PARTICIONAMIENTO DE NODOS DE UN DIGRAFO EN CIRCUITOS: MODELOS Y APLICACION A LA ASIGNACION CONDUCTORES-VIAJES EN UNA EMPRESA DE TRANSPORTE.

Graciela Nasini
Facultad de Ciencias Exactas, Ingeniería y Agrimensura - Universidad Nacional de Rosario
CONICET
Pellegrini 250, (2000) Rosario, Argentina
nasini@fceia.unr.edu.ar

Daniel Severin
Facultad de Ciencias Exactas, Ingeniería y Agrimensura - Universidad Nacional de Rosario
CONICET
Pellegrini 250, (2000) Rosario, Argentina
daniel@fceia.unr.edu.ar

ABSTRACT

En este trabajo, planteamos un problema de particionamiento de vértices de un digrafo en circuitos dirigidos con la particularidad de que cada circuito debe tener exactamente un arco de entre un conjunto de arcos preestablecido, y proponemos dos nuevas formulaciones para resolver este problema. La segunda formulación contiene una familia de restricciones de tamaño exponencial, lo que nos lleva a separarlas durante la optimización mediante un algoritmo de planos de corte.

Luego, experimentamos con ambas formulaciones sobre instancias pequeñas y la comparamos con una formulación ya existente en la literatura, mostrando la superioridad de éstas últimas.

Finalmente, obtenemos mejoras razonables en una instancia real de una empresa de transporte de la Provincia de Buenos Aires, mediante el uso de un algoritmo de mejora con una de las formulaciones presentadas.

KEYWORDS. Particionamiento en digrafos, circuitos dirigidos, asignación conductores-viajes, programación entera.

Main Area: Particionamento em Grafos

1. Introducción

Este trabajo tiene sus orígenes en un problema de asignación conductores-viajes planteado por una empresa de transporte interurbano de la Provincia de Buenos Aires: ésta tiene adjudicados 700 viajes diarios que debe cubrir con una planta de 200 conductores y 95 coches. Cada uno de los viajes tiene definido origen, destino, recorrido y horario de salida. El objetivo es diseñar jornadas laborales para sus conductores que minimicen las horas extras a pagar.

Este tipo de problemas es una de las tareas más significativas en planificación del transporte (ver, por ejemplo, Borndörfer et al. (2010) y Bussieck et al. (1997)) y han dado lugar al desarrollo de sistemas informáticos comerciales muy sofisticados que combinan múltiples herramientas de la Investigación de Operaciones como, por ejemplo, metaheurísticas y resolución de modelos enteros mediante generación de columnas, entre otras (Weider, 2007).

En particular, la empresa mencionada adquirió uno de estos sistemas comerciales para la planificación de jornadas de sus conductores. Sin embargo, este "paquete cerrado" no permite parametrizar la totalidad y variedad de condicionamientos particulares, generalmente derivados de acuerdos laborales particulares con los trabajadores de estas líneas y políticas propias de la empresa. Así, las soluciones provistas por la herramienta no siempre respetan estos convenios y, por otro lado, muchas veces pueden ser mejoradas (en términos de costos) mediante simples intercambios entre viajes de dos o más jornadas. La "mejora" de las soluciones aportadas por la herramienta comercial actualmente se realiza en forma manual por los empleados del área de planificación.

En un primer trabajo (Alvarado et al., 2013) propusimos un modelo de Programación Lineal Entera (PLE) para este problema de optimización que incluía todas y cada una de las restricciones específicas de la empresa y de los convenios laborales. Este modelo se desarrolló a partir de las variables de decisión "naturales": por cada par viaje-conductor se define una variable 0-1 que vale 1 si y sólo si ese par es una asignación. Para modelar ciertas restricciones y la función objetivo fue necesario incorporar algunas variables artificiales. En este modelo, para una instancia de n viajes y k conductores, el tamaño del modelo es del orden de kn variables y kn^2 restricciones.

En este trabajo presentamos una reducción del problema de asignación conductores-viajes a un problema de particionamiento de nodos de un digrafo en k circuitos con cierta propiedad. Esta reducción se presenta en la Sección 2. Luego proponemos dos modelos de PLE para nuestro problema de particionamiento en la Sección 3.

