	22-11-15 0:00:00	22-11-30 0:00:00
3	15	2	23-03-15 0:00:00	23-03-31 0:00:00
4	20	2	23-05-15 0:00:00	23-05-31 0:00:00
5	15	1	22-11-15 0:00:00	22-11-30 0:00:00
6	15	2	23-03-15 0:00:00	23-03-31 0:00:00

	A	B	C	D
1	15	1	0	360
2	15	2	2880	3264
3	20	2	4344	4728
4	15	1	0	360
5	15	2	2880	3264

Figura 5.2: El archivo `productores.csv` contiene los datos en un formato amigable al usuario, por ejemplo las fechas. El archivo `productoresPRE.csv` se obtiene del anterior y expresando las fechas con $dt = 1$ hora

En lo que resta del capítulo presentaremos brevemente cada uno de estos archivos de código y de datos, en el siguiente orden:

- archivos en los que los usuarios pueden proporcionar los datos de entrada
- archivos de datos que se producen tras procesar los anteriores y desde los cuales se pasan parámetros al modelo AMPL
- cómo los scripts “`cargarDatosCaso.run`” y “`preambulo.run`” comandan a la sesión AMPL a leer los parámetros
- como `modelo3.mod` y `modelo3b.mod` implementan el modelo desarrollado.
- finalmente los directorios `Test0`, `Test1` y `Test2`, que contienen casos de test para el modelo implementado.

5.3. Datos provistos por el usuario

El usuario puede proveer los datos para el modelo en la forma de archivos CSV. Tomemos como ejemplo el caso base de la página 3.

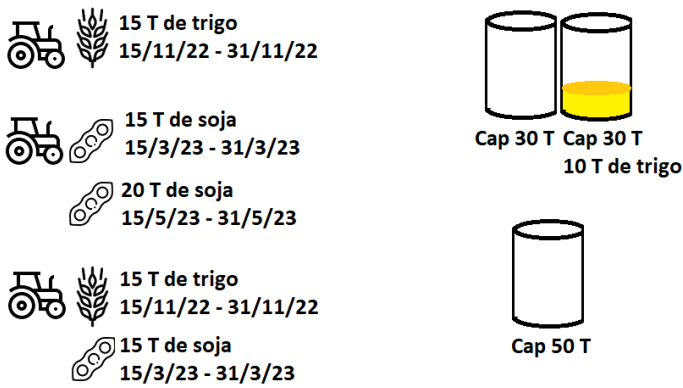


Figura 5.3: Lado izquierdo del Caso Base

Los archivos productores.csv y consumidores.csv contienen los datos relativos a los envíos y los pedidos. Sus columnas son:

- carga: tamaño en toneladas del envío o del pedido
- tgrano: tipo de grano (entero positivo)
- desde,hasta: inicio y fin de la ventana de tiempo, en formato aa-mm-dd hh:mi:ss
- tdescarga: en el caso de consumidores se indica el tiempo de descarga de un camión (suponemos que no depende del tipo de camión) en segundos.

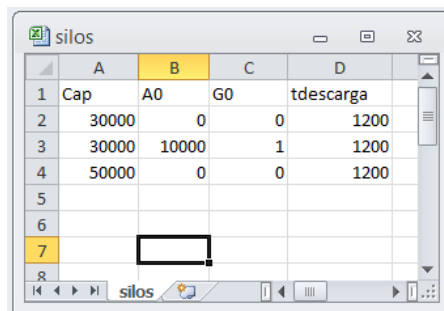
productores				
	A	B	C	D
1	carga	tgrano	desde	hasta
2	15	1	22-11-15 0:00:00	22-11-30 0:00:00
3	15	2	23-03-15 0:00:00	23-03-31 0:00:00
4	20	2	23-05-15 0:00:00	23-05-31 0:00:00
5	15	1	22-11-15 0:00:00	22-11-30 0:00:00
6	15	2	23-03-15 0:00:00	23-03-31 0:00:00

consumidores					
	A	B	C	D	E
1	carga	tgrano	desde	hasta	tdescarga
2	35	1	23-03-25 0:00:00	23-03-31 0:00:00	3600
3	50	2	23-06-01 0:00:00	23-10-06 0:00:00	1800

Figura 5.4: Archivos productores.csv y consumidores.csv para el caso base

El archivo silos.csv contiene la información de los silos. Para cada uno se conoce:

- Cap: su capacidad en toneladas
- A0: el acopio inicial del silo, en toneladas.
- G0: el tipo de grano inicial. Si el silo está vacío al inicio, se asigna $G0 = 0$ (aunque es indistinto que valor se asigne en este caso).
- tdescarga: tiempo de descarga en el silo, en segundos.

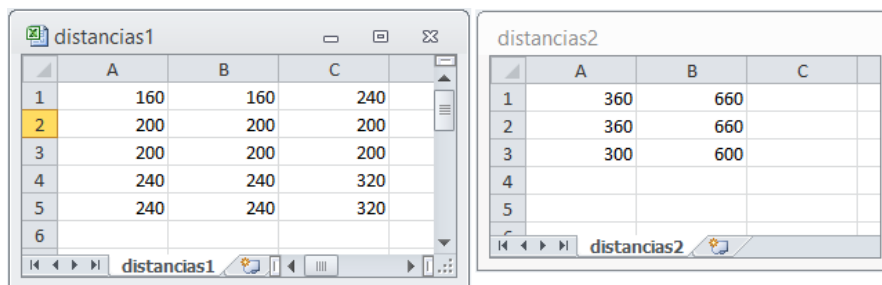


	A	B	C	D
1	Cap	A0	G0	tdescarga
2	30000	0	0	1200
3	30000	10000	1	1200
4	50000	0	0	1200
5				
6				
7				

Figura 5.5: Archivo silos.csv para el caso base

Las distancias de los productores a los silos y de los silos a los consumidores se indican en los archivos distancias1.csv y distancias2.csv respectivamente.

El archivo distancias1 es un CSV sin cabeceras, conteniendo las distancias de cada envío a cada silo. Por ejemplo, el primer renglón indica las distancias desde el punto de retiro del primer envío a cada uno de los silos. Estas distancias forman una matriz de dimensiones $I \times J$, donde I es la cantidad de envíos y J la cantidad de silos.



	A	B	C
1	160	160	240
2	200	200	200
3	200	200	200
4	240	240	320
5	240	240	320
6			

	A	B	C
1	360	660	
2	360	660	
3	300	600	
4			
5			

Figura 5.6: El archivo distancias1.csv contiene las distancias de cada cosecha a cada silo.

El archivo camiones.csv contiene información de la flota. Para cada camión se indica su capacidad en toneladas y su precio por kilómetro por tonelada. Este precio se usaría si aplicáramos una tarifa que, por ejemplo, sea lineal tanto respecto a la distancia y la carga.

	A	B	C	D
1	Cap	PrecioPorKmPorTon	Desc	
2	5	0.05	Camion tipo 1	
3	10	0.09	Camion tipo 2	
4				
5				

Figura 5.7: camiones.csv contiene información de los tipos de camión.

5.4. Datos preprocesados para AMPL

El conjunto de datos anterior representa una instancia del problema en un formato amigable a los usuarios y ajeno a las particularidades de la etapa de optimización. Por ejemplo, se puede suponer que un conjunto de datos de este estilo se podría obtener fácilmente desde un sistema dedicado a la gestión de la empresa de acopio y sus clientes.

En esta sección se muestra como los archivos de datos provistos por el usuario se procesan para que puedan ser leídos por el modelo de optimización.

Algunos de los archivos de datos producidos tendrán sufijo “PRE”, como por ejemplo “consumidoresPRE.csv”, y estarán prontos para ser cargados en una instancia del problema ejecutada en una sesión AMPL.

5.4.1. Argumentos para el preprocesamiento

Todo el preprocesamiento descrito a continuación se implementa en el módulo `preprocesamiento.py`. Su procedimiento `main()` recibe los siguientes argumentos:

- `dir`: directorio donde se encuentran los archivos de usuario a procesar. En dicho directorio se creará una subcarpeta “PRE”, donde se escribirán los archivos producidos.
- `dt`: duración de los intervalos de tiempo en los cuales se divide la duración del problema. Esta duración se expresa en segundos y su valor por defecto es 3600, es decir $dt = 1$ hora.
- `tarifa`: función que recibe una distancia y una carga y devuelve el costo del viaje. El módulo provee dos funciones de tarifa, pero eventualmente se podría pasar como argumento cualquier otra función para el cálculo de la tarifa.

Como ya señalamos en otros capítulos, el valor de dt seleccionado y la función

de tarifa provista deben elegirse adecuadamente.¹ Por otra parte, la función de tarifa debe dar preferencia a una solución que utilice menos camiones.²

5.4.2. IJKC.csv

Primero se examinan los archivos de consumidores, silos, productores y camiones para obtener el tamaño de cada conjunto. Recordar que:

- $\{1, \dots, I\} = \mathbb{I}$: el conjunto de las cosechas
- $\{1, \dots, J\} = \mathbb{J}$: el conjunto de silos
- $\{1, \dots, K\} = \mathbb{K}$: el conjunto de pedidos de exportación
- $\{1, \dots, C\} = \mathbb{C}$: el conjunto de los tipos de camiones

Los valores de I,J,K y C se escriben en ese orden en el archivo IJKC.csv

	A	B	C	D	E	F
1	5	3	2	2		
2						

Figura 5.8: Archivo IJKC.csv

5.4.3. consumidoresPRE.csv y productoresPRE.csv

Se examinan los archivos de productores y consumidores para buscar el mínimo de las fechas (fechas-hora más específicamente). Este es necesario para construir el conjunto $\mathbb{T}$ de los instantes de tiempo.

Dicha fecha se considera como el instante inicial del problema, y por tanto se le asigna el valor $t = 0$. Luego, a las demás fechas se les asigna el instante correspondiente a la cantidad de dt transcurridos desde el instante inicial.

En nuestro ejemplo vemos que la menor fecha es 22-11-15 00:00:00 y por tanto se corresponde con $t = 0$. Luego, si utilizamos $dt = 1$ hora, tenemos que a la fecha 22-11-31 00:00:00 le corresponde $t = 384$, pues dista 384 horas del instante inicial.

Luego, en los archivos de consumidores y productores se sustituyen las fechas por los enteros que las representan. También se quita el renglón de cabezales.

¹El valor de dt afecta directamente el tamaño de la dimensión tiempo. Por ejemplo si se elige $dt = 1$ hora el conjunto $\mathbb{T}$ tendrá 24 veces más elementos que si se eligiera $dt = 1$ día. Si la dimensión tiempo tiene más elementos obtendremos una **planificación más intensiva**, suponiendo que esto no genere problemas de performance por granularizar demasiado el tiempo.

²Por ejemplo, usar 1 camión de 20 toneladas debería ser más económico que usar 2 camiones de 10 toneladas.

Los archivos resultantes se guardan como productoresPRE.csv y consumidoresPRE.csv.

	A	B	C	D
1	carga	tgrano	desde	hasta
2	15	1	22-11-15 0:00:00	22-11-30 0:00:00
3	15	2	23-03-15 0:00:00	23-03-31 0:00:00
4	20	2	23-05-15 0:00:00	23-05-31 0:00:00
5	15	1	22-11-15 0:00:00	22-11-30 0:00:00
6	15	2	23-03-15 0:00:00	23-03-31 0:00:00

	A	B	C	D	E
1	carga	tgrano	desde	hasta	tdescarga
2	35	1	23-03-25 0:00:00	23-03-31 0:00:00	3600
3	50	2	23-06-01 0:00:00	23-10-06 0:00:00	1800

Figura 5.9: Archivos productoresPRE.csv y consumidoresPRE.csv

5.4.4. tiempos1 y tiempos2

Los tiempos de transporte entre productores y silos T_{ij} , y entre silos y consumidores Γ_{jk} , se guardan en los archivos tiempos1.csv y tiempos2.csv. Recordar que por tiempo de transporte no referimos al tiempo necesario para cargar un camión en el origen, realizar el viaje y descargar el grano en el destino.

Los tiempos T_{ij} se calculan a partir de las distancias contenidas en distancias1 y suponiendo que los camiones viajan a 80 km/h. Al tiempo de viaje se le suma el tiempo de descarga en el silo ³.

Por último, se redondea este tiempo a una cantidad entera de dt , de modo que se cumpla que si $t \in \mathbb{T}$ entonces $t + T_{ij} \in \mathbb{T}$ ⁴. Este procesamiento para obtener los tiempos de transporte se representa en la Figura 5.10.

5.4.5. costos1_*.csv y costos2_*.csv

Los archivos costos1_*.csv indican los costos de transporte de los productores a los silos. Se genera uno de estos archivos para cada tipo de camión. Por ejemplo, el archivo costos1.1.csv contiene los costos de los viajes productor-silo para el tipo de camión 1.

Como mencionabamos en la Sección 4.4.1, en el módulo preprocesamiento.py se proveen dos funciones a usar como cálculo de las tarifas. Una de ellas es la siguiente:

```

1 #tarifa lineal respecto a la distancia y la carga
2 #factor es el precio por kilometro por tonelada
3 def tarifaLineal(kms, tons, factor):
4     return kms*tons*factor

```

³Podría sumarse también un tiempo de carga en el lugar del productor, aunque por simplicidad no lo hemos hecho. Además se ha considerado que el tiempo de descarga en el silo depende solo del silo, y no del tipo de camión. Esto último también podría hacerse sin inconvenientes.

⁴Recordar que $\mathbb{T} = \{0, 1, 2, \dots, T_{max}\}$

distancias1

	A	B	C
1	160	160	240
2	200	200	200
3	200	200	200
4	240	240	320
5	240	240	320

silos

	A	B	C	D
1	Cap	A0	G0	tdescarga
2	30	0	0	1200
3	30	10	1	1200
4	50	0	0	1200
5				

preprocesar.main(dt = 1 hora)

tiempos1

	A	B	C
1	3	3	4
2	3	3	3
3	3	3	3
4	4	4	5
5	4	4	5

Figura 5.10: tiempos1.csv contiene los tiempos de transporte de las cosechas a los silos.

Por ejemplo, para calcular el elemento ij de costos1_1 tomamos el elemento ij de distancias1 y le aplicamos `tarifalinear()` junto con la capacidad y el precio por kilómetro y por tonelada del tipo de camión 1. El siguiente pseudocódigo muestra la obtención de los valores para costos1_1.csv:

```

1 #pseudo código
2 for i in productores:
3     for j in silos:
4         costos1_1[i,j] = tarifalinear(
5             kms = distancias1[i,j],
6             tons = camion[1].Cap,
7             factor = camion[1].PrecioPorKmPorTon
8         )

```

El archivo costos1_1 obtenido para el caso base se muestra en la Figura 5.11.

Todo el preprocesamiento descrito en esta sección se ejecutó localmente. Los archivos producidos se transfirieron al servidor Dantzig vía scp. Como mostraremos a continuación, allí se resolverá la instancia del problema usando el entorno AMPL.

5.5. Modelo

5.5.1. Lectura de los parámetros desde los archivos

AMPL ofrece un lenguaje de modelado para programación matemática. También ofrece comandos para scripting, los cuales permiten, entre otras cosas, asignar parámetros desde archivos, imprimir valores resultantes tras la resolución de una instancia y resolver una misma instancia usando distintos solvers.

