Minimum Cost Hyperassignments

Master's Thesis

Olga Heismann

TU Berlin

October 18, 2010

Eidesstattliche Versicherung

Die selbstständige und eigenhändige Anfertigung versichert an Eides statt

Berlin, den _____ Unterschrift: _____

Zusammenfassung in deutscher Sprache

Die Zugumlaufplanung ist ein fundamentales Problem im Schienenpersonenverkehr. Ihre Aufgabe ist die Zuordnung von Fahrzeugen zu Fahrten eines wöchentlichen Fahrplans. Dabei ist die sogenannte Gleichförmigkeit zu berücksichtigen. Diese Aufgabe kann als Hyperzuordnungsproblem formuliert werden. Bei Hyperzuordnungen handelt es sich um eine neuartige Verallgemeinerung von Zuordnungen.

Zuordnungen sind wie folgt definiert. Sei $k \in \mathbb{N}$. Gegeben ist eine Menge A von Paaren $(\{i\}, \{j\})$, $i, j \in \{1, \dots, k\}$ und Kosten für jedes dieser Paare. Eine Zuordnung ist eine Teilmenge von A, sodass $1, \dots, k$ jeweils genau ein Mal im ersten und zweiten Element eines Paars der Teilmenge vorkommen.

Wir verallgemeinern dieses Konzept wie folgt. Den Ausgangspunkt bilden Paare von Teilmengen von $\{1, \dots, k\}$ mit möglicherweise mehr als einem Element. Dann ist A eine Menge von Paaren $(\{i_1, \dots, i_m\}, \{j_1, \dots, j_n\})$, $m, n \in \mathbb{N}$. Wieder kann man Teilmengen von A betrachten, sodass $1, \dots, k$ jeweils genau ein Mal im ersten und zweiten Element eines Paars der Teilmenge vorkommen. So eine Teilmenge nennen wir Hyperzuordnung.

In der Anwendung, in unserem Fall die ICE/IC-Zugumlaufplanung, entsprechen die Paare aufeinanderfolgenden Zugfahrten, die Teilmengen Verkehrstagen, an denen die Fahrten gültig sind. Die vorliegende Arbeit entwickelt eine mathematische Theorie der Hyperzuordnungen zur Berechnung gleichförmiger Zugumläufe.

Die Arbeit untersucht dazu Hyperzuordnungen im Allgemeinen und für spezielle Problemklassen, die durch die praktische Anwendung motiviert sind. Wir zeigen, dass das Hyperzuordnungsproblem – das Finden einer Hyperzuordnung mit minimalen Kosten – $\mathcal{NP}$-schwer ist und dass eine direkte Formulierung als ganzzahliges lineares Programm (ILP) zu großen Abständen zwischen dem ganzzahligen Optimum und dem Optimum der LP-Relaxierung führt. Wir stellen deshalb eine alternative Formulierung als ganzzahliges lineares Programm in Form einer sogenannten Extended Formulation vor, die den Abstand zwischen LP- und ILP-Optimalwert beweisbar verkleinert und u. a. alle Cliquenungleichungen impliziert. Diese Formulierung lässt sich mit Hilfe eines Spaltenerzeugungsverfahrens lösen. Wir untersuchen das zugehörige Pricing-Problem und entwickeln für die aus Anwendungssicht relevanten Fälle schnelle Algorithmen.

Preface

Vehicle rotation planning for long distance passenger railways is a fundamental problem in rail transport. It deals with the allocation of vehicles to trips in a cyclic weekly schedule. Its result is an assignment of each trip to a follow-on trip which will be serviced by the same vehicle.

Since April 2009, vehicle rotation planning for IC and ICE trains is investigated in a project at the Zuse Institute Berlin in cooperation with DB Fernverkehr AG. I work in this project as a student assistant. Especially I am concerned with the so-called regularity in vehicle rotation planning, which is an important quality criterion: The more the operation of vehicles on different weekdays resembles each other, the easier is personnel planning and adjustment of the plan if unforeseen events occur.

To take regularity into account, vehicle rotation planning can be modeled as a hyperassignment problem (HAP), which asks for the minimum cost hyperassignment. We propose hyperassignments as a novel concept in combinatorial optimization. We study hyperassignments in general and for special classes of hypergraphs that are motivated by our ICE/IC application. This thesis focuses on theory which helps to solve the practical application.

Hyperassignments are generalizations of assignments. Suppose $k \in \mathbb{N}$. The input of the assignment problem is a set A of pairs $(\{i\}, \{j\})$, $i, j \in \{1, \dots, k\}$, and costs for each of these pairs. An assignment is a subset of A such that $1, \dots, k$ appear exactly once in the first element and exactly once in the second element of a pair in the subset. The assignment problem, which consists of finding an assignment with minimum cost, is an important and well-known problem in combinatorial optimization and can be solved in polynomial time.