Finalmente, en la seccion 4 reportamos resultados computacionales que manifiestan la importante ventaja de estos modelos respecto al original dado en Alvarado et al. (2013). También abordamos una instancia real de la empresa y, a partir de una solución obtenida con la herramienta que actualmente utiliza la empresa, generamos una solución mejor mediante un algoritmo de mejora basado en una de las nuevas formulaciones presentadas.

2. El Problema de asignación conductores-viajes como problema de particionamiento

A continuación hacemos una breve descripción del problema. Los detalles pueden ser consultados en Alvarado et al. (2013).

Los viajes a cargo de la empresa corresponden a dos líneas entre tres ciudades que denotaremos con 0, 1 y 2. Las dos líneas corresponden a viajes entre las ciudades 1 y 2 y entre las ciudades 0 y 1. Los viajes entre 1 y 2 se realizan por seis rutas diferentes, es decir, esta línea cuenta con seis ramales con distintos recorridos y duraciones. Los viajes entre 0 y 1 se llevan a cabo por tres rutas distintas. En total contamos con 9 recorridos, cada uno de ellos con distintas duraciones.

La empresa cuenta con depósitos para guardar los coches sólo en las ciudades 1 y 2. Por cuestiones asociadas a las tareas de mantenimiento de los coches, la jornada de cada conductor debe comenzar y terminar en el mismo depósito. En caso de que el primer viaje asignado a un conductor tenga su inicio en 0, éste deberá empezar su jornada tomando un coche en alguno de los depósitos de 1 o 2 y realizar un viaje en vacío hasta 0. De igual manera, como al finalizar la jornada los conductores deben dejar el coche en el mismo depósito de donde lo sacaron, también es posible

que deban realizar un viaje en vacío al final de su día laboral. Los viajes en vacío también pueden utilizarse para combinar, por ejemplo, un viaje que termina en 2 con uno que empieza en 0. Sin embargo, por política de la empresa, no se admiten viajes en vacío entre 1 y 2. En resumen, los viajes comerciales se realizan entre las ciudades origen-destino 0-1, 1-0, 1-2 y 2-1, mientras que los posibles viajes en vacío pueden darse entre 0-1, 1-0, 0-2 y 2-0.

Denominamos con K al conjunto de conductores de la planta de la empresa y con V al conjunto de viajes a cubrir por la empresa con dichos conductores. Para cada $v \in V$ contamos con los siguientes datos: ciudad de origen, ciudad de destino, horario de salida (contado en minutos desde la medianoche) y duración en minutos.

Las reglas para determinar si un conductor puede tomar un viaje v luego de un viaje u (en términos de sus orígenes, destinos, horarios de salida y duración) se detallan en Alvarado et al. (2013). En estas reglas se incluyen la totalidad de los acuerdos laborales específicos entre conductores y empresa así como también las políticas propias de la empresa respecto, por ejemplo, a la duración y lugar de descanso de los conductores.

También en Alvarado et al. (2013) se detallan las condiciones que deben cumplir un par de viajes para poder ser primer y último viaje de una jornada laboral. En estas reglas vuelven a reiterarse convenios específicos y requirimientos de la empresa respecto a la duración máxima y mínima de una jornada o las condición de empezar y terminar en el mismo depósito.

Con todos estos datos, una instancia del problema queda definido por:

- Un conjunto K de conductores y un conjunto V de viajes.
- Un conjunto $A \subset V \times V$ donde $(u, v) \in A$ si y sólo si un mismo conductor puede realizar los viajes u y v en forma consecutiva.
- Un conjunto $J \subset V \times V$ donde $(u, v) \in J$ si y sólo si un conductor puede realizar como primer viaje de su jornada a u y como último viaje a v.
- Un vector $c \in \mathbb{R}^J$ de costos donde, para todo $(u, v) \in J$, c_{uv} mide la cantidad de horas extras de una jornada que comienza con el viaje u y termina con v.

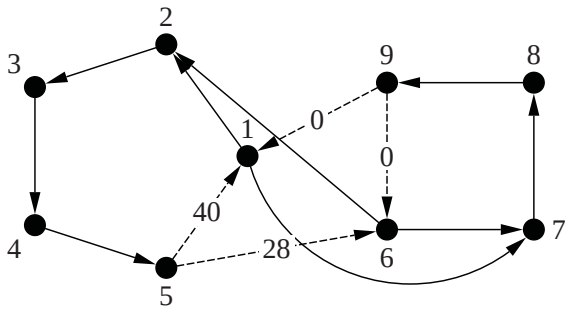
Una asignación de viajes a conductores es un conjunto de $|K|$ jornadas laborales, donde cada jornada es una sucesión de viajes $v_1, \ldots, v_s$ en V tal que $(v_1, v_s) \in J$ y, para todo $i = 1, \ldots, s-1$, $(v_i, v_{i+1}) \in A$. Es importante señalar que, por la naturaleza del problema, el digrafo definido por (V, A) es acíclico.