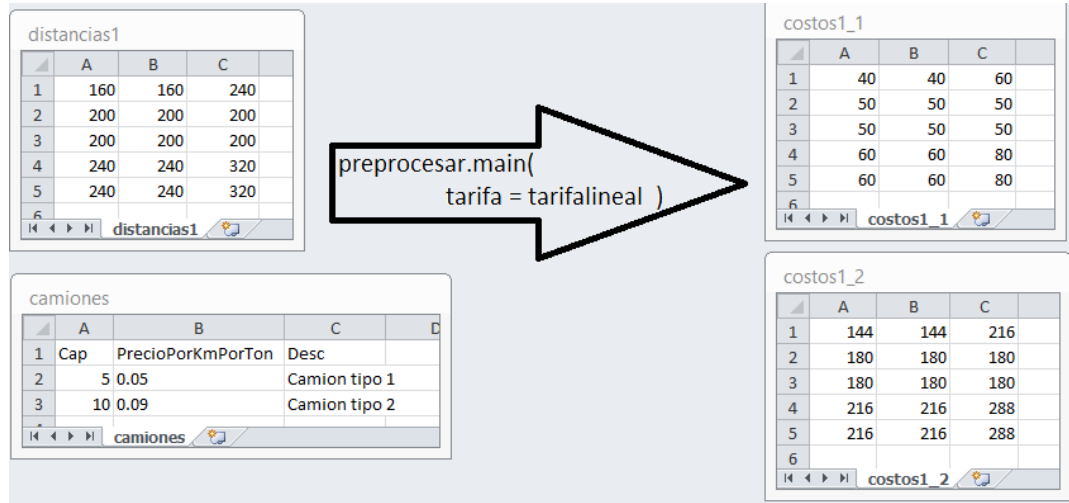


Figura 5.11: El archivo `costos1.1.csv` contiene los costos los viajes productor-silo del tipo de camion 1

Los archivos `preambulo.run` y `cargarDatosCaso.run` son scripts que se encargan de cargar los parámetros para una ejecución del modelo en una sesión AMPL. En particular, el comando `read` permite leer parámetros desde un archivo. Por ejemplo, `preambulo.run` lo usa para leer los parámetros I, J, K y C desde el archivo `IJKC.csv`. También se setea el directorio donde se encuentran los datos del caso a ejecutar.

```

1 set DIRS = {"CasoBase/"};
2 for {d in DIRS} {
3   read I,J,K,C < ( d & "IJKC.csv" );
4 }

```

Código 5.1: Script AMPL “`preambulo.run`”

Luego el script `cargarDatosCaso.run` lee el resto de los parámetros desde el directorio seteado.

```

1 for {d in DIRS} {
2   read {i in productores} (p[i], g[i], a[i], b[i]) < ( d & "
3     productoresPRE.csv" );
4   read {j in plantas} ( capSilo[j], A0[j], G0[j], tDescargaSilo[j]
5     ) < ( d & "silosPRE.csv" );
6   read {k in consumidores} ( q[k], h[k], e[k], l[k] ) < ( d & "
7     consumidoresPRE.csv" );
8   .... }

```

Código 5.2: El script “`cargarDatosCaso.run`” lee los parámetros

5.5.2. Código del modelo

El archivo modelo3.mod implementa el modelo presentado en el Capítulo 4. La traducción del modelo teórico al código AMPL es casi directa.

Las restricciones están numeradas tal como aparecen en el modelo matemático. Nos centraremos en explicar como se implementaron sus restricciones.

Usar AMPL en conjunto con el solver CPLEX permite implementar restricciones condicionales. Por ejemplo, gracias a esto, la restricción 2, relativa a las ventanas de tiempo en los productores, se implementó de forma casi idéntica a su formulación teórica, como se muestra a continuación:

$$t < a_i \vee b_i < t \Rightarrow \rho_{ijt} = 0 \quad , \forall (i, j, t) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \quad (2)$$

```
1 subject to rest2 {i in productores, j in silos, t in tiempo}:
2   t < a[i] or t > b[i] ==> rho[i,j,t] = 0;
```

Código 5.3: Implementación de la restricción 2.

Otras restricciones de tipo condicional son aquellas referentes al manejo de los tipos de granos. Por ejemplo, consideremos las restricción “Granos 2”:

$$\rho_{ijt} > 0 \Rightarrow G_j^{t+T_{ij}} = g_i \quad , (i, j, t, c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C} \quad (19)$$

Recordar que ρ_{ijt} es la cantidad de grano que sale del productor i rumbo al silo j en el instante t .

El primer intento de implementar esta restricción fue el siguiente:

```
1 # variable indicatriz de si Rho es positivo
2 # pmax es una constante cota superior de rho[i,j,t]
3 subject to rest19 {i in productores, j in silos, t in 0..Tmax-T[i,
4   j] }:
   rho[i,j,t] > 0 ==> G[j, t + T[i,j]] = g[i];
```

Código 5.4: Implementación de rho.es_positivo

Esta formulación de la restricción no fue aceptada por AMPL arrojando el siguiente error: **“cplex 20.1.0.0 logical constraint _slogcon[1] is not an indicator constraint”**.

Como se menciona en (Fourer, Bob, 2013), esto se debe a que la interfaz AMPL-CPLEX solo acepta restricciones condicionales de la siguiente forma, donde x es una variable binaria y rel es una de las relaciones $\leq, =, \geq$:

$$x = 0 \text{ o } 1 \Rightarrow A \text{ rel } B$$

Para sortear esta limitante, procedimos a definir una variable indicatriz $\rho_{es_positivo_{ijt}}$, dada por:

$$\rho_{es_positivo_{ijt}} = \begin{cases} 1, & \text{si } \rho_{ijt} > 0 . \\ 0, & \text{si } \rho_{ijt} = 0 . \end{cases}$$

El Código 5.5 implementa el comportamiento de la variable $\rho_{es_positivo_{ijt}}$.

Sea A una variable no negativa y x una variable binaria
Se cumple que

$$x = 1 \Leftrightarrow A > 0$$

es equivalente al siguiente conjunto de restricciones:

$$\begin{cases} A \leq x.M \\ x \leq A.M \end{cases}$$

Además, si $A \in \mathbb{Z}^+ \cup \{0\}$ la segunda restricción
se puede sustituir por $x \leq A$
y alcanza con que M sea una cota superior de A .

Cuadro 5.2: Restricciones “Gran M” para indicatriz de positividad

```

1 # variable indicatriz de si Rho es positivo
2 # pmax es una constante cota superior de rho[i,j,t]
3 subject to DEF_rho_es_positivo {i in productores, j in silos, t in
4   tiempo} :
5   rho[i,j,t] <= rho_es_positivo[i,j,t]*pmax;
6 subject to DEF_rho_es_positivo2 {i in productores, j in silos, t
7   in tiempo} :
8   rho_es_positivo[i,j,t] <= rho[i,j,t];

```

Código 5.5: Implementación de rho_es_positivo

Estas restricciones se basan en la formulación “Gran M” de ciertas propiedades, como la de positividad de una variable en este caso (ver Cuadro 5.2). Este tipo de restricciones pueden afectar el rendimiento del modelo, por lo cual se recomienda que el valor de M no sea excesivamente grande (Máté Hegyháti, 2022). En atención a esto, hemos seleccionado en cada caso el menor valor posible para M .

Finalmente la restricción 19 queda implementada por el Código 5.6.

```

1 # ----- Restriccion 19, GRANOS 2
2 subject to rest19 {i in productores, j in silos, t in 0..(Tmax - T
3   [i,j])}:
4   rho_es_positivo[i,j,t] = 1 ==> G[j, t + T[i,j]] = g[i];;

```

Código 5.6: Implementación de la restricción 19 “Granos 2”

Como último ejemplo consideremos otra restricción condicional, la restricción 18, “Granos 1”:

$$\rho_{ijt} > 0 \wedge \delta A_j^{t+T_{ij}-1} = 1 \Rightarrow G_j^{t+T_{ij}-1} = g_i, \forall (i, j, t, c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C} \quad (18)$$

El primer intento de implementación se muestra en el siguiente Código 5.7. Sin embargo, también arrojó el error “... not an indicator constraint”.

```

1 # ----- Restriccion 18, GRANOS 1
2 subject to rest18 {i in productores, j in silos, t in 0..(Tmax-(T[
   i,j] - 1)) }:
3   rho_es_positivo[i,j,t] = 1 and delta_A[j, t + T[i,j] - 1] ==> G
   [j, t + T[i,j] - 1] = g[i];

```

Código 5.7: Primer intento de implementación de la restricción 18

Para sortear este inconveniente primero planteamos la restricción 18 de modo equivalente, como se muestra a continuación:

$$\rho_{ijt} > 0 \Rightarrow (\delta A_j^{t+T_{ij}-1} = 1 \Rightarrow G_j^{t+T_{ij}-1} = g_i)$$

Ahora reescribimos el condicional de la derecha usando la equivalencia del Cuadro 5.3.

Sea x una variable binaria.
 Se cumple que la restricción condicional
 $x = 1 \Rightarrow A = B$
 es equivalente al siguiente conjunto de restricciones:

$$\begin{cases} A \leq B + (1 - x) \cdot M \\ A \geq B - (1 - x) \cdot M \end{cases}$$

Cuadro 5.3: Restricción “Gran M” para igualdad condicionada

Finalmente, la restricción queda implementada por el Código 5.8, donde $1..G_{max}$ son los tipos de granos.

```

1 # ----- Restriccion 18, GRANOS 1 rest 18a y 18b
2 subject to rest18a {i in productores, j in silos, t in 0..(Tmax-(T[
   i,j] - 1)) }:
3   rho_es_positivo[i,j,t] = 1 ==> G[j, t + T[i,j] - 1] <= g[i] + (1-
   delta_A[j, t + T[i,j] - 1])*Gmax;
4
5 subject to rest18b {i in productores, j in silos, t in 0..(Tmax-(T[
   i,j] - 1)) }:
6   rho_es_positivo[i,j,t] = 1 ==> G[j, t + T[i,j] - 1] >= g[i] - (1-
   delta_A[j, t + T[i,j] - 1])*Gmax;

```

Código 5.8: Implementación de la restricción 18

5.5.3. Testeo unitario del modelo

Se han implementado casos de test para verificar que el código implementa correctamente las restricciones, y para validar que estas a su vez modelan correctamente los aspectos fundamentales del problema. Estas pruebas también han sido de apoyo a la etapa de implementación y permitieron detectar casos

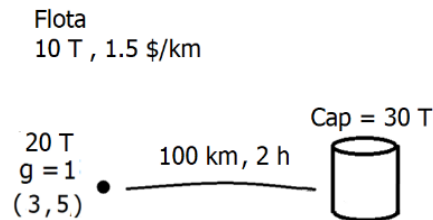


Figura 5.12: test0.01 es el primer caso del flujo Test0. El valor esperado de la función objetivo es 300.

que no eran tenidos en cuenta por las primeras versiones del código implementadas.

“Test0” es el primer flujo de test y busca verificar que:

- las ventanas de tiempo se respetan
- la cantidad de grano almacenado en los silos está bien computada a lo largo del tiempo
- que la cantidad y tipos de camiones usados es la correcta
- que la capacidad de los silos no se sobrepasa

El flujo de test inicia con una instancia del problema la cual se va modificando y se chequea que la función objetivo y la factibilidad de la solución varíe acordeamente. Por ejemplo, el primer test del flujo cuenta con un único productor y un único silo. El productor cuenta con 20 toneladas del tipo de grano 1 a recoger en la ventana de tiempo (3,5). El silo se encuentra vacío al inicio y su capacidad es 30 toneladas. La Figura 5.12 representa este test. El arco entre el nodo productor y el silo está etiquetado con la distancia y el tiempo de viaje entre ellos. La flota contiene un único tipo de camión, del cual se conoce su capacidad y su precio por kilómetro.

Luego, en el segundo test del flujo, test0.02, representado en la Figura 5.13, se agrega acopio inicial al silo.

El último test del flujo termina incluyendo productores, consumidores y dos tipos de camiones. El mismo se representa en la Figura 5.14. Invitamos al lector a que halle la solución óptima y calcule el valor objetivo. Considere con cuidado las ventanas de tiempo y los tiempos de viaje.

Para cada flujo de test se provee un script que lo ejecuta. Por ejemplo, el flujo de test “Test0” se puede ejecutar desde un terminal mediante:

```
1 $ ampl correrTest0.bat
```

El flujo de test “Test1” testea el correcto modelado de los distintos tipos de grano. En él se consideran distintos tipos de granos y se chequea que los

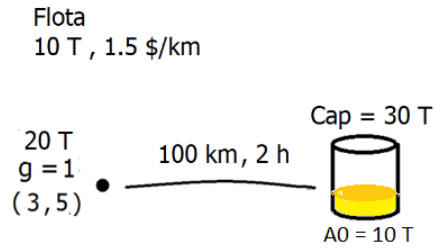


Figura 5.13: test0_02, el valor esperado de la función objetivo se mantiene en 300.

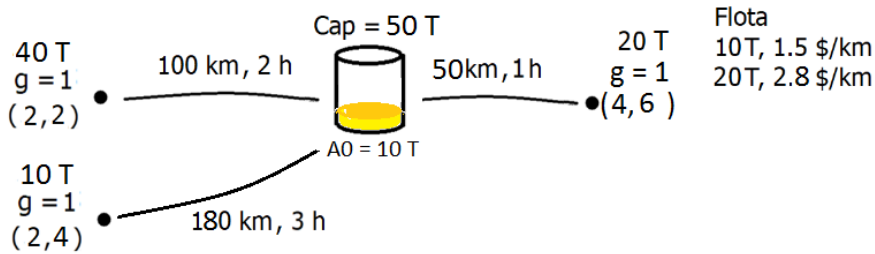


Figura 5.14: test0_16: ¿Qué viajes se realizarán y en qué momento? ¿Cuál será el valor objetivo?

mismos se almacenen por separado. A modo de ejemplo, en la Figura 5.15 se muestra la instancia del test1_05. En ella aparecen dos tipos de granos. Además, se considera flota con dos tipos de camiones.

Los conjuntos de test Test0 y Test1 buscan testear el correcto funcionamiento del Modelo 3. Recordemos que este modelo espera que los parámetros T_{ij} y Γ_{jk} , que representan los tiempos de transporte, sean positivos.

Recordemos que, como se discutió al final del Capítulo 3, es deseable que el modelo pueda recibir tiempos de transporte nulos. Para ese caso se implementó el Modelo 3b. Este modelo acepta que los tiempos de transporte sean positivos o cero. De este modo, permite que ingrese grano a los silos en el instante $t = 0$.

Con el fin de testear el Modelo 3b específicamente, se diseñó un nuevo flujo de test, “Test2”, el cual fue superado satisfactoriamente. También se le aplicó al Modelo 3b los flujos Test0 y Test1, los cuales también superó exitosamente.

Los flujos de test Test0, Test1 y Test2 están documentados en el archivo “Casos de test”, el cual se entrega junto con el directorio del proyecto.