We can generalize A to be a set of pairs of subsets of $\{1, \dots, k\}$ with possibly more than one element. Then, A is a set of pairs $(\{i_1, \dots, i_m\}, \{j_1, \dots, j_n\})$, $m, n \in \mathbb{N}$. Again, we can look at subsets of A such that $1, \dots, k$ appear exactly once in the first element and exactly once in the second element of a pair in the subset. We call such a subset a hyperassignment.

It turns out that HAP is $\mathcal{NP}$-hard for the practically relevant cases and that

the canonical integer linear programming (ILP) formulation results in large integrality gaps. The main contribution of this thesis is an extended ILP formulation, which provably reduces the integrality gap and implies important classes of inequalities, e. g., all clique inequalities. The extended formulation can be solved by column generation. We propose fast algorithms for the pricing subproblem.

We assume that the reader is familiar with the standard terminology and methodology of graph theory, complexity theory and linear and integer linear programming as presented, for example, in [GLS88] and [Chv83], the lecture notes [Grö09] and [Grö10] in German.

This thesis is structured as follows.

Chapter 1 introduces the terminology needed in this thesis. We introduce directed hypergraphs and discuss some practical problems which have been previously modeled using these. We also identify special classes of directed hypergraphs, which arise in ICE/IC vehicle rotation planning, and some associated concepts.

HAP and its application to ICE/IC vehicle rotation planning is explained in chapter 2. Also all integer linear programs used throughout this thesis are introduced there.

Chapter 3 is about the complexity of HAP. We prove that HAP is $\mathcal{NP}$-hard, even if one imposes several restrictions, which hold for applications to vehicle rotation planning. We also show that HAP has many connections to the $\mathcal{NP}$-hard set partitioning problem, that the integrality gap in the canonical integer linear programming formulation can be arbitrarily large, and that determinants of basis matrices in this formulation can be arbitrarily large.

Chapter 4 starts with computational results using the canonical ILP. They imply that finding maximal cliques in the so called conflict graph is very important. In this chapter and the next we deal only with partitioned hypergraphs. These are directed hypergraphs of a special type. All directed hypergraphs arising from vehicle rotation planning are partitioned. We show that cliques can appear only in special subgraphs of the conflict graph and analyze these subgraphs. We then introduce a different extended ILP formulation of HAP which implies all clique inequalities and also separates other fractional solutions of the LP relaxation of the canonical formulation. This extended formulation has a large but still polynomial number of variables and can be solved by column generation.

We proceed in chapter 5 with algorithms for pricing the variables in the extended formulation. We present two different ideas, that lead to fast algorithms

for practically important situations.

In the last chapter we summarize our theoretical results and give practical conclusions.

Before ending this preface I would like to thank all those people without whom this thesis would not have been possible. In particular, this is Prof. Dr. Dr. h.c. mult. Martin Grötschel, whose very interesting lectures sparked my interest in combinatorial optimization and who gave me the possibility to develop this thesis at Zuse Institute Berlin. I am grateful to Dr. habil. Ralf Borndörfer for his many ideas and his support. Further, I would like to thank Markus Reuther, especially for his comments on the practical application.

Contents

Chapter 1

Hypergraph Terminology

Summary of this chapter *We begin with basic notions about directed hypergraphs. We present some applications from the literature to demonstrate their potential to model practical problems. Afterwards, we introduce special classes of directed hypergraphs, which arise in vehicle rotation planning, and concepts about these types of hypergraphs. These classes have not been considered before. In the end, we define some further terms needed in the following.*

1.1 Basic Directed Hypergraph Terminology

Definition 1.1.1 (directed hypergraph, directed graph)**.** A *directed hypergraph* D is a pair (V,A) consisting of a *vertex* set V and a set $A \subseteq 2^V \times 2^V$ of *hyperarcs* (T_a, H_a) such that $T_a, H_a \neq \emptyset$. We call T_a the *tail* of the hyperarc $a \in A$ and H_a the *head* of a.

A hyperarc a is called an *arc* if $|H_a| = |T_a| = 1$.

A directed hypergraph is a *directed graph* if all its hyperarcs are arcs. In this case, we also write arcs $(\{v\},\{w\})$ simply as (v,w).

Contrary to usual assumptions, we do not require that the tail and the head of a hyperarc have to be disjoint. Many constructions would be more complicated but results would not change. Hyperarcs whose tail and head are not disjoint can be transformed by adding vertices. Moreover, such hyperarcs may occur in the directed hypergraph from our practical application.