Definiendo $J^* = \{(v, u) : (u, v) \in J\}$, si consideramos el digrafo $D = (V, A^*)$ donde $A^* = A \cup J^*$ es fácil ver que cada sucesión de viajes asociada a una jornada define un circuito dirigido en D con exactamente un arco de J^* y recíprocamente. Por otro lado, la restricción de que todos los viajes deben ser asignados a alguna jornada nos impone la condicion de "cubrir" todos los vértices con circuitos dirigidos. La condición de que un viaje no puede ser asignado a dos conductores a la vez nos determina que el conjunto de circuitos dirigidos de D con exactamente un arco de J^* debe ser una partición de V.

Si a cada arco (v, u) de J^* le asignamos como costo la cantidad de minutos extras de la jornada que comienza con el viaje u y termina con v, el problema de minimización de horas extras se reduce a encontrar una partición de los vértices de V en circuitos dirigidos que contengan exactamente un arco de J^* que minimice la suma de los costos de los arcos de J^* utilizados.

A modo de ejemplo, consideremos la instancia representada en la siguiente figura. Aquí $K = \{1, 2\}$ y $V = \{1, \ldots, 9\}$. Los arcos pertenecientes a A están marcados con flechas, mientras que los arcos de J^* se denotan con flechas punteadas y, en el medio, su costo asociado (por ejemplo, $(1, 2) \in A$ mientras que $(1, 5) \in J$ y $c_{15} = 40$).

Una posible solución de costo total 40 es $W_1 = \{1, 2, 3, 4, 5\}$, $W_2 = \{6, 7, 8, 9\}$. Sin embargo, una mejor solución de costo total 28 es $W_1 = \{2, 3, 4, 5, 6\}$, $W_2 = \{1, 7, 8, 9\}$.



3. Modelos de PLE para el problema de particionamiento en circuitos

En esta sección proponemos dos modelos de Programación Lineal Entera para nuestro problema de particionamiento.

3.1. Formulación 1

En este primer modelo utilizamos variables binarias que nos informan a cuál de los k circuitos pertenece cada arco de D. Asi, para todo $k \in K$ y $(u, v) \in A^*$, consideramos

$$z_{kuv} = \begin{cases} 1 & \text{si } (u,v) \text{ está en el circuito } k \\ 0 & \text{si no} \end{cases}$$

El modelo de PLE se detalla a continuación:

$$\min \sum_{k \in K} \sum_{(u,v) \in J} c_{uv} z_{kvu}$$

$s.a.$

$$\sum_{k \in K} \sum_{v \in \delta^+(u)} z_{kuv} = 1 \qquad \forall\, u \in V \tag{1}$$

$$\sum_{u \in \delta^-(v)} z_{kuv} = \sum_{w \in \delta^+(v)} z_{kvw} \qquad \forall\, k \in K,\ v \in V \tag{2}$$

$$\sum_{(u,v) \in J} z_{kvu} = 1 \qquad \forall\, k \in K \tag{3}$$

$$z_{kuv} \in \{0, 1\} \qquad \forall\, k \in K, (u,v) \in A^*$$

donde $\delta^+(u) = \{v : (u, v) \in A^*\}$ y $\delta^-(v) = \{u : (u, v) \in A^*\}$.

La restricción (1) asegura que todo vértice de D (viaje) pertenezca a alguno de los circuitos (jornadas). La (2) es la típica restricción de conservación de flujo que asegura que las soluciones factibles definan circuitos. La restricción (3) asegura que cada circuito tenga exactamente un arco de J^*.

Este modelo incluye una gran cantidad de soluciones simétricas que corresponden a una misma solución. Las mismas resultan del hecho de que los *índices* que referencian a cada circuito son indistinguibles. Esto obliga al optimizador a explorar una enorme cantidad de soluciones innecesarias, disminuyendo así su performance.