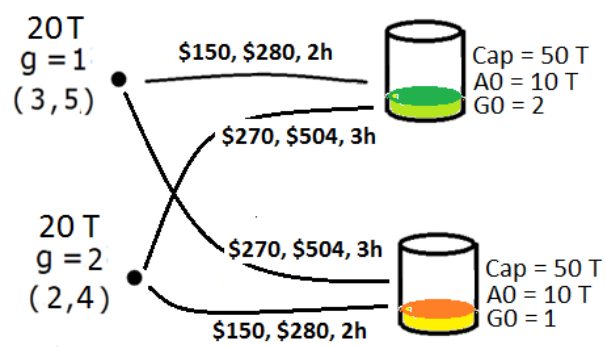


Figura 5.15: test1_05: la flota cuenta con camiones de 10 y 20 toneladas. Los arcos muestran los costos de usar cada tipo de camión. El valor objetivo esperado es 1008.

Capítulo 6

Test de Performance

En este capítulo mostraremos el estudio experimental del desempeño del modelo implementado. Se han ejecutado cinco test de performance. Cada uno consiste de una secuencia de casos en la cual se ha ido aumentando el tamaño del problema, a fin de obtener instancias cada vez mayores y medir el tiempo de resolución de cada una.

Los primeros tres test tienen una intención más de tipo exploratoria, y constituyeron las primeras observaciones sobre el desempeño del modelo y cómo este responde a cambios en los distintos parámetros.

Los test cuarto y quinto constituyen un conjunto de instancias benchmark que miden la escalabilidad, robustez y eficacia del modelo al ejecutarlo ante instancias de gran tamaño.

6.1. Observaciones teóricas sobre el tamaño del problema

Antes de pasar a los test, hagamos algunas observaciones. Viendo la especificación teórica del modelo podemos ver que el tamaño del problema (por ejemplo, el tamaño de su espacio de soluciones o la cantidad de variables y restricciones) depende de varios factores. Digamos que el problema tiene varias dimensiones y que puede crecer en cualquiera de ellas:

- La cantidad de envíos $\mathbf{I}$ y la cantidad de pedidos $\mathbf{K}$:

Dado un envío $i \in \mathbf{I}$, para cada camión necesario para su traslado, se deberá evaluar a cual de los $\mathbf{J}$ silos será enviado. Lo mismo ocurre con los pedidos. Para cada pedido y cada camión destinado a ese pedido, se debe evaluar desde cual de los $\mathbf{J}$ silos se provee.

En resumen, varias restricciones y variables están indizadas por $\mathbf{I}$ y por $\mathbf{J}$; por tanto al aumentar la cantidad de envíos o la cantidad de pedidos, la cantidad de restricciones y de variables del problema aumenta.

- La cantidad de silos **J**:

Para cada silo se deben evaluar varias restricciones. Por ejemplo, aquellas que controlan que en caso de estar llegando un viaje al silo no se sobrepase su capacidad. O si el silo ya contiene acopio, que el tipo de grano acopiado sea el mismo que el tipo de grano recibido.

- La dimensión tiempo **T**:

Recordemos que el conjunto de los instantes de tiempo se forma al dividir la duración del problema en intervalos de largo **dt**.

Nuevamente, muchas restricciones y variables están indizadas sobre **T**. Una duración del problema grande o un valor de **dt** chico puede producir un conjunto **T** con muchos elementos y por tanto aumentar mucho el tamaño del problema.

- La cantidad de tipos de camión **C** y sus capacidades:

Esto determinará de cuántas formas es posible retirar un envío desde el sitio de un productor. Por ejemplo, consideremos una cosecha de 40 T. Supongamos que la flota cuenta con camiones de 20 T únicamente. En este caso, la **única** forma de retirar la carga del envío es con dos camiones: $20 + 20$.

Sin embargo, si en la flota también hay camiones de 10 T, la cantidad de formas de retirar la cosecha ya no es única. Hay tres formas: $20 + 20$, $20 + 10 + 10$ y $10 + 10 + 10 + 10$.

Es decir, si la flota es heterogénea (cuenta con camiones de distinta capacidad) entonces la cantidad de formas de retirar un envío particular puede crecer mucho a medida que el tamaño del envío crece. Esto último hará que con solo introducir un nuevo tipo de camión, el espacio de soluciones factibles crezca mucho.

Por ejemplo, para la flota de 10 y 20 toneladas, vimos que hay 3 formas de levantar un envío de 40 toneladas. Si el envío fuera de 80 toneladas, la cantidad de formas pasa a 5. Y en general si el envío i es de p_i toneladas, habrá $\frac{p_i}{20} + 1$ formas de retirarlo con flota de 10 y 20 toneladas.

En cambio, para flota homogénea siempre hay una única combinación de camiones con la cual se puede retirar una carga p_i . En los test 2 y 3 comprobaremos como afecta al tiempo de resolución el uso de flota homogénea o heterogénea.

- El tamaño de los envíos p_i :

Al aumentar el tamaño de los envíos, la cantidad de camiones que salen del productor aumenta. Esto hará que la cantidad de combinaciones de a qué silo enviar cada uno de ellos aumente exponencialmente.

Hay que observar que esto aumenta fuertemente el tamaño del espacio de soluciones. Sin embargo, la cantidad de variables y restricciones se mantiene igual al aumentar el tamaño de los envíos. Observaremos esto

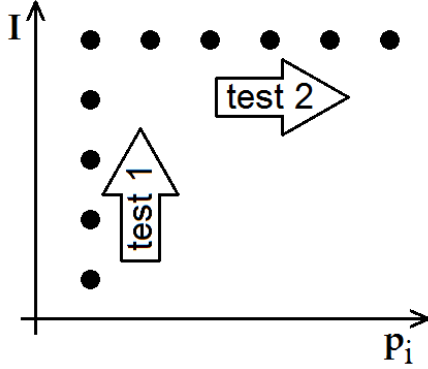


Figura 6.1: En los test 1 y 2 se aumentan la cantidad de envíos primero y el tamaño de las mismos luego. Cada punto representa una instancia de test.

en los test 2. Esto se debe a que la cantidad de variables y restricciones depende de la cantidad de envíos I y no del tamaño de los envíos p_i .

6.2. Descripción general de los test

Con el fin de mantener cierta simplicidad, los test de performance incluirán solo productores y silos; no habrá consumidores. En cada test la cantidad de silos y su capacidad se mantendrá fija. Todos estarán vacíos al inicio.

En los primeros dos test se usará flota heterogénea con camiones de 10 y 20 toneladas. Luego, a partir del tercer test en adelante usaremos solo camiones de 20 toneladas. Este cambio en el tipo de flota y otros ajustes fueron decisiones tomadas a medida que se realizaban observaciones sobre la performance del modelo.

Como mencionábamos, los test 1, 2 y 3 buscan obtener una noción inicial de como responde el modelo a cambios en los parámetros.

Como se representa en la Figura 6.1, en el test 1 iremos aumentando la cantidad de envíos I , dejando fijo el tamaño de las mismos. Luego en el test 2, fijado el valor de I , iremos aumentando los tamaños de los envíos $p_1, p_2, p_3, \dots, p_I$.

A partir del test 3 pasaremos a usar flota homogénea, ya que comprobamos experimentalmente que usar flota heterogénea es más costoso en términos de memoria.

En el test 3 continuaremos aumentando aún más el tamaño de los envíos en cada instancia del test. También realizaremos una observación interesante acerca de diferencias obtenidas en los tiempos de resolución de CPLEX, según si la secuencia de instancias se resuelve de forma independiente una de otra, cada una en una sesión, o si la secuencia de instancias se resuelve de forma iterativa en una misma sesión AMPL.

Por último, en los test 4 y 5 buscamos exigir al máximo el desempeño del

modelo. Nuevamente seguiremos el esquema de los test 1 y 2: primero aumentaremos al máximo la cantidad de envíos **I**. Luego, fijado el mayor valor de **I** para el cual pudimos resolver la instancia obtenida, pasaremos a aumentar al máximo el tamaño de los envíos.

6.3. Test de performance 1: aumentando la cantidad de envíos **I**

6.3.1. Descripción de los datos

Los productores serán idénticos, en el sentido de que todos tendrán los mismos tres envíos que realizar. Cada productor realiza los siguientes tres envíos de distintos tipos de granos:

- 100 toneladas del tipo de grano 1 a retirar entre el 1 y el 9 de marzo
- 80 toneladas del tipo de grano 2 a retirar entre el 5 y el 14 de marzo
- 40 toneladas del tipo de grano 3 a retirar entre el 10 y el 19 de marzo

Para ser más específicos, las ventanas de tiempo se representan por marcas de tiempo del estilo fecha y hora (timestamp). Por ejemplo, la primer ventana de tiempo tiene por inicio el timestamp “1 de marzo 00:00:00”, y por fin el timestamp “9 de marzo 23:59:59” (o “10 de marzo 00:00:00” si se prefiere).

En el primer test de performance iremos aumentando el número de productores, y por tanto el número de envíos **I**. Estudiaremos como aumenta el tiempo de ejecución, y buscaremos hallar un valor de **I** adecuado para los siguientes test de performance. Buscamos un valor de **I** que sea un poco exigente para su resolución pero no excesivamente, de modo de poder realizar varias ejecuciones en un tiempo razonable.

Usaremos en principio una flota heterogénea, con camiones de 10 y 20 toneladas.

Usaremos silos de capacidad 15.000 toneladas. Como no deseamos que la cantidad de silos sea muy ajustada, hemos decidido usar 40 silos distribuidos en 8 plantas (5 silos por planta). Los silos estarán vacíos al inicio.

Las distancias de cada productor a cada planta se han sorteado en el rango de 150 a 400 km y cada 10 km.

Se usa una flota con dos tipos de camión: de 10 y 20 toneladas. Para cada par productor-silo y cada tipo de camión se calculó el costo del viaje respectivo usando la tarifa mostrada en el Cuadro 6.1. La misma es una tarifa real. Pero dado que es lineal respecto a la cantidad de toneladas, al precio resultante para los camiones de 20 toneladas se le aplicó un descuento de 5 %. De este modo es menos costoso usar un camión de 20 toneladas que usar dos de 10 toneladas.

6.3. TEST DE PERFORMANCE 1: AUMENTANDO LA CANTIDAD DE ENVÍOS **I**63

distancia(km)	precio (US\$ / ton)
1 - 5	7,14
6 - 10	7,64
11 - 15	8,21
....	
181 - 200	25,95
> 200	0.078 por km
> 300	0.064 por km

Cuadro 6.1: Tarifa real. La tarifa completa se encuentra en el anexo.

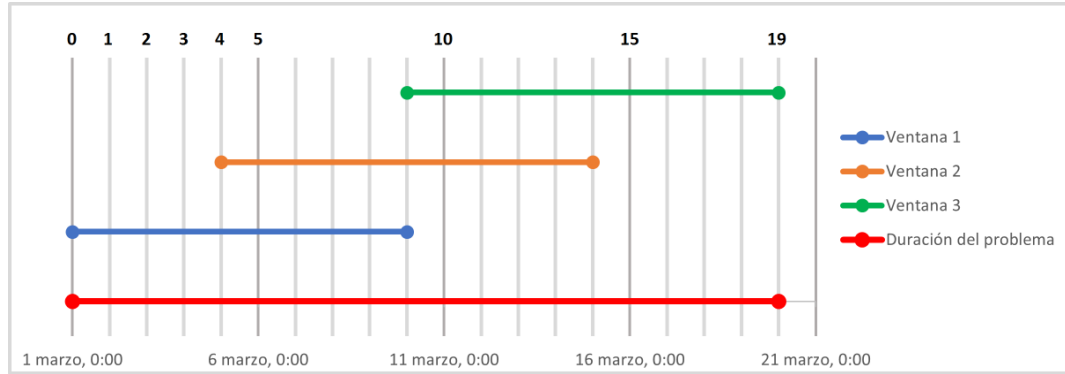


Figura 6.2: Ventanas de tiempo de los test de performance

Por último, dado que la duración del problema es del 1 al 19 de marzo, se ha usado $dt = 1$ día. Ya que elegimos $dt = 1$ día, los tiempos de transporte T_{ij} se han seteado en cero.

Así, el inicio y fin del problema vienen dados por las marcas de tiempo 1 de marzo 00:00 y 20 de marzo 00:00. Y el índice de los instantes de tiempo $\mathbb{T}$ es $\{0, 1, 2, \dots, 19\}$, donde 0 y 19 se corresponden con el inicio y el fin del problema, como se muestra en la Figura 6.2.

Finalmente, las ventanas de tiempo se mapean a sus correspondientes valores enteros:

- 1 de marzo 00:00 al 10 de marzo 00:00 $\rightarrow (0, 9)$
- 5 de marzo 00:00 al 15 de marzo 00:00 $\rightarrow (4, 14)$
- 10 de marzo 00:00 al 20 de marzo 00:00 $\rightarrow (9, 19)$

6.3.2. Casos del test

Fijados todos los parámetros, se ha ejecutado una secuencia de 10 casos. En cada caso se agregan 10 productores y 30 envíos, partiendo de 10 productores

y 30 envíos en el primer caso, para llegar a 100 productores y 300 envíos en el último caso.

Para ejecutar estos casos se provee la carpeta “*TestPerformance1*” conteniendo los datos de distancias, envíos, silos, y la subcarpeta “*PRE*” con los datos preprocesados para el modelo. También se provee el script *correrTestPerformance1.bat* que se encarga de ejecutar los casos 1 a 10.

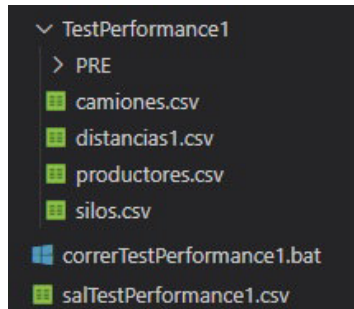


Figura 6.3: Test Performance 1: script y datos provistos.

Observando el código del script *correrTestPerformance1.bat* podemos notar que tras resolver cada caso, y al momento de cargar los datos para el caso siguiente, se vuelve a setear el tamaño del índice I , y por tanto todos los parámetros y variables que dependen del mismo también son reseteados.

```

1 for { caso in casos } {
2   let I:= 10 * 3 * caso;
3   reset data p,g,a,b;
4   read {i in 1..I} (p[i], g[i], a[i], b[i]) < "productoresPRE.csv"
5   ;
6   reset data T;
7   for {i in 1..I, j in silos} let T[i,j]:= 0 ;
8   reset data costs1;
9   for {c in 1..C} {
10    read {i in 1..I, j in silos} costs1[i,j,c] < ( "costos1_" & c &
11    ".csv"); }
12   solve;
13 }
```

Código 6.1: *correrTestPerformance1.bat*, diremos que las instancias se resuelven de forma independiente

6.3.3. Resultados

En el archivo *salTestPerformance1.csv* se escriben los resultados de la secuencia de casos ejecutados por el script. Dichos resultados se muestran en el Cuadro 6.2.

Recordar que I es la cantidad de envíos (tres por productor). Las columnas NVARS y NCONS indican la cantidad de variables y de restricciones, respecti-

6.3. TEST DE PERFORMANCE 1: AUMENTANDO LA CANTIDAD DE ENVÍOS **I** 65

Caso	I	NVARS	NCONS	Costo	$T_{ejecucion}$	Λ_{max}
1	30	100800	77630	38532	79.16	10
2	60	196800	149660	73492	299.16	20
3	90	292800	221690	113696	440.34	28
4	120	388800	293720	157035	1426.94	26
5	150	484800	365750	196042	1371.60	35
6	180	580800	437780	229197	963.14	40
7	210	676800	509810	265278	2118.21	40
8	240	772800	581840	307059	3241.32	89
9	270	868800	653870	344318	2998.82	140
10	300	964800	725900	386099	3597.32	125

Cuadro 6.2: Test de performance 1: envíos de 100, 80 y 40 toneladas. En cada caso se agregan 30 envíos. Flota con camiones de 20 y 10 toneladas. En todos los casos, el óptimo hallado es global.

vamente, de la instancia resultante. La cantidad de variables y de restricciones suele tomarse como una medida del tamaño del problema.

Las últimas tres columnas nos dan datos de la solución hallada ¹. Costo es el valor óptimo de la función objetivo que arrojó la ejecución del modelo. En todos los casos de este y los demás test, el valor óptimo hallado es un óptimo global. $T_{ejecucion}$ es el tiempo de resolución en segundos. La columna Λ_{max} indica el máximo número de camiones que llega a un silo en un período dt ($dt = 1$ día) para la solución óptima hallada².

En la Figura 6.4 se grafica el crecimiento del tiempo de ejecución respecto a la cantidad de envíos **I**.

Eligiremos usar $I = 150$ para el segundo test de performance, pues el tiempo de ejecución obtenido (1371 segundos $\approx$ 23 minutos) parece adecuado como punto de partida. Como veremos a continuación, en el segundo test de performance mantendremos fijo la cantidad de envíos **I** e iremos aumentando el tamaño de los mismos.

¹Suponiendo que existe una solución factible para ese caso y que pudo ser hallada con los recursos disponibles. Aparecerán casos no factibles recién en el Test 5.

²Es interesante conocer este valor ya que nuestro modelo no tiene en cuenta la tasa de llegada de camiones a las plantas y la posible formación de colas en las mismas. Fácilmente puede incluirse alguna restricción que acote la cantidad de camiones que llegan a un silo por unidad de tiempo. En el anexo se trata el tema de la tasa de llegada de camiones a las plantas y la eventual formación de colas. Allí se propone un conjunto de restricciones que modelan el comportamiento de la cola y el tiempo de cola de los vehículos que llegan a la planta.



Figura 6.4: Test Performance 1: Tiempo de ejecución en función de I

6.4. Test de performance 2: aumentando el tamaño de los envíos p_i

6.4.1. Descripción de los datos

En el segundo test de performance, habiendo fijado la cantidad de envíos en 150, procederemos a ir aumentando el tamaño de los mismos y estudiaremos como se comporta el tiempo de ejecución.

Mantendremos flota de dos tipos de camión. Hemos decidido bajar la cantidad de silos a 20 (5 plantas con 4 silos cada una). Además, hemos disminuído la capacidad de los silos a 10000 toneladas. Disminuiremos un poco las distancias productor-silo; ahora serán sorteadas entre 10 y 100 km, cada 10 km.

6.4.2. Casos del test

Como en el test anterior, cada productor tiene 3 envíos, los cuales serán de 100, 80 y 40 toneladas al inicio. Luego, en cada caso iremos aumentando el tamaño de los envíos en un 50% de su valor inicial. El Cuadro 6.3 esquematiza el aumento en el tamaño de los envíos.

Envío	caso 1	caso 2	caso 3	caso 4	...
de grano 1	100 T	150 T	200 T	250 T	...
de grano 2	80 T	120 T	160 T	200 T	...
de grano 3	40 T	60 T	80 T	100 T	...

Cuadro 6.3: Test de Performance 2: el tamaño de los envíos aumenta

Para ejecutar este segundo test de performance se provee la carpeta con los

6.5. TEST 3: AUMENTANDO EL TAMAÑO DE LOS ENVÍOS CON FLOTA HOMÓGENEA.67

datos *TestPerformance2*, el script *correrTestPerformance2.bat* y un archivo con los resultados de cada caso, *salTestPerformance2.csv*.

En el Cuadro 6.4 se indican los resultados. Nuevamente, para cada caso se indican valores que dan una idea del tamaño de las instancias: I, cantidad de variables y de restricciones y el total de grano enviado. Podemos ver que en todos los casos se tienen la misma cantidad de variables y restricciones. Esto se debe a que el valor de I no cambia.

Caso	I	Grano Total (grano 1-2-3)	NVARS	NCONS	Costo	tejecución	Λ_{max}
1	150	11000 (5000 + 4000 + 2000)	242400	182950	81301	464.79	75
2	150	16500 (7500+6000+3000)	242400	182950	125647	209.37	128
3	150	22000 (10000+8000+4000)	242400	182950	162602	691.55	88
4	150	27500 (12500+10000+5000)	242400	182950	—	—	—

Cuadro 6.4: Test de Performance 2: el tamaño de los envíos aumenta. En los casos resueltos se llegó al óptimo global.

La tercer columna indica la cantidad total de grano a transportar y dichas cantidades separadas por tipo de grano. A mayor cantidad de grano, mayor cantidad de camiones serán necesarios. Además como estamos usando flota heterogénea, la cantidad de formas en la que se puede retirar un envío crece con el tamaño del envío.

Para cada caso resuelto se indica el valor de la función objetivo y el tiempo de ejecución (Columnas 6 y 7). Nuevamente Λ_{max} es la máxima cantidad de vehículos que llega a un silo en un día.

Para el caso 2, si bien estamos mostrando un resultado, este lo obtuvimos en el cuarto intento. Los intentos anteriores dieron código de salida 9, el cual esta asociado a problemas de memoria (Fourer, Bob, 2011). En la Figura 6.5 se muestra la salida de CPLEX del tercer intento fallido, junto con la del comando dmesg en consola. El proceso cplex fue matado por excederse de memoria. La memoria alojada alcanzó 61,16 GiB y la memoria virtual total llegó a 92.39 GiB. El caso 4 se intentó ejecutar en tres oportunidades resultando siempre en Out Of Memory.

Dadas las dificultades que parece acarrear el uso de flota heterogénea, en el test de performance 3 pasaremos a usar flota homogénea. Veremos que con este cambio, lograremos superar sin problemas el caso 4.

6.5. Test 3: Aumentando el tamaño de los envíos con flota homogénea.

En este nuevo test, y al igual que en el test de performance 2, dejaremos fija la cantidad de envíos en $I = 150$ e iremos aumentando el tamaño de los mismos.

```

nohup.out
14
15 caso 2
16 Cosechas por productor: 150, 120 y 60
17 COSECHA TOTAL = 16500
18 CPLEX 20.1.0.0: error running cplex:
19   termination code 9
20 <BREAK>

[10500530.000968] [2225323] 5000415 2225323 102350 0 851968 97660 0 ampl
[10500530.001494] [2225324] 5000415 2225324 24221167 16034513 193851392 8074492 0 cplex
[10500530.002018] oom-kill:constraint=CONSTRAINT_NONE,nodemask=(null),cpuset=/,mems_allowed=0,global_oom,ta
sk_memcg=/user.slice/user-5000415.slice/session-1196.scope,task=cplex,pid=2225324,uid=5000415
[10500530.002554] Out of memory: Killed process 2225324 (cplex) total-vm:96884668kB, anon-rss:64138052kB, f
ile-rss:0kB, shmem-rss:0kB, UID:5000415 pgtables:189308kB oom_score_adj:0
[10500531.681294] oom_reaper: reaped process 2225324 (cplex), now anon-rss:0kB, file-rss:0kB, shmem-rss:0kB
[aldo.plazzotta@dantzig ampl]$

```

Figura 6.5: Test 2 caso 2: varias veces se obtuvo código 9. Recién en el cuarto intento el proceso cplex arrojó una solución.

Usaremos flota homogénea: solo camiones de 20 toneladas.

Al igual que en el test de performance 2, se partirá de envíos con cierto tamaño en el caso 1, las cuales se aumentarán en un 50 % del valor inicial en cada uno de los casos subsiguientes. Dado que los tamaños de los envíos deben ser múltiplos de la capacidad del camión, los envíos de grano 1 cambiarán a un tamaño inicial de 120 T. El Cuadro 6.5 esquematiza como aumenta en el tamaño de los envíos en la secuencia de casos:

Envíos	caso 1	caso 2	caso 3	caso 4	...
de grano 1	120 T	180 T	240 T	300 T	...
de grano 2	80 T	120 T	160 T	200 T	...
de grano 3	40 T	60 T	80 T	100 T	...

Cuadro 6.5: Test de Performance 3: el tamaño de los envíos aumenta al igual que en el test 2, pero con flota homogénea esta vez. Los envíos de grano 1 inician en 120 toneladas.

6.5.1. Resultados

En el Cuadro 6.6 se muestran los resultados de ejecutar los casos 1 a 10 del Test de performance 3. Una primer observación es que, como mencionabamos antes, al pasar de flota heterogénea en el Test 2 a flota homogénea en este Test 3, vemos que la barrera del caso 4 pudo superarse.

6.5. TEST 3: AUMENTANDO EL TAMAÑO DE LOS ENVÍOS CON FLOTA HOMÓGENEA.69

Caso	p_1	p_2	p_3	Grano Total (grano 1 + 2 + 3)	Costo	tejecucion	Λ_{max}
1	120	80	40	12000 (6000 + 4000 + 2000)	88692	2719.24	102
2	180	120	60	18000 (9000 + 6000 + 3000)	133038	581.01	171
3	240	160	80	24000 (12000 + 8000 + 4000)	177384	719.63	204
4	300	200	100	30000 (15000 + 10000 + 5000)	221730	1645.66	190
5	360	240	120	36000 (18000 + 12000 + 6000)	266076	674.98	328
6	420	280	140	42000 (21000 + 14000 + 7000)	310422	4402.99	280
7	480	320	160	48000 (24000 + 16000 + 8000)	354768	1168.70	384
8	540	360	180	54000 (27000 + 18000 + 9000)	399114	4148.05	446
9	600	400	200	60000 (30000 + 20000 +10000)	443460	1076.85	210
10	660	440	220	66000 (33000 + 22000 +11000)	487806	3721.83	500

Cuadro 6.6: Test Performance 3: casos 1 a 10, ejecutados por separado

Cada uno de estos casos fue ejecutado en una sesión AMPL separada. Diremos que **los casos se han ejecutado de forma independiente**. Otra posibilidad es ejecutar los casos de **forma iterativa**, en una misma sesión AMPL.

Las siguientes porciones de código muestran la diferencia entre ambas formas de ejecutar los casos. En el Código 6.2 se ejecuta unicamente el caso 10: se asignan los tamaños de los 150 envíos, se resuelve la instancia y la sesión termina.

```

1 I := 150;
2 caso := 10;
3 for { iter in 0..I/3-1 }
4 { let p[3*iter + 1] := 120 + (caso - 1)*60;
5   let p[3*iter + 2] := 80 + (caso - 1)*40;
6   let p[3*iter + 3] := 40 + (caso - 1)*20;
7 }
8 solve;

```

Código 6.2: Caso ejecutado de forma independiente: se asigna los tamaños de los envíos, se resuelve la instancia y la sesión AMPL termina.

En el Código 6.3 se ejecutan los casos 1 a 10 de forma iterativa, reasignando los tamaños de los envíos entre una instancia y la siguiente³. Vemos que todas las instancias se resuelven de forma secuencial en la misma sesión AMPL.

Notamos un resultado muy interesante: al ejecutar varios casos de forma iterativa, el primer caso tiene un tiempo de resolución dentro de lo normal (del orden de varios minutos), pero los casos subsiguientes muestran un tiempo de resolucion notablemente bajo (del orden de solo algunos segundos).

Esto puede observarse en el Cuadro 6.7. En la Columna 1 se muestran los tiempos de resolución independiente copiados del Cuadro 6.6. Observar que estos tiempos son de casi 10 minutos el menor de ellos (caso 2) hasta más de 70

³Esta resolución iterativa de instancias es posible ya que la cantidad de envíos I no cambia.

```

1 I := 150;
2 casos := 1..10
3 for { caso in casos }
4 {
5     for { iter in 0..I/3-1 }
6     { let p[3*iter + 1] := 120 + (caso - 1)*60;
7       let p[3*iter + 2] := 80 + (caso - 1)*40;
8       let p[3*iter + 3] := 40 + (caso - 1)*20;
9     }
10    solve;
11 }

```

Código 6.3: Casos ejecutados de forma iterativa: varios casos se instancian y resuelven en una misma sesión AMPL.

minutos (caso 6).

En la Columna 2 se muestran los tiempos de resolución de los casos 1 a 10 ejecutados de forma iterativa. Al compararlos con los tiempos de la primer columna vemos que el primer caso da un tiempo de ejecución similar en ambas ejecuciones (2719 y 2715 segundos respectivamente). Pero al ejecutar iterativamente, los tiempos de resolución del caso 2 en adelante son de solo unos segundos.

Este es un fenómeno muy interesante, pues las instancias son las mismas pero ejecutarlas de forma iterativa permite bajar notablemente los tiempos de ejecución de la segunda iteración en adelante. Incluso, para los casos 7 y 10 el tiempo de resolución independiente es un 32% mayor al tiempo de resolución iterativo acumulado (Columna 3).

Caso	$t_{ejecucion}$ independiente	$t_{ejecucion}$ iterativa	Acumulado iterativo
1	2719.24	2715.04	2715.04
2	581.01	9.09	2724.12
3	719.63	9.29	2733.41
4	1645.66	7.53	2740.94
5	674.98	8.86	2749.80
6	4402.99	9.51	2759.31
7	1168.69	9.81	2769.12
8	4148.05	25.10	2794.22
9	1076.85	9.26	2803.48
10	3721.83	11.01	2814.49

Cuadro 6.7: Test Performance 3: casos 1 al 10 ejecutados iterativamente tienen tiempos de ejecución notablemente bajos en comparación con la ejecución por separado.

En el gráfico de la Figura 6.6 se comparan los tiempos de ejecución de los

casos de forma independiente contra el acumulado de la ejecución iterativa. Se observa que en el caso 7 y en el caso 10, resolver el caso por separado demora más que resolverlo a él y todos los casos anteriores en forma iterativa.

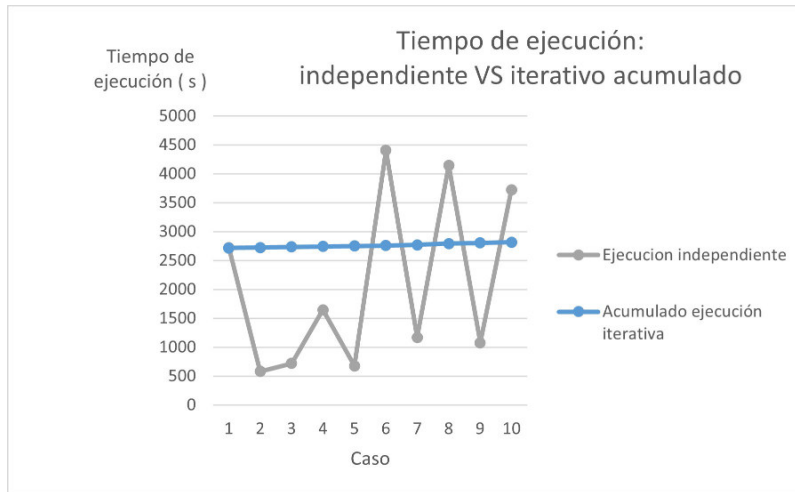


Figura 6.6: Test de performance 3: ejecución independiente comparada con ejecución iterativa.