Por eso, a los efectos de su resolución computacional agregamos las siguientes restricciones que evitan algunas de estas soluciones simétricas:

$$z_{kuv} = 0 \qquad \forall\, 1 \leq u < k \leq |K|,\ v \in \delta^+(u)$$

Esta restricción sólo admite soluciones en las cuales al circuito k sólo se le puede asignar arcos (u, v) tales que $u \geq k$.

Este modelo cuenta con $|K|.(|A| + |J|)$ variables y $|K|.(|V| + 1) + |V|$ restricciones. Es decir, en una instancia de k circuitos y n vértices, tenemos del orden de kn^2 variables y kn restricciones.

3.2. Formulación 2

El segundo modelo surge de la observación de que, en el modelo presentado en la Sección 3.1, contamos con información irrelevante: en la partición en circuitos (asignación de viajes en jornadas) no es importante conocer el número k asignado a cada uno de ellos (qué conductor asignamos a cada jornada). Ahora nos proponemos definir variables que sólo nos informen respecto a si un arco de D está o no en un circuito de la partición.

Por lo cual nos planteamos obtener un modelo basado en las siguientes variables: para todo $(u, v) \in A^*$,

$$z_{uv} = \begin{cases} 1 & \text{si } (u,v) \text{ está en algún circuito} \\ 0 & \text{si no} \end{cases}$$

De esta manera, disminuimos el número de variables de $|K|.(|A| + |J|)$ a $|A| + |J|$, bajando un orden de magnitud. Sin embargo, para modelar la condición de que cada circuito posea exactamente un único arco de J^* deberemos incrementar significativamente el número de restricciones. A tal efecto, definimos el conjunto $\mathcal{P}$ de todos los caminos dirigidos en D tales que sus primer y último arcos pertenecen a J y son distintos. Nos referimos a los elementos de este conjunto como *caminos prohibidos*. Observemos que, si un circuito tiene más de un arco de J, tendremos al menos dos caminos prohibidos. Por eso, nuestro modelo debe incluir restricciones que no permiten soluciones con caminos prohibidos.

El modelo de PLE se detalla a continuación:

$$\min \sum_{(u,v)\in J} c_{uv} z_{vu}$$

$$s.a.$$

$$\sum_{v\in\delta^+(u)} z_{uv} = 1 \qquad \forall\, u \in V \qquad (4)$$

$$\sum_{u\in\delta^-(v)} z_{uv} = \sum_{w\in\delta^+(v)} z_{vw} \qquad \forall\, v \in V \qquad (5)$$

$$\sum_{(u,v)\in J} z_{vu} = |K| \qquad (6)$$

$$\sum_{e\in P} z_e \leq |P| - 1 \qquad \forall\, P \in \mathcal{P} \qquad (7)$$

$$z_{uv} \in \{0,1\} \qquad \forall\, (u,v) \in A^*$$

Las restricciones (4)-(6) tienen un comportamiento análogo a las (1)-(3) de la primera formulación, mientras que la restricción (7) es la que evita que en una solución incluyamos todos los arcos de algún camino prohibido.

En una instancia de k circuitos y n vértices, disminuimos la cantidad de variables de un orden de kn^2 a n^2 respecto a la formulación anterior y eliminamos completamente la simetría existente al asignar los índices de conductores a viajes. Sin embargo, el número de restricciones (7) puede ser exponencial.

Afortunadamente, el problema de separación asociado a las desigualdades (7) es polinomial ya que, en el peor de los casos, puede reducirse a resolver un número polinomial de problemas de camino dirigido más corto, como veremos a continuación.

3.3. Problema de separación de restricciones asociadas a caminos prohibidos

Dada una solución $z^* \in [0,1]^{A^*}$ que satisface las restricciones (4)-(6) queremos determinar si existe algún camino prohibido cuya restricción asociada está siendo violada por z^*. Este problema puede ser resuelto en tiempo polimomial de la siguiente manera. Para todo arco $(u, v) \in J$

tal que $z^*_{uv} > 0$, se ejecuta uno de los siguientes procedimientos (dependiendo de si la solución es entera o fraccionaria):