Este fenómeno podría deberse a que al cargar y resolver las instancias de forma iterativa, cada ejecución del proceso CPLEX toma como punto de partida la solución óptima hallada en el caso anterior. Esta estaría “muy cerca” de la solución óptima de la instancia actual en el espacio de soluciones (si bien no es solución factible de la instancia actual).

Lo anterior es solo una suposición que entendemos deber estar relacionado a detalles de implementación del solver CPLEX o de la interfaz AMPL/CPLEX. Sería interesante estudiar si el mismo fenómeno sucede si se utiliza AMPL con otro solver distinto de CPLEX.

6.6. Análisis de robustez bajo instancias más grandes

Los test de performance anteriores conformaron un primer acercamiento al desempeño del modelo y tuvieron un carácter más de tipo exploratorio.

En esta sección describiremos los test 4 y 5, que buscan estudiar la robustez del modelo ante instancias de tamaño cada vez mayor. Seguiremos el camino de los test 1 y 2, aumentando primero la cantidad de envíos I , para luego aumentar

el tamaño de los mismos. En cada test aumentaremos al límite el tamaño de las instancias a resolver, hasta alcanzar una que, siendo factible, no se pueda resolver con los recursos computacionales disponibles.

6.6.1. Test de performance 4: aumentando al límite el número de envíos

En este test retomaremos la idea de ir aumentando la cantidad de envíos. En el test 1 habíamos llegado a $I = 150$ envíos. Partiremos ahora de $I = 300$ envíos y aumentaremos de a 150.

Al igual que en el test 3:

- las envíos serán inicialmente de 120, 80 y 40 toneladas por cada productor.
- El conjunto de los silos estará formado por 20 silos agrupados en 5 plantas (4 silos por planta). Los silos tendrán capacidad de 10 mil toneladas y estarán vacíos al inicio.
- Y usaremos solo camiones de 20 toneladas pues, como vimos, usar flota homogénea tiene mejor desempeño.

6.6.2. Resultados

En la tabla del Cuadro 6.8 se muestran los resultados obtenidos. Vemos que la instancia para $I = 300$ envíos se resuelve en 17783 segundos (4 horas con 56 minutos). El caso 2 con $I = 450$ envíos arrojó error Out Of Memory luego de ejecutar por alrededor de 35 horas.

Fijaremos $I = 300$ envíos y procederemos en el test 5 a aumentar los tamaños de los envíos $p_1, p_2, p_3, \dots, p_I$, hasta alcanzar la mayor instancia que sea posible resolver. Recordamos que la cantidad de variables y de restricciones no varía al dejar I fijo. Por tanto, durante todo el test 5 las cantidades de variables y la de restricciones serán 362400 y 363100 respectivamente.

Caso	I	NVARS	NCONS	p_1	p_2	p_3	Grano Total (grano 1 + 2 + 3)	Costo	$t_{ejecucion}$	Λ_{max}
1	300	362400	363100	120	80	40	24000 (12000 + 8000 + 4000)	193116	17783.33	174
2	450	542400	543250	120	80	40	36000 (18000 + 12000 + 6000)	XX	XX	XX

Cuadro 6.8: Test de Performance 4: resultados

6.6.3. Test de performance 5: aumentando el tamaño de los envíos p_i

Retomaremos el test anterior fijando $I = 300$ envíos e iremos aumentando el tamaño de los envíos. Similar al test 3, los envíos de los tipos de grano 1, 2 y 3

tendrán tamaños iniciales de 120, 80 y 40 toneladas respectivamente. Luego, en cada caso aumentaremos el tamaño de los envíos un 50 % de su valor inicial.

El Cuadro 6.9 muestra los casos ejecutados y resueltos. Los casos 1 a 18 pudieron resolverse llegando al óptimo global siempre. El caso 19, si bien es factible, no pudo resolverse, arrojando error Out Of Memory. Así, el caso 18 es el de mayor tamaño que se pudo resolver con los recursos computacionales disponibles.

Caso	p_1	p_2	p_3	Grano Total (grano 1 + 2 + 3)	Costo	$t_{ejecucion}$	$t_{acumulado}$	Λ_{max}
1	120	80	40	24000 (12000 + 8000 + 4000)	193116	4336.16	4336.16	114
2	180	120	60	36000 (18000 + 12000 + 6000)	289674	18.30	4354.47	171
3	240	160	80	48000 (24000 + 16000 + 8000)	386232	20.98	4375.45	228
4	300	200	100	60000 (30000 + 20000 + 10000)	482790	18.20	4393.65	285
5	360	240	120	72000 (36000 + 24000 + 12000)	579348	18.24	4411.89	288
6	420	280	140	84000 (42000 + 28000 + 14000)	675906	26.18	4438.08	336
7	480	320	160	96000 (48000 + 32000 + 16000)	772464	17.29	4455.37	384
8	540	360	180	108000 (54000 + 36000 + 18000)	869022	17.65	4473.02	432
9	600	400	200	120000 (60000 + 40000 + 20000)	965580	21.25	4494.26	480
10	660	440	220	132000 (66000 + 44000 + 22000)	1062138	38.39	4532.65	486
11	720	480	240	144000 (72000 + 48000 + 24000)	1158696	33.09	4565.75	553
12	780	520	260	156000 (78000 + 52000 + 26000)	1255254	1668.96	6234.71	711
13	840	560	280	168000 (84000 + 56000 + 28000)	1351812	397.34	6632.05	714
14	900	600	300	180000 (90000 + 60000 + 30000)	1448370	22.26	6654.31	750
15	960	640	320	192000 (96000 + 64000 + 32000)	1544928	300.70	6955.02	750
16	1020	680	340	204000 (102000 + 68000 + 34000)	1641486	39.30	6994.32	750
17	1080	720	360	216000 (108000 + 72000 + 36000)	1738177	48.32	7042.64	750
18	1140	760	380	228000 (114000 + 76000 + 38000)	1834602	15341.77	22384.41	750
19	1200	800	400	240000 (120000 + 80000 + 40000)	—	—	—	—

Cuadro 6.9: Test de Performance 5

Describamos mejor la instancia del caso 18. La misma consiste de 300 envíos ($I = 300$), provenientes de 100 productores. Cada productor tiene tres envíos, uno de cada tipo de grano enumerados 1, 2 y 3. Los envíos de tipo de grano 1 son, cada uno, de 1140 toneladas (como se muestra en la columna " p_1 "). Los envíos de tipo de grano 2 y 3 son de 800 y 400 toneladas respectivamente (columnas p_2 y p_3). Por tanto, de cada productor han partido 114 camiones. Las cantidades de granos transportadas son 114 mil, 76 mil y 38 mil toneladas de cada tipo de grano respectivamente. La producción total suma 228 mil toneladas de granos, requiriendo un total de 11400 viajes de camión.

Para cada día t , la variable A_j^t nos indica cuánto grano ha ingresado al silo j hasta el día t inclusive. Dado que no hay retiros de grano de los silos, A_j^t coincide con la cantidad de grano almacenado el día t . En particular, para

$t = 19$ (instante final del problema), los valores A_j^t y G_j^t nos indican el estado final de los silos.

Para los casos del test 5 hemos guardado el estado final de los silos (ver directorio “TestPerformance5/PRE/EstadoSilos”). En la Figura 6.7 se representa el estado final de los silos agrupados por planta para la solución óptima global alcanzada en la instancia del Caso 18 del Cuadro 6.9.

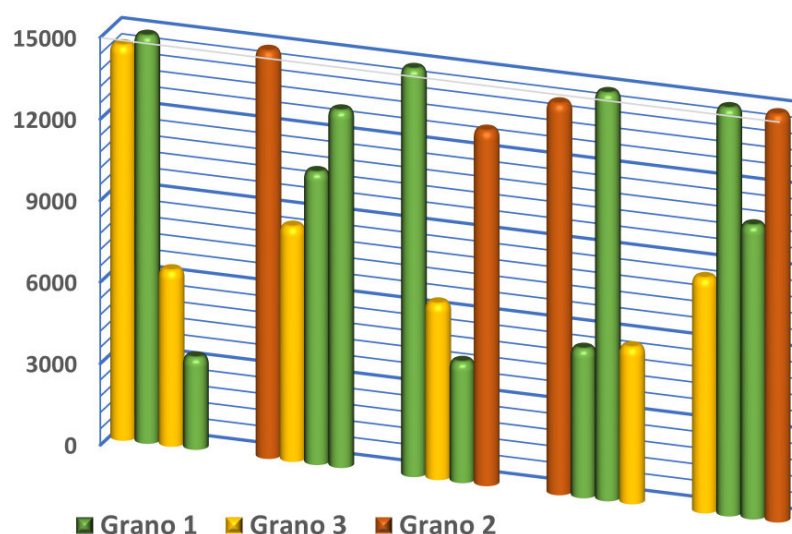


Figura 6.7: Test de Performance 5, caso 18: Estado final de los silos, agrupados por planta.

Analicemos ahora los tiempos de ejecución obtenidos durante el test. Nuevamente hemos seguido el esquema de ejecución iterativa (los casos se ejecutan en secuencia, tras actualizar los p_i). Esperábamos que se repitiera el fenómeno observado en el test 3: tras resolver el primer caso, los casos siguientes se resuelven en solo unos segundos.

Esto efectivamente ocurrió. Ver que la instancia del caso 1 demoró en resolverse 4336 segundos, pero los casos siguientes (hasta el caso 11 incluido) demoraron solo unos segundos. En la Figura 6.8 vemos gráficamente el tiempo de ejecución para cada caso, representados estos por la cantidad total de grano transportada.

Podemos observar que tras el caso 1, los tiempos de resolución son prácticamente nulos. Estos valores bajos se mantienen hasta el caso 11 incluido. Habíamos supuesto que este fenómeno se podría deber a que, ante una nueva instancia, CPLEX tomaba como punto de partida la solución del caso anterior. Si bien la solución del caso anterior no es factible para el nuevo caso, podemos

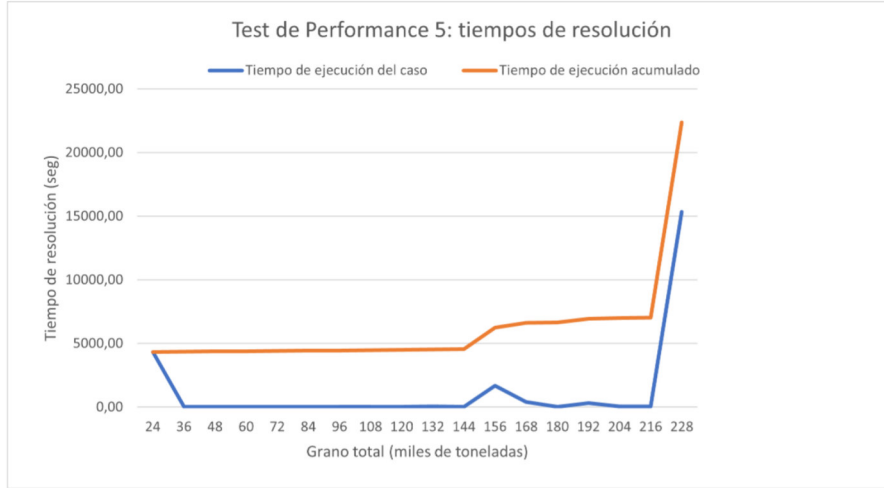


Figura 6.8: Tiempos de resolución

ver que alcanza con sumarle algunos camiones a la misma para obtener una solución factible del caso nuevo. Podemos suponer que, en el espacio de las soluciones x_{ijtc} , la solución óptima del caso anterior se encuentra muy cerca de una solución factible (y posiblemente óptima) del nuevo caso.

Capítulo 7

Conclusiones

En este proyecto hemos definido un problema de asignación de producciones de distintos tipos de granos a plantas de acopio, las cuales a su vez proveen a eventuales consumidores de granos.

Hemos creado un modelo de programación lineal entera que resuelve dicho problema y cuya solución devuelve una planificación en el tiempo indicando qué viajes de carga deben realizarse de productores a plantas y de plantas a consumidores.

Este modelo tiene gran versatilidad en tanto es un modelo de asignación pero con una componente temporal, pues permite considerar aspectos como los tiempos de viaje y las ventanas de tiempo en los productores y consumidores. Además permite considerar flota heterogénea.

En lo que respecta a su uso el modelo permite elegir la granularidad con la que será tratada la dimensión tiempo. Más específicamente, el valor dt funciona como una “unidad de tiempo”, en la que se desea resolver una instancia práctica del problema. La solución del problema representará una planificación de qué viajes de camión estarán partiendo o arribando en cada período de tiempo t .

7.1. Trabajo realizado

En los primeros capítulos se desarrolló modelo teórico. Una de sus componentes más novedosas es el conjunto de restricciones que modelan el manejo de los distintos tipos de granos. Estas cumplen con el requerimiento fundamental de que distintos tipos de granos sean almacenados por separado.

Para la versión final del modelo se incorporó la posibilidad de aceptar flota heterogénea y se refinó el diseño de modo de aceptar setear los tiempos de transporte como nulos. Dadas las variables que se definieron, la solución hallada tras

la ejecución del modelo provee una planificación de qué viajes se deben realizar: qué origen, qué destino y en qué momento. Esta solución también nos indicará qué cantidad y qué tipo de grano hay en cada planta a lo largo del tiempo.

Se realizó la implementación del modelo en la plataforma AMPL/CPLEX. Dicha implementación se apoyó en una serie de test unitarios, previamente diseñados.

También se definió un esquema de datos en formato CSV para proveer un medio amigable a los usuarios en el cual presentar los datos al modelo. Este conjunto de datos funciona como una capa de presentación, ya que permite a un usuario humano u otro sistema de software interactuar con nuestro modelo. Dichos datos deben procesarse para generar los parámetros de la instancia a resolver. Se implementaron scripts python necesarios para dicho procesamiento. La implementación del modelo junto con estos scripts se entregan como productos de este proyecto.

Finalmente, se definieron un conjunto de instancias benchmark para investigar el desempeño del modelo y su respuesta ante distintos tipos de instancias. Estas instancias consistían de cantidades de granos producidos cada vez mayores, llegando a una instancia límite que pudimos resolver con los recursos computacionales disponibles.

Pudimos observar que el uso de flota heterogénea parece afectar gravemente el desempeño del modelo, a diferencia de la flota homogénea con la cual pudimos resolver instancias más grandes con mayor facilidad.

También se observó un fenómeno por el cual al resolver secuencialmente un conjunto particular de instancias, todas excepto la primera tenían un tiempo de resolución notablemente bajo. Nos referimos a este fenómeno como *tiempo de la ejecución iterativa*. Se conjeturó que el mismo se debía a una suerte de cercanía entre la solución hallada en la resolución de una instancia y el punto de partida para la próxima resolución.

Los scripts y datos necesarios para correr todos los test de performance, así como los archivos con la salida de los test se entregan junto con el proyecto.

7.2. Propuestas para trabajos a futuro

En este proyecto se planteó crear un modelo teórico de optimización. Como todo modelo, se abstrae parte de una realidad compleja y se la representa en una forma más elemental pero manejable. A partir de los resultados de este proyecto se realizan tres propuestas de trabajos a futuro :

- Se realizaron test de performance estudiando la respuesta del modelo frente al aumento del volumen de la producción (más productores y con mayor

rendimiento). Sería interesante realizar más testeo, agregando consumidores o con flota heterogénea. Otro aspecto a testear sería el de aplicar el modelo con una granularidad temporal más pequeña, como $dt = 1$ hora.