1. *Caso* $z^* \in \mathbb{Z}$ (en particular, $z^*_{uv} = 1$).
   - Inicialización. Sea $P \leftarrow \{(u, v)\}$ y $s \leftarrow v$.
   - Iteración. Sea $t \in \delta^+(s)$ tal que $z^*_{st} = 1$. Si $(s, t) \notin J^*$ entonces $P \leftarrow P \cup \{(s, t)\}$, $s \leftarrow t$, y continuamos iterando. Caso contrario (es decir, $(s, t) \in J^*$), verificamos si $(u, v) = (s, t)$. En tal caso, no hay ninguna restricción violada que contenga a (u, v), y finalizamos. Si $(u, v) \neq (s, t)$ entonces la restricción asociada a $P \cup \{(s, t)\}$, i.e. $\sum_{e \in P \cup \{(s,t)\}} z^*_e \leq |P|$, está siendo violada. La agregamos como *lazy constraint* y finalizamos.

2. *Caso* $z^* \notin \mathbb{Z}$.
   - Consideremos la siguiente instancia del problema de camino dirigido más corto: el digrafo D es el mismo que el utilizado en la formulación, el costo de cada arco $e \in A^*$ es $1 - z^*_e$ y el arco inicial es (u, v).
   - Resolvemos la instancia utilizando un algoritmo de Programación Dinámica que consiste en una modificación del Algoritmo de Dijkstra en donde, cada vez que alcanzamos un arco $(s, t) \in J^*$ lo almacenamos en una lista $\tilde{J}^*$ y, además, no evaluamos los vértices de $\delta^+(t)$ (es decir, no se marcan como alcanzados desde t).
   - Para cada $(s, t) \in \tilde{J}^*$ tal que $z^*_{uv} + z^*_{st} > 1$, verificamos si la restricción $\sum_{e \in P} z_e \leq |P| - 1$ es violada, donde P es el camino que comienza en (u, v) y termina en (s, t). En caso de ser violada, se agrega como *corte*.

4. Experimentos computacionales

Esta sección está destinada a evaluar y comparar las distintas formulaciones. Nos referiremos como M_0 a la propuesta en Alvarado et al. (2013) y M_1 a la enunciada en la Sección 3.1.

Para evaluar la segunda formulación implementamos un algoritmo que utiliza la librería Concert C++ de IBM ILOG CPLEX 12.6, la cual permite personalizar las funciones de dicho optimizador en determinados casos. Específicamente en nuestro caso, se le proporciona a CPLEX una formulación entera que consiste en las restricciones (4)-(6), se procede a resolverla y se activa una característica para que, cada vez que CPLEX encuentre una solución z^*, se ejecute el Procedimiento 1 en caso de que la solución sea entera y el Procedimiento 2 en caso de que sea fraccionaria.

Nos proponemos evaluar 2 versiones. Una, que llamamos M_2, sólo utiliza el Procedimiento 1; es decir, sólo se podan nodos correspondientes a soluciones enteras y se agregan las correspondientes *lazy constraints*. La otra, que llamamos M'_2, utiliza ambos procedimientos pero el Procedimiento 2 se ejecuta con la siguiente frecuencia: hasta 40 veces en el nodo raíz, hasta 2 veces en nodos del árbol de enumeración con profundidad de 1 a 4, hasta 1 vez en nodos de profundidad 5 a 10. En profundidades de 11 a 20 se ejecuta 1 vez cada 100 nodos mientras que para profundidades de 21 en adelante no se ejecuta.

El primer experimento consistió en generar instancias de distintos tamaños, y optimizar los modelos correspondientes con CPLEX sobre una computadora Intel i5 2.67Ghz con sistema operativo Linux.

Con M_0, una instancia donde hay que partir un digrafo de 40 vértices en 10 circuitos se resolvió en 2700 segundos, mientras que otra instancia de 44 vértices y 11 circuitos no pudo ser resulta en el término de 2 horas. Estas instancias se resolvieron rapidamente con M_1, M_2 y M'_2. Es más, una instancia de 156 vértices y 40 circuitos se resolvió en 1055 seg. con M_1. La misma instancia se resolvió en 334 seg. con M_2 y 104 seg. con M'_2. Y, finalmente, una instancia

de 160 vértices y 41 circuitos que es resuelta por M_1 en 6741 seg., también lo es por M_2 en 1402 seg. y M_2' en 541 seg. Concluimos que, en términos generales, M_1 permite resolver instancias 4 veces mayores que M_0. Además, la performance de la formulación M_1 es razonable si sólo se desea resolver instancias sin realizar ninguna implementación específica del problema, pero para obtener la mejor performance conviene utilizar la segunda formulación y agregar (7) tanto como restricciones del tipo *lazy* como restricciones del tipo corte.