- Respecto a las aplicaciones prácticas de este modelo, sería de gran interés aplicarlo a un caso real del sector agropecuario. Para esto sería necesario recolectar datos respecto a un conjunto de productores, el cronograma de cosechas, la infraestructura de acopio y distribución: rutas, flota disponibles, capacidad de las plantas, etc.
- Por último, respecto al desarrollo teórico, el modelo podría modificarse y extenderse para reflejar otros aspectos de la realidad. Por ejemplo, hemos considerado que el tiempo de descarga en planta de un viaje es constante y que no depende de la cantidad de viajes que hayan llegado a la planta en ese momento. En el anexo se proponen un conjunto de restricciones que buscan modelar la aparición de colas en las plantas o acotar la tasa de llegada de vehículos. Un trabajo a futuro podría consistir en extender el actual modelo para incorporar dichas restricciones, implementarlo y testearlo.

7.2.1. A modo de cierre

Entendemos que se cumplió con los objetivos y alcance del proyecto, en el tiempo planificado y con los recursos disponibles.

El producto obtenido se compone por el desarrollo teórico del modelo, su implementación en software, los datos arrojados por los test y las instancias de benchmark definidas.

Entendemos que el modelo producido es de gran aplicabilidad práctica y abre varias posibilidades en tanto a futuros desarrollos teóricos y aplicaciones prácticas.

Referencias

- Asgari, N., Farahani, R. Z., Rashidi-Bajgan, H., y Sajadieh, M. S. (2013). Developing model-based software to optimise wheat storage and transportation: A real-world application. *Applied Soft Computing*, 13(2), 1074-1084. Descargado de <https://www.sciencedirect.com/science/article/pii/S1568494612004504> doi: <https://doi.org/10.1016/j.asoc.2012.10.002>
- Bayá, G., Canale, E., Nesmachnow, S., Robledo, F., y Sartor, P. (2022). Production optimization in a grain facility through mixed-integer linear programming. *Applied Sciences*, 12(16). Descargado de <https://www.mdpi.com/2076-3417/12/16/8212> doi: 10.3390/app12168212
- Fisher, M. L., Jaikumar, R., y Van Wassenhove, L. N. (1986). A multiplier adjustment method for the generalized assignment problem. *Management Science*, 32(9), 1095-1103.
- Fourer, Bob. (2011). *Error running cplexamp: Termination code 9*. https://groups.google.com/g/ampl/c/6W4qI0ssLio/m/_iQX6IkxcWcJ. (Accesed: 2022-10-02)
- Fourer, Bob. (2013, mayo). *The standard ampl/cplex interface will accept certain kinds of logical constraints ...[Comentario en el foro 'Problem with AMPL and logical constraint slogcon[1] is not an indicator constraint']*. Google Grupos. <https://groups.google.com/g/ampl/c/HEIaYj13WNA/m/4kC9S3KilaIJ>. (Accedido: 2-10-2022)
- Jablonsky, J. (2015, 01). Benchmarks for current linear and mixed integer optimization solvers. *Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis*, 63, 1923-1928. doi: 10.11118/201563061923
- Jaques, L. B., Coradi, P. C., Eguilhor Rodrigues, H., Dubal, I., Padia, C., Lima, R., y Souza, G. (2022, 04). Post-harvesting of soybean seeds - engineering, processes technologies, and seed quality: a review. *International Agrophysics*, 36, 59-81. doi: 10.31545/intagr/147422
- Jian, F., y Jayas, D. (2021). *Grains: Engineering fundamentals of drying and storage*. doi: 10.1201/9781003186199
- Karande, A., Krishna, A., Jayasurya, R., Gopan, G., Gopinath, M. V., Kumar, S., ... Varaprasad, G. (2019). Performance analysis of storage warehouses in a food grain supply chain using data envelopment analysis. En *2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN)* (p. 1-4). doi: 10.1109/ICSCAN.2019.8878776

- Kovalyov, M., y Pesch, E. (2010, 10). A generic approach to proving np-hardness of partition type problems. *Discrete Applied Mathematics*, 158, 1908-1912. doi: 10.1016/j.dam.2010.08.001
- Lamsal, K., Jones, P., y Thomas, B. (2015, 07). Harvest logistics in agricultural systems with multiple, independent producers and no on-farm storage. doi: 10.13140/RG.2.1.5061.3606
- Miplib – 2010*. (s.f.). <https://miplib.zib.de/>. (Accedido: 2022-11-10)
- Miplib – the mixed integer programming library*. (s.f.). <https://miplib2010.zib.de/>. (Accedido: 2022-11-10)
- Mogale, D., Kumar, M., Kumar, S. K., y Tiwari, M. K. (2018). Grain silo location-allocation problem with dwell time for optimization of food grain supply chain network. *Transportation Research Part E: Logistics and Transportation Review*, 111, 40-69. Descargado de <https://www.sciencedirect.com/science/article/pii/S1366554517306361> doi: <https://doi.org/10.1016/j.tre.2018.01.004>
- Mogale, D., Kumar, S. K., Márquez, F. P. G., y Tiwari, M. K. (2017). Bulk wheat transportation and storage problem of public distribution system. *Computers & Industrial Engineering*, 104, 80-97. Descargado de <https://www.sciencedirect.com/science/article/pii/S0360835216305058> doi: <https://doi.org/10.1016/j.cie.2016.12.027>
- Máté Hegyháti. (2022, febrero). *Advanced modelling techniques: big m constraints*. https://hegyhati.github.io/IMOLS/pages/modelling_bigM.html. (Accedido: 2-10-2022)
- Percus, A., Istrate, G., y Cristopher, M. (2006). *Computational complexity and statistical physics (santa fe institute studies in the sciences of complexity proceedings)*. USA: Oxford University Press, Inc.
- Rong, A., Akkerman, R., y Grunow, M. (2011). An optimization approach for managing fresh food quality throughout the supply chain. *International Journal of Production Economics*, 131(1), 421-429. Descargado de <https://www.sciencedirect.com/science/article/pii/S0925527309004290> (Innsbruck 2008) doi: <https://doi.org/10.1016/j.ijpe.2009.11.026>
- Seda, M. (2022). The assignment problem and its relation to logistics problems. *Algorithms*, 15(10). Descargado de <https://www.mdpi.com/1999-4893/15/10/377> doi: 10.3390/a15100377
- Yousaf, Z., Saleh, N., Ramazan, A., y Aftab, A. (2016, 10). Postharvesting techniques and maintenance of seed quality. doi: 10.5772/64994

Anexo: Tasa de llegada de camiones y formación de colas en las plantas

Durante el desarrollo anterior del modelo, debimos discretizar el tiempo, dividiéndolo en intervalos de duración dt . De esta manera generamos el conjunto de instantes de tiempo $\mathbb{T} = 0 \dots T_{max}$, donde T_{max} es la duración del problema expresada en períodos de largo dt . La duración del problema no es más que el lapso de tiempo en el que potencialmente puede ocurrir el inicio o fin de un viaje de carga.

Por tanto, al momento de instanciar el problema se debe definir adecuadamente que valor de dt se utilizará. Este funciona como una unidad de tiempo que define que tan fina será la planificación de los viajes que devuelva la solución. La elección de su valor depende también de la duración del problema y de los recursos computacionales disponibles.

Por ejemplo, se podría elegir un dt de una hora para problemas de planificación más intensiva. Por ejemplo, si las ventanas de tiempo en los productores y consumidores estuvieran más separadas en el tiempo, usar dt de una semana podría ser suficiente; como en el caso de ejemplo de la figura 1.1, donde las ventanas de tiempo son de algo más de dos 2 semanas, y además estos períodos están separados por meses de diferencia. Aquí una planificación semanal sería suficiente. Esto es saber cada semana qué camiones salen de los productores y qué camiones salen de los silos. En este caso, los tiempos de transporte deben considerarse nulos ya que sino serían redondeados a una semana.

A continuación se describe la posibilidad de modelar colas en los silos. Este modelado depende principalmente de la relación entre dt y el tiempo de descarga en las plantas.

.1. Tiempo de descarga en planta y tasa de llegada de vehículos

Si bien cada planta cuenta con varios silos, supondremos que las mismas cuentan con un único punto de descarga, al que llamaremos “servidor”. Sea $1..P$ un índice sobre el conjunto de plantas. Para cada planta $\pi \in 1..P$, definamos μ_π como el tiempo de descarga en la planta.

Para cada silo j , sea $planta_j \in 1..P$ la planta a la que pertenece. Entonces, si la carga de un camión está destinada al silo j , su tiempo de descarga será el valor μ de $planta_j$. Sin perder formalidad lo anotaremos como $\mu_{planta(j)}$.

A modo de ejemplo, supongamos que para toda planta π tenemos $\mu_\pi = 15$ minutos ¹.

Redefinamos el parámetro T_{ij} como la suma del tiempo de carga en el productor i más el tiempo de viaje desde i hasta j (es decir, hasta la planta del silo j). Observar que T_{ij} ya no incluye el tiempo de descarga en la planta.

Si en el instante t llega un camión a la planta del silo j , proveniente del productor i , es porque partió de i en el instante $t - T_{ij}$.

Entonces, la cantidad de camiones cuya carga está asignada al silo j que lleguen a planta en el instante t vienen dada por:

$$\sum_{i \in \mathbb{I}} \sum_{c \in C} x_{i,j,t-T_{ij},c}, \quad \forall (j, t) \in \mathbb{J} \times \mathbb{T}$$

Por tanto, sumando dicha expresión sobre todos los silos de una misma planta obtenemos la cantidad de vehículos que han llegado a planta en el instante t . En otras palabras, si dada una planta π y un instante t , definimos X_π^t la cantidad de camiones que llegan a la planta en el instante t , entonces se cumple que:

$$X_\pi^t = \sum_{j \in \mathbb{J}: planta_j = \pi} \sum_{i \in \mathbb{I}} \sum_{c \in C} x_{i,j,t-T_{ij},c}, \quad \forall (\pi, t) \in \{1..P\} \times \mathbb{T} \quad (1)$$

.2. Caso fácil: $dt \geq \mu_\pi$

En este caso, un período dt es suficiente tiempo para que uno o más camiones descargue. En este caso alcanza con una restricción que acote la tasa de llegada de camiones a la planta:

$$X_\pi^t \leq \frac{dt}{\mu_\pi}, \quad \forall (\pi, t) \in \{1..P\} \times \mathbb{T}$$

¹Observar que por simplicidad hemos supuesto que el tiempo de descarga depende a lo sumo de la planta y no del tipo de camión, por ejemplo.

Por ejemplo, supongamos $dt = 60$ minutos. Como $\mu_\pi = 15$ minutos, tenemos que en un período t pueden descargar hasta 4 camiones. Si la tasa de llegada se mantiene no mayor a 4, entonces no se formarán colas que crezcan indefinidamente. Eventualmente en instantes puntuales pueden haber camiones en espera, pero terminado el período en que arribaron a planta todos habrán sido atendidos.

.3. Caso difícil: $dt < \mu_\pi$

Supongamos ahora que dt es menor μ_π . En este caso un período dt no es suficiente para completar la descarga de un camión. Es más, la descarga podría requerir varios períodos dt .

Supondremos además que μ es una cantidad entera de dt . Es decir $\mu = k * dt$ con $k \in \{2, 3, \dots\}$. Supongamos por ejemplo que $dt = 5$ minutos y $\mu_\pi = 15$ minutos.

Lo que haremos es exigir que a lo sumo un camión llegue a la planta en cada instante t . Incluimos la siguiente restricción:

$$X_\pi^t \leq 1, \forall (\pi, t) \in \{1..P\} \times \mathbb{T} \quad (2)$$

Así podemos asignar a cada camión que llegue a planta un tiempo de espera en cola λ_π^t . La variable λ_π^t indica el tiempo en cola que tendrá **el camión**² que llegue a la planta π en el instante t , si eventualmente llega uno. Buscaremos luego una expresión para λ_π^t .

Por tanto, si un camión sale de i rumbo al silo j^3 en el instante $t - T_{ij}$ entonces:

- llegará a planta en el instante t .
- tendrá un tiempo de cola $\lambda_{planta(j)}^t$
- y su carga estará almacenada en el silo en el instante $t + \lambda_{planta(j)}^t + \mu_{planta(j)}$

Observemos que a lo sumo un camión puede salir de i rumbo a j en el instante $t - T_{ij}$, por la restricción 2. En ese caso su carga se almacenará en el instante indicado anteriormente. Obtenemos por tanto la siguiente restricción:

$$x_{i,j,t-T_{ij},c} = 1 \Rightarrow \alpha_j^{t+\lambda_{planta(j)}^t+\mu_{planta(j)}} = CapCamion_c, \forall (i,j,t,c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C} \quad (3)$$

²Es uno a lo sumo por la restricción (2)

³“Rumbo a j ” significa que su carga está destinada al silo j

Recordar que α_j^t es la cantidad de grano que ingresa al silo j en el instante t (es decir en el período $(t - 1, t]$). La restricción 3 modela el caso en que α_j^t sea no nulo. El caso $\alpha_j^t = 0$ lo veremos luego.

Cabe notar que por la restricción 2 hemos acotado a que X_π^t valga 0 o 1. Por tanto, esta variable funciona como una indicatriz de si llegó un camión a la planta π en el instante t . Es decir:

$$X_\pi^t = \begin{cases} 1, & \text{si en el instante } t \text{ llega un camión a la planta } \pi. \\ 0, & \text{sino.} \end{cases}$$

Usaremos esta indicatriz en las siguientes secciones.

.4. Largo de cola y tiempo de servicio

La siguiente imagen esquematiza el modelo de colas de una planta $\pi \in \{1..P\}$. Existe un único servidor con un tiempo de atención μ_π . El mismo atiende un camión a la vez.

Supongamos que llega un camión en el instante t . Si en ese instante no hay vehículos en la cola y el servidor está ocioso, el camión recién llegado será atendido de inmediato. En caso contrario, deberá unirse a la fila y esperar su turno, teniendo un tiempo de cola no nulo.



Definamos, para cada planta π y cada instante t , las siguientes variables:

- L_π^t = Largo de la cola en el instante t (es decir, cantidad de camiones en cola)
- U_π^t = Tiempo de servicio restante para el camión que está siendo atendido. Es decir, tiempo que falta hasta que el camión que está descargando termine de hacerlo. Si en el instante t no hay un camión descargando, entonces $U_\pi^t = 0$
- λ_π^t = Tiempo en cola de un camión que llegue en el instante t

Supongamos que en el instante t llega un camión al silo j . El mismo tendrá un tiempo en cola λ_π^t dado por la siguiente expresión:

$$\lambda_{\pi}^t = (L_{\pi}^t - 1) * \mu_{\pi} + U_{\pi}^t, \forall (\pi, t) \in \{1..P\} \times \mathbb{T} \quad (4)$$

De aquí en más y por claridad, omitiremos el índice π en λ, μ, L y U . La ecuación anterior queda expresada como:

$$\lambda^t = (L^t - 1) * \mu + U^t, \forall t \in \mathbb{T} \quad (4b)$$

Resta obtener expresiones para L^t y U^t

4.1. Modelando el comportamiento de U^t

Supongamos que en el instante t , tenemos $U^t = 1$ Entonces en el próximo instante, $t + 1$, el servidor se habrá liberado y pasará una de las siguientes cosas:

- o bien el servidor pasará a atender a otro camión, si ya había alguno en cola o si justo llega uno en ese instante,
- o bien el servidor pasará a estado ocioso.

En el primer caso tendremos $U^{t+1} = \mu$, y en el 2do caso tendremos $U^{t+1} = 0$. Las siguientes restricciones modelan estos casos, donde a la indicatriz X^{t+1} también le omitimos el subíndice π :

$$U^t = 1 \wedge (L^t > 0 \vee X^{t+1} = 1) \Rightarrow U^{t+1} = \mu, \forall t \in \mathbb{T} \quad (5b)$$

$$U^t = 1 \wedge \neg(L^t > 0 \vee X^{t+1} = 1) \Rightarrow U^{t+1} = 0, \forall t \in \mathbb{T} \quad (6b)$$

Supongamos ahora que $U^t = 0$ (el servidor entró en estado ocioso pues cuando se liberó no había vehículos que atender). En el siguiente instante $t + 1$ podrá pasar una de dos cosas:

- Llega un vehículo en el instante $t + 1$ y pasa a ser atendido, provocando que $U^{t+1} = \mu$
- No llega un vehículo en el instante $t + 1$ y el servidor sigue ocioso: $U^{t+1} = 0$

Podemos representar ambos casos con una única restricción

$$U^t = 0 \Rightarrow U^{t+1} = \mu * X^{t+1}, \forall t \in \mathbb{T} \quad (7b)$$

Por último, en el caso que U^t no sea ni cero ni uno, simplemente se decrementará en el instante siguiente:

$$U^t > 1 \Rightarrow U^{t+1} = U^t - 1, \forall t \in \mathbb{T} \quad (8b)$$

Las restricciones 5b a 8b modelan correctamente el comportamiento de U^t . Las hemos numerado con “b” pues omitimos el subíndice π . Sobre el final de este anexo presentaremos todas las restricciones sin omisiones.

Antes de pasar al comportamiento de L^t , volvamos a la restricción 3 y revisemos el comportamiento de la variable α . Habíamos definido que

$$x_{i,j,t-T_{ij,c}} = 1 \Rightarrow \alpha_j^{t+\lambda_{planta(j)}^t + \mu_{planta(j)}} = CapCamion_c, \forall (i,j,t,c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C} \quad (3)$$

Esto computaba correctamente que, si un camión llega en el instante t , su carga se computa como ingresada en el silo en el instante adecuado, $t + \lambda_{planta(j)}^t + \mu_{planta(j)}$.

Sin embargo esta restricción no evita que α tome valores no nulos en otros casos. Para ello consideremos que si α en el instante actual es no nulo, es porque el servidor se acaba de liberar; es decir, en el instante anterior un camión estaba en el servidor pero le faltaba un dt más para terminar de descargar. La siguiente restricción modela esto y evita que α tome valores no nulos cuando no debe:

$$\alpha_j^t > 0 \Rightarrow U_{planta(j)}^{t-1} = 1, \forall (j,t) \in \mathbb{J} \times \mathbb{T} \quad (9)$$

4.2. Modelando el comportamiento de L^t

Primero notemos que L^t y L^{t+1} difieren a lo sumo en uno. En efecto, si el largo de la cola crece, decrece o se mantiene igual depende de si el servidor se libera y/o llega un camión.

Consideremos el Cuadro 1, donde 1 y 0 en la primera y segunda columna representan verdadero y falso respectivamente.

Este cuadro nos da la pauta de cómo son las restricciones que modelan el comportamiento de L . Recordar que “Llega vehículo en $t+1$ ” está representado por $X^{t+1} = 1$, y que “El servidor se libera en $t+1$ ” es lo mismo que decir $U^t = 1$.

Además en el segundo caso, como la cola decrece pediremos también que $L^t > 0$. Por tanto, el conjunto de restricciones buscadas es:

$$X^{t+1} = 0 \wedge U^t < 1 \Rightarrow L^{t+1} = L^t, \forall t \in \mathbb{T} \quad (10b)$$

$$X^{t+1} = 0 \wedge U^t = 1 \wedge L^t > 0 \Rightarrow L^{t+1} = L^t - 1, \forall t \in \mathbb{T} \quad (11b)$$

$$X^{t+1} = 1 \wedge U^t < 1 \Rightarrow L^{t+1} = L^t + 1, \forall t \in \mathbb{T} \quad (12b)$$

$$X^{t+1} = 1 \wedge U^t = 1 \Rightarrow L^{t+1} = L^t, \forall t \in \mathbb{T} \quad (13b)$$

Llega vehículo en $t+1$	El servidor se libera en $t+1$	$L^{t+1}..?.L^t$
0	0	$L^{t+1} = L^t$
0	1	$L^{t+1} = L^t - 1$
1	0	$L^{t+1} = L^t + 1$
1	1	$L^{t+1} = L^t$

Cuadro 1: Entre instantes consecutivos, el largo de la cola puede aumentar, disminuir o mantenerse igual.

.5. Bosquejo de un modelo con colas en los silos

En resumen, es posible modelar las colas de camiones en las plantas. Para ellos exigiremos que en cada instante de tiempo, a los sumo un camión llegue a la planta.

A su vez el parametro T_{ij} incluye solo los tiempos de carga en i y de viaje de i a j . Los tiempos de cola y de descarga del camión se consideran por separado, siendo variable el tiempo de cola.

μ es el tiempo de descarga en la planta; eventualmente puede depender de cada planta y por eso anotamos μ_π

A modo de síntesis se muestran a continuación las restricciones aquí presentadas. Se muestran sin omitir el subíndice π .

El modelo con colas se obtiene agregando estas restricciones al Modelo 3 (o 3b). Eventualmente, alguna de las restricciones del Modelo 3 será sustituida por estas, como por ejemplo las que setean el valor de α_j^t .

Además otras restricciones se pueden agregar. Por ejemplo, restricciones que acoten el tiempo de cola o el tiempo de actividad del conductor (tiempo de carga, más tiempo de viaje, más tiempo de cola, más tiempo de descarga).

Aquí se han presentado algunos lineamientos para agregar colas en las plantas al modelo ya desarrollado. Implementar, afinar y testear estas modificaciones escapan al alcance de este proyecto. Pero se consideran de gran valor el aporte del desarrollo teórico aquí presentado.

.6. Restricciones para un modelo con colas en las plantas

$$X_{\pi}^t = \sum_{j \in \mathbb{J}: planta_j = \pi} \sum_{i \in \mathbb{I}} \sum_{c \in C} x_{i,j,t-T_{ij},c}, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (1)$$

$$X_{\pi}^t \leq 1, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (2)$$

$$x_{i,j,t-T_{ij},c} = 1 \Rightarrow \alpha_j^{t+\lambda_{planta(j)}^t + \mu_{planta(j)}} = CapCamion_c, \quad \forall(i, j, t, c) \in \mathbb{I} \times \mathbb{J} \times \mathbb{T} \times \mathbb{C} \quad (3)$$

$$\lambda_{\pi}^t = (L_{\pi}^t - 1) * \mu_{\pi} + U_{\pi}^t, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (4)$$

$$U_{\pi}^t = 1 \wedge (L_{\pi}^t > 0 \vee X_{\pi}^{t+1} = 1) \Rightarrow U_{\pi}^{t+1} = \mu_{\pi}, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (5)$$

$$U_{\pi}^t = 1 \wedge \neg(L_{\pi}^t > 0 \vee X_{\pi}^{t+1} = 1) \Rightarrow U_{\pi}^{t+1} = 0, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (6)$$

$$U_{\pi}^t = 0 \Rightarrow U_{\pi}^{t+1} = \mu_{\pi} * X_{\pi}^{t+1}, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (7)$$

$$U_{\pi}^t > 1 \Rightarrow U_{\pi}^{t+1} = U_{\pi}^t - 1, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (8)$$

$$\alpha_j^t > 0 \Rightarrow U_{planta(j)}^{t-1} = 1, \quad \forall(j, t) \in \mathbb{J} \times \mathbb{T} \quad (9)$$

$$X_{\pi}^{t+1} = 0 \wedge U_{\pi}^t < 1 \Rightarrow L_{\pi}^{t+1} = L_{\pi}^t, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (10)$$

$$X_{\pi}^{t+1} = 0 \wedge U_{\pi}^t = 1 \wedge L_{\pi}^t > 0 \Rightarrow L_{\pi}^{t+1} = L_{\pi}^t - 1, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (11)$$

$$X_{\pi}^{t+1} = 1 \wedge U_{\pi}^t < 1 \Rightarrow L_{\pi}^{t+1} = L_{\pi}^t + 1, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (12)$$

$$X_{\pi}^{t+1} = 1 \wedge U_{\pi}^t = 1 \Rightarrow L_{\pi}^{t+1} = L_{\pi}^t, \quad \forall(\pi, t) \in \{1..P\} \times \mathbb{T} \quad (13)$$

Anexo: código AMPL del modelo

```
1 param I >= 0;
2 param J > 0;
3 param K >= 0 ;
4 param C > 0;
5
6 set productores := 1 .. I ;
7 set silos := 1 .. J ;
8 set consumidores := 1 .. K;
9 set camiones := 1 .. C;
10
11 param p{productores};
12 param g{productores};
13 param a{productores};
14 param b{productores};
15
16 # pmax se usa para las restriccion big-M que modelan a
17   rho_es_positivo
18 param pmax = max{i in productores} p[i];
19
20 param A0{silos} >=0;
21 param G0{silos} >=0;
22 param capSilo{silos } > 0;
23 param tDescargaSilo{silos } > 0;
24
25 param q{consumidores};
26 param h{consumidores};
27 param e{consumidores} ;
28 param l{k in consumidores} > e[k] - 1;
29
30 # qmax se usara con un proposito similar a pmax
31 param qmax = max{k in consumidores} q[k];
32
33
34 param Gmax = max (max{i in productores} g[i] , max{j in silos } G0[
35   j] , max {k in consumidores} h[k] );
36 set granos = 0..Gmax;
37
38 param capCamion{c in camiones} ;
39 param descCamion{c in camiones};
```

```

39
40 param T {productores , silos } >= 0;
41 param T2 {silos , consumidores} >= 0;
42
43 param costs1{productores,silos ,camiones};
44 param costs2{silos ,consumidores,camiones};
45
46
47 param Tmax =max( max{i in productores , j in silos } ( b[i] + T[i,j]
48   ) , max{k in consumidores } l[k] ) ;
49 set tiempo = 0 .. Tmax;
50
51 var x{i in productores , j in silos , t in tiempo ,c in camiones}
52   integer >= 0;
53 var rho_es_positivo{i in productores , j in silos , t in tiempo }
54   binary;
55 var rho{i in productores , j in silos , t in tiempo} integer >= 0 ;
56
57 var y{j in silos , k in consumidores ,t in tiempo ,c in camiones}
58   integer >= 0;
59 var phi{j in silos , k in consumidores , t in tiempo} integer >= 0 ;
60 var phi_es_positivo{j in silos , k in consumidores , t in tiempo}
61   binary;
62
63 var alfa{j in silos , t in tiempo } >= 0;
64 var A{ j in silos , t in tiempo } >= 0;
65 var r{ j in silos , t in tiempo } >= 0;
66 var R{ j in silos , t in tiempo } >= 0;
67
68 var delta_A{j in silos , t in tiempo} binary;
69 var G{j in silos , t in tiempo} in granos;
70
71 minimize Cost:
72
73 sum{i in productores , j in silos , t in tiempo , c in camiones} (
74   costs1[i,j,c]*x[i,j,t,c] )
75 + sum{ j in silos , k in consumidores ,t in tiempo , c in camiones}
76   ( costs2[j,k,c]*y[j,k,t,c] ) ;
77
78 subject to rest2 {i in productores , j in silos , t in tiempo}:
79   t < a[i] or t > b[i] ==> rho[i,j,t] = 0;
80
81 subject to rest3 {i in productores , j in silos , t in tiempo}:
82   sum{c in camiones} (x[i,j,t,c]*capCamion[c]) = rho[i,j,t];
83
84 subject to rest4 {i in productores}:
85   sum{j in silos , t in tiempo} rho[i,j,t] = p[i];
86
87 subject to rest6{j in silos , k in consumidores , t in tiempo , c in
88   camiones}:
89   t + T2[j,k] < e[k] or t + T2[j,k] > l[k] ==> phi[j,k,t] = 0;
90
91 subject to rest7 {j in silos ,k in consumidores , t in tiempo}:
92   sum{c in camiones} (y[j,k,t,c]*capCamion[c]) = phi[j,k,t];
93
94 subject to rest8 {k in consumidores}:
95   sum{j in silos , t in tiempo} phi[j,k,t] = q[k];

```

```

88
89 # -----
90 # FUNCIONES DE ACOPIO Y RETIRO
91 # -----
92
93 subject to rest9 {j in silos , t in tiempo}:
94     alfa[j,t] = sum{i in productores} (if t >= T[i,j] then rho[i,
95         j, t - T[i,j] ] else 0 ) ;
96
97 subject to rest10 {j in silos , t in tiempo} :
98     A[j,t] = A0[j] + sum{ iter in 0 .. t} alfa[j,iter];
99
100 # -----
101 # definicion de r_jt y R_jt , funciones de retiro
102
103 subject to rest11 {j in silos , t in tiempo}:
104     r[j,t] = sum{k in consumidores} phi[j,k,t] ;
105
106 subject to rest12 {j in silos , t in tiempo} :
107     R[j,t] = sum{ iter in 0 .. t} r[j,iter];
108
109
110 # -----
111 # No se supera capacidad de los silos
112
113 subject to rest13 {j in silos , t in tiempo} :
114     0 <= A[j,t] - R[j,t] <= capSilo[j];
115
116
117 # -----
118 # definicion de delta_A, restricciones 15 y 16 implementadas con
119 # Conditional constraints
120 subject to defDeltaA1 {j in silos , t in tiempo} :
121     #delta_A[j,t] = 0 ==> A[j,t] = R[j,t] ;
122     A[j,t] - R[j,t] <= delta_A[j,t]* capSilo[j] ;
123
124 subject to defDeltaA2 {j in silos , t in tiempo} :
125     #delta_A[j,t] = 1 ==> A[j,t] < R[j,t] ;
126     delta_A[j,t] <= A[j,t] - R[j,t] ;
127
128 # -----
129 # comenzamos con las restricciones que gestionan los tipos de grano
130 # -----
131
132 # Restriccion 17: Asigno tipo de grano inicial G_jt en t = 0
133 # pero solo si el silo no esta vacio al inicio
134 subject to valorInicialG { j in silos } :
135     A0[j] > 0 ==> G[j,0] = G0[j];
136
137 # variable indicatriz de si Rho es positivo, se usa luego
138 subject to DEF_rho_es_positivo {i in productores , j in silos , t in
139     tiempo} :
140     rho[i,j,t] <= rho_es_positivo[i,j,t]*pmax;
141
142 subject to DEF_rho_es_positivo2 {i in productores , j in silos , t