4.1. Aplicación al problema de la empresa de transporte

Nos planteamos un segundo experimento, en donde abordamos la resolución de una instancia real de 191 conductores y 715 viajes comerciales, más 103 viajes artificiales que se incorporan a fin de poder modelar ciertas restricciones relacionadas con políticas de la empresa. Esto implica resolver el problema de partir un digrafo de 818 vértices en 191 circuitos.

Dado el gran tamaño de la instancia, no es posible resolverla por optimalidad. Por ejemplo, el modelo M_0 tiene 171000 variables y 8 millones de restricciones, mientras que el modelo M_1 tiene 7 millones de variables y 153000 restricciones. Incluso, el modelo M_2' no ha podido resolver instancias de más de 180 vértices.

Sin embargo, debido a la buena performance de las formulaciones en pequeñas instancias y dado que la empresa nos facilitó una asignación de conductores a viajes de dicha instancia generada por su herramienta, nos proponemos mejorar dicha solución (en forma similar a cómo realizan estas "mejoras" los empleados del área de planificación).

Como dijimos en la introdución, la asignación proporcionada por la herramienta de la empresa no siempre es óptima y, a la vez, puede contener jornadas infactibles. Ambos casos se presentaron en la asignación brindada.

Básicamente, nuestro algoritmo de mejora considera una instancia correspondiente a un subconjunto de n conductores y sus viajes asociados, y la resuelve con una de las formulaciones mencionadas. En caso de que la suma de las horas extras de las jornadas obtenidas sea inferior a la de la actual asignación, se actualizan las jornadas involucradas en dicha mejora. También se considera una mejora si una o más jornadas eran infactibles y, luego de la resolución del modelo, se obtiene una asignación factible. La cantidad de modelos resueltos está vinculada a la cantidad de combinaciones de tomar (ciertos) subconjuntos de n elementos del conjunto K. Para mayores detalles, véase Alvarado et al. (2013).

La siguiente tabla muestra el desempeño de este algoritmo al considerar $n \in \{2, 3\}$:

n	Cant. de modelos resueltos	Jornadas infactibles (antes)	Jornadas infactibles (después)	Cant. horas extras (antes)	Cant. horas extras (después)	Tiempo en segundos M_0	Tiempo en segundos M_1
2	14500	65	57	3956	3872	45	39
3	1250000	65	55	3956	3815	5256	2050

En la tabla se muestran la cantidad total de modelos de n conductores resueltos, la cantidad de jornadas infactibles y horas extras de la solución inicial y la obtenida luego de aplicar el algoritmo de mejora, y el tiempo total que se necesitan para resolver todos los modelos según si se utiliza M_0 o M_1.

Con respecto a la diferencia entre $n = 2$ y $n = 3$, éstos reducen la cantidad de conductores infactibles en un 12 % y 15 % respectivamente, y la cant. de horas extras en un 2.1 % y 3.6 %. Con respecto a los modelos, la diferencia en tiempo se hace significativa para $n = 3$, siendo M_1 un 156 % más rápido que M_0. No se evaluó M_2 ni M_2' dado que no se manifiesta diferencia en tiempo para modelos pequeños.

Agradecimientos

Agradecemos la colaboración de María Eugenia Alvarado y Federico Bertero en la discusión e implementación de varios aspectos de este trabajo. Este trabajo fue subsidiado con los proyectos PID CONICET 11220120100277 y PICT-2013-0586.

Referencias

Alvarado M. E., Nasini G., Severin D. (2013), Algoritmos de mejora para el Problema de Asignación Conductores-Viajes en una empresa de transporte. *X Congreso del Instituto Chileno de Investigación Operativa ICHIO (OPTIMA)*. Concepción, Chile.

Borndörfer R., Grötschel M., Jäger U. (2010), Planning Problems in Public Transport. *Production Factor Mathematics*, Part 2.

Bussieck M. R., Winter T., Zimmermann U. T. (1997), Discrete optimization in public rail transport. *Mathematical Programming* 79, 415–444.

Weider, S. (2007), Integration of Vehicle and Duty Scheduling in Public Transport. *PhD Thesis*.