```

```

142     in tiempo} :
143         rho_es_positivo[i,j,t] <= rho[i,j,t];
144
145 # ----- Restriccion 18a y 18b, GRANOS 1
146 subject to rest18a1 {i in productores, j in silos, t in tiempo: 1
147     <= t and t <= Tmax- T[i,j] } :
148     rho_es_positivo[i,j,t] = 1 ==> G[j, t + T[i,j] -1] <= g[i] +
149     (1-delta_A[j, t + T[i,j] - 1 ])*Gmax;
150
151 subject to rest18a2 {i in productores, j in silos, t in tiempo: 1
152     <= t and t <= Tmax- T[i,j] } :
153     rho_es_positivo[i,j,t] = 1 ==> G[j, t + T[i,j] -1] >= g[i] - (1-
154     delta_A[j, t + T[i,j] - 1 ])*Gmax;
155
156 # caso especial, llega grano en t = 0
157 subject to rest18b1 {i in productores, j in silos } :
158     rho_es_positivo[i,j,0] = 1 and T[i,j] > 0 ==> G[j, T[i,j] -1]
159     <= g[i] + (1-delta_A[j, T[i,j] - 1 ])*Gmax;
160
161 subject to rest18b2 {i in productores, j in silos } :
162     rho_es_positivo[i,j,0] = 1 and T[i,j] > 0 ==> G[j, T[i,j] -1] >=
163     g[i] - (1- delta_A[j, T[i,j] - 1 ]);
164
165 # ----- Restriccion 19, GRANOS 2
166 subject to rest19 {i in productores, j in silos , t in 0..(Tmax - T
167     [i,j])}:
168     rho_es_positivo[i,j,t] = 1 ==> G[j, t + T[i,j] ] = g[i];
169
170 # ----- Restriccion 20, GRANOS 3
171 subject to rest20 { j in silos , t in tiempo: t > 0}:
172     delta_A[j, t-1] = 1 ==> G[j,t] = G[j,t-1];
173
174 # ----- GRANOS 4 : El grano retirado es del tipo del consumidor de
175     destino
176
177 # ----- Antes de modelar la restriccion granos 4, debemos definir
178     la variable indicatriz phi_es_positivo
179 subject to DEF_phi_es_positivo {j in silos , k in consumidores, t
180     in tiempo} :
181     phi[j,k,t] <= phi_es_positivo[j,k,t]*qmax;
182
183 subject to DEF_phi_es_positivo2 {j in silos , k in consumidores, t
184     in tiempo} :
185     phi_es_positivo[j,k,t] <= phi[j,k,t];
186
187 # ----- ahora podemos modelar la restriccion 21 granos 4
188
189 subject to rest21 {j in silos , k in consumidores, t in tiempo}:
190     phi_es_positivo[j,k,t] = 1 ==> G[j,t] = h[k];

```

Código 1: Código fuente del Modelo 3.

A continuación se muestra el código del Modelo 3b, el cual consideramos el modelo final. Las diferencias con respecto al Modelo 3 son:

- en la línea 40: se permiten parámetros tiempos de transporte nulos
- líneas 93 y 96, restricciones 9 y 10: el cálculo de las variables α_j^t y A_j^t cambio
- línea 132, restricción 17: asigna el valor del tipo de grano inicial en los silos
- línea 145, restricción 18: condición “Granos 1”

```

1 param I >= 0;
2 param J > 0;
3 param K >= 0 ;
4 param C > 0;
5
6 set productores := 1 .. I ;
7 set silos := 1 .. J ;
8 set consumidores := 1 .. K;
9 set camiones := 1 .. C;
10
11 param p{productores};
12 param g{productores};
13 param a{productores};
14 param b{productores};
15
16 # pmax se usa para las restricción big-M que modelan a
17   rho_es_positivo
18 param pmax = max{i in productores} p[i];
19
20 param A0{silos} >=0;
21 param G0{silos} >=0;
22 param capSilo{silos } > 0;
23 param tDescargaSilo{silos } > 0;
24
25 param q{consumidores};
26 param h{consumidores};
27 param e{consumidores} ;
28 param l{k in consumidores} > e[k] - 1;
29
30 # qmax se usar con un proposito similar a pmax
31 param qmax = max{k in consumidores} q[k];
32
33
34 param Gmax = max (max{i in productores} g[i] , max{j in silos } G0[
35   j] , max {k in consumidores} h[k] );
36 set granos = 0..Gmax;
37
38 param capCamion{c in camiones} ;
39 param descCamion{c in camiones};
40
41 param T {productores , silos } >= 0;
42 param T2 {silos , consumidores} >= 0;

```

```

42
43 param costs1{productores,silos ,camiones};
44 param costs2{silos ,consumidores,camiones};
45
46
47 param Tmax =max( max{i in productores, j in silos } ( b[i] + T[i,j]
48   ) , max{k in consumidores } l[k] ) ;
49 set tiempo = 0 .. Tmax;
50
51 var x{i in productores, j in silos , t in tiempo,c in camiones}
52   integer >= 0;
53 var rho_es_positivo{i in productores, j in silos , t in tiempo }
54   binary;
55 var rho{i in productores, j in silos , t in tiempo} integer >= 0 ;
56
57 var y{j in silos , k in consumidores ,t in tiempo,c in camiones}
58   integer >= 0;
59 var phi{j in silos , k in consumidores, t in tiempo} integer >= 0 ;
60 var phi_es_positivo{j in silos , k in consumidores, t in tiempo}
61   binary;
62
63 var alfa{j in silos , t in tiempo } >= 0;
64 var A{ j in silos , t in tiempo } >= 0;
65 var r{ j in silos , t in tiempo } >= 0;
66 var R{ j in silos , t in tiempo } >= 0;
67
68 var delta_A{j in silos , t in tiempo} binary;
69 var G{j in silos , t in tiempo} in granos;
70
71 minimize Cost:
72
73 sum{i in productores, j in silos , t in tiempo, c in camiones} (
74   costs1[i,j,c]*x[i,j,t,c] )
75 + sum{ j in silos , k in consumidores,t in tiempo, c in camiones}
76   ( costs2[j,k,c]*y[j,k,t,c] ) ;
77
78 subject to rest2 {i in productores, j in silos , t in tiempo}:
79   t < a[i] or t > b[i] ==> rho[i,j,t] = 0;
80
81 subject to rest3 {i in productores, j in silos , t in tiempo}:
82   sum{c in camiones} (x[i,j,t,c]*capCamion[c]) = rho[i,j,t];
83
84 subject to rest4 {i in productores}:
85   sum{j in silos , t in tiempo} rho[i,j,t] = p[i];
86
87 subject to rest6{j in silos , k in consumidores, t in tiempo, c in
88   camiones}:
89   t + T2[j,k] < e[k] or t + T2[j,k] > l[k] ==> phi[j,k,t] = 0;
90
91 subject to rest7 {j in silos ,k in consumidores, t in tiempo}:
92   sum{c in camiones} (y[j,k,t,c]*capCamion[c]) = phi[j,k,t];
93
94 subject to rest8 {k in consumidores}:
95   sum{j in silos , t in tiempo} phi[j,k,t] = q[k];
96
97 # -----
98 # FUNCIONES DE ACOPIO Y RETIRO

```

```

91 # -----
92
93 subject to rest9 {j in silos , t in tiempo}:
94     alfa[j,t] = sum{i in productores} (if t >= T[i,j] then rho[i,
95         j, t - T[i,j] ] else 0 ) ;
96
97 subject to rest10 {j in silos , t in tiempo} :
98     A[j,t] = A0[j] + sum{ iter in 0 .. t} alfa[j,iter];
99
100 # -----
101 # definicion de r_jt y R_jt , funciones de retiro
102
103 subject to rest11 {j in silos , t in tiempo}:
104     r[j,t] = sum{k in consumidores} phi[j,k,t] ;
105
106 subject to rest12 {j in silos , t in tiempo} :
107     R[j,t] = sum{ iter in 0 .. t} r[j,iter];
108
109
110 # -----
111 # No se supera capacidad de los silos
112
113 subject to rest13 {j in silos , t in tiempo} :
114     0 <= A[j,t] - R[j,t] <= capSilo[j];
115
116
117 # -----
118 # definicion de delta_A, restricciones 15 y 16 implementadas con
119     Conditional constraints
120
121 subject to defDeltaA1 {j in silos , t in tiempo} :
122     #delta_A[j,t] = 0 ==> A[j,t] = R[j,t] ;
123     A[j,t] - R[j,t] <= delta_A[j,t]* capSilo[j] ;
124
125 subject to defDeltaA2 {j in silos , t in tiempo} :
126     #delta_A[j,t] = 1 ==> A[j,t] < R[j,t] ;
127     delta_A[j,t] <= A[j,t] - R[j,t] ;
128
129 # -----
130 # comenzamos con las restricciones que gestionan los tipos de grano
131 # -----
132
133 # Restriccion 17: Asigno tipo de grano inicial G_jt en t = 0
134 # pero solo si el silo no esta vacio al inicio
135 subject to valorInicialG { j in silos } :
136     A0[j] > 0 ==> G[j,0] = G0[j];
137
138 # variable indicatriz de si Rho es positivo, se usa luego
139 subject to DEF_rho_es_positivo {i in productores, j in silos , t in
140     tiempo} :
141     rho[i,j,t] <= rho_es_positivo[i,j,t]*pmax;
142
143 subject to DEF_rho_es_positivo2 {i in productores, j in silos , t
144     in tiempo} :
145     rho_es_positivo[i,j,t] <= rho[i,j,t];

```

```

144
145 # ----- Restriccion 18a y 18b, GRANOS 1
146 subject to rest18a1 {i in productores, j in silos, t in tiempo: 1
147     <= t and t <= Tmax- T[i,j] }:
148     rho_es_positivo[i,j,t] = 1 ==> G[j, t + T[i,j] -1] <= g[i] +
149     (1-delta_A[j, t + T[i,j] - 1 ])*Gmax;
150
151 subject to rest18a2 {i in productores, j in silos, t in tiempo: 1
152     <= t and t <= Tmax- T[i,j] }:
153     rho_es_positivo[i,j,t] = 1 ==> G[j, t + T[i,j] -1] >= g[i] - (1-
154     delta_A[j, t + T[i,j] - 1 ])*Gmax;
155
156 # caso especial, llega grano en t = 0
157 subject to rest18b1 {i in productores, j in silos }:
158     rho_es_positivo[i,j,0] = 1 and T[i,j] > 0 ==> G[j, T[i,j] -1]
159     <= g[i] + (1-delta_A[j, T[i,j] - 1 ])*Gmax;
160
161 subject to rest18b2 {i in productores, j in silos }:
162     rho_es_positivo[i,j,0] = 1 and T[i,j] > 0 ==> G[j, T[i,j] -1] >=
163     g[i] - (1- delta_A[j, T[i,j] - 1 ]);
164
165 # ----- Restriccion 19, GRANOS 2
166 subject to rest19 {i in productores, j in silos , t in 0..(Tmax - T
167     [i,j])}:
168     rho_es_positivo[i,j,t] = 1 ==> G[j, t + T[i,j] ] = g[i];
169
170 # ----- Restriccion 20, GRANOS 3
171 subject to rest20 { j in silos , t in tiempo: t > 0}:
172     delta_A[j, t-1] = 1 ==> G[j,t] = G[j,t-1];
173
174 # ----- GRANOS 4 : El grano retirado es del tipo del consumidor de
175     destino
176
177 # ----- Antes de modelar la restriccion granos 4, debemos definir
178     la variable indicatriz phi_es_positivo
179 subject to DEF_phi_es_positivo {j in silos , k in consumidores, t
180     in tiempo} :
181     phi[j,k,t] <= phi_es_positivo[j,k,t]*qmax;
182
183 subject to DEF_phi_es_positivo2 {j in silos , k in consumidores, t
184     in tiempo} :
185     phi_es_positivo[j,k,t] <= phi[j,k,t];
186
187 # ----- ahora podemos modelar la restriccion 21 granos 4
188
189 subject to rest21 {j in silos , k in consumidores, t in tiempo}:
190     phi_es_positivo[j,k,t] = 1 ==> G[j,t] = h[k];

```

Código 2: Código fuente del Modelo 3b.

Anexo: Complejidad del Problema

Demostraremos aquí que nuestro problema de asignación de granos es NP-duro. Para ello mostraremos que el problema de la 2-partición de conjuntos se puede reducir a una instancia de nuestro problema.

El problema de la 2-partición se define así:

Sean $a_1, a_2, \dots, a_n$ una lista de enteros cuya suma es $2A$. El problema de la 2-partición consiste en separar los n enteros en dos subconjuntos cuya suma sea A (suponiendo que tal partición existe).

Este problema es NP-Completo. Mostraremos ahora que el mismo se puede reducir a nuestro problema de asignación de granos a silos. Para ello definiremos una instancia del modelo 1 que sea equivalente a una instancia del problema de partición.

Sean $a_1, a_2, \dots, a_n$ la lista de enteros a particionar. Supondremos que son todos positivos (en caso contrario se puede sumar una misma constante a cada uno de modo de que sean positivos y obteniendo un problema equivalente.)

La instancia buscada es la siguiente:

- Envíos: por cada entero a_i definiremos un envío de a_i toneladas. Todos los envíos son del mismo tipo de grano y las ventanas de tiempo son todas iguales a $(0, 0)$
- Tiempos de transporte: les asignaremos un valor arbitrario, por ejemplo 1. Por tanto todos los envíos se retirarán en el instante $t = 0$ y se descargarán en los silos en el instante $t=1$.
- Silos: solo hay dos silos, ambos de capacidad A y que se encuentran vacíos al inicio.
- Pedidos: no hay consumidores en esta instancia
- Flota: la flota es homogénea y la capacidad del camión es igual a $\max\{a_i\}_{i=1..n}$. Por tanto, para cada envío se usará un exactamente un camión (n camiones en total)

- Costo de los viajes: todos los viajes productor-planta tiene el mismo costo arbitrario, por ejemplo costo = 1

Para esta instancia del problema del problema de transporte de granos tenemos que todas las soluciones factibles son óptimas. En efecto, dado que ocurrirán n viajes y todos tienen costo 1, la función objetivo siempre valdrá n .

Dado que de cada envío i se transporta en un único viaje, garantizamos que el entero a_i será “puesto por completo” en un único silo. Cada silo se corresponde con uno de los subconjuntos de la partición.

Si la instancia del problema de optimización resulta factible, la solución arrojada no dará una solución del problema de partición.

Y también en el otro sentido: si existe una partición de la lista $a_1, a_2, \dots, a_n$, entonces la misma representa una asignación de los envíos de $a_1, a_2, \dots, a_n$ toneladas a los dos silos. En esa asignación los silos serán llenados a su máxima capacidad.

Anexo: Tarifa real

A continuación se muestra la tarifa usada en los test de performance del Capítulo 6. La misma se presentó en el Cuadro 6.1. Aquí se muestra la tabla con los valores de la tarifa completa.

distancia(km)	precio (US\$ / ton)
1 - 5	7.14
6 - 10	7.64
11 - 15	8.21
20 - 16	8.73
25 - 21	9.30
30 - 26	9.84
35 - 31	10.40
40 - 36	10.91
45 - 41	11.47
50 - 46	11.95
55 - 51	12.49
56 - 60	13.10
65 - 61	13.61
70 - 66	14.16
71 - 75	14.68
76 - 80	15.24
81 - 90	16.38
91 - 100	17.33
101 - 120	18.98
121 - 140	20.80
141 - 160	22.52
161 - 180	24.20
181 - 200	25.95
> 200	0.078 por km
> 300	0.064 por km

Cuadro 2: Tarifa real usada para los test de performance.