Definition 1.1.2 (outgoing and ingoing hyperarcs)**.** Let $D = (V,A)$ be a directed hypergraph. For $W \subseteq V$, $B \subseteq A$ we define

$$\delta_B^{\text{out}}(W) := \{a \in B : T_a \cap W \neq \emptyset\}$$

to be the *outgoing* and

$$\delta_B^{\text{in}}(W) := \{a \in B : H_a \cap W \neq \emptyset\}$$

to be the *ingoing hyperarcs* of W.

For $\delta_B^{\text{out}}(\{v\})$ and $\delta_B^{\text{in}}(\{v\})$ we simply write $\delta_B^{\text{out}}(v)$ and $\delta_B^{\text{in}}(v)$, respectively.

If no set B is given in the index of δ^{out} or δ^{in}, B is assumed to be the full hyperarc set of the hypergraph in question.

A way of representing directed hypergraphs is illustrated in figure 1.1.

To describe applications of directed hypergraphs we need the following

Definition 1.1.3 (cost function)**.** Given a set S, a *cost function* is a function $c_S : S \to \mathbb{R}$.

For $T \subseteq S$ we define

$$c_S(T) := \sum_{s \in T} c_S(s).$$

1.2 Applications of Directed Hypergraphs

Directed hypergraphs can be used to model relations between sets of objects. In this section, we give a very short overview of different application fields. For detailed explanations see the referenced literature.

In traffic assignment problems, directed hypergraphs can be used to model situations where users must take into account that a desirable path may become unavailable ([MN98]). Hyperarcs represent sets of alternatives to travel from or to a given vertex and the values of the cost function on the hyperarcs represent probabilities of their availability.

Different applications for directed hypergraphs can be found in the field of production and manufacturing systems ([GS98]). Linear production systems can be modeled as directed hypergraphs where the vertices represent goods and their outgoing and ingoing hyperarcs represent certain activities, for which the goods are part of the input or output, respectively. In assembly problems, hyperarcs can describe which items can be constructed from which item combinations. Creating a directed hypergraph with cutting patterns as its vertex set can help in cutting stock problems.

Directed hypergraphs were used to model metadata about resources in the World Wide Web in the Resource Description Framework ([MS07],[WLW08]), which employs triples of a subject, a predicate and an object. In the directed hypergraph model, the tail of each hyperarc consists of a subject and a predicate, and the head is an object.

In data mining, directed hypergraphs are used for local pruning of association rules ([CDP04]).

Also, directed hypergraphs were used for the verification of rule-based expert systems ([RSC97]), where the vertices are used to represent compound clauses.

Further applications include relational databases ([ADS84]), formal languages ([GS98]) and the modeling of biological pathways ([KNO$^+$03]).

1.3 Special Classes of Directed Hypergraphs

Since the concept of directed hypergraphs is very general, it is unlikely to find good solution methods or specific theory for problems on these. Therefore, it can be helpful to try to find a special structure of directed hypergraphs resulting from applications and to investigate the problem for this type of hypergraphs.

For directed hypergraphs which result from vehicle rotation planning we found two special characteristics. Here we introduce two classes of directed hypergraphs, which reflect these characteristics. Particularly the idea of partitioned hypergraphs leads to an extended formulation with interesting and practically useful properties.

For partitioned hypergraphs we employ that regularity conditions in ICE/IC vehicle rotation planning can always be described using trips such that there is no subset of more than two trips which differ pairwise by more than the weekday on which they begin.

Definition 1.3.1 (partitioned hypergraph, configuration)**.** Let $d \in \mathbb{N}$ be some constant number. A directed hypergraph $D = (V,A)$ is called *partitioned hypergraph*, if there exists $w \in \mathbb{N}$ and *parts* $P_1, P_2, \ldots, P_w \subseteq V$ with $P_i \neq \emptyset$, $|P_i| \leq d$ for all $i \in \{1, \ldots, w\}$ such that

1. $V = P_1 \dot\cup P_2 \dot\cup \ldots \dot\cup P_w$ and
2. for every $a \in A$ there exist $t(a), h(a) \in \{1, \ldots, w\}$ such that $T_a \subseteq P_{t(a)}$, $H_a \subseteq P_{h(a)}$.

In this case, d is called the *maximum part size* and for every $i \in \{1, \ldots, w\}$ we define the set of all *outgoing* and *ingoing configurations* of part P_i to be

$$\mathscr{C}_i^{\text{out}} = \left\{ C \subseteq A : H_a \cap H_b = \emptyset \ \forall a, b \in C \text{ with } a \neq b, \dot{\bigcup_{a \in C}} T_a = P_i \right\}$$

and

$$\mathscr{C}_i^{\text{in}} = \left\{ C \subseteq A : T_a \cap T_b = \emptyset \ \forall a, b \in C \text{ with } a \neq b, \bigcup_{a \in C}^{\cdot} H_a = P_i \right\},$$

respectively.

Finally, we write $\mathscr{C}^{\text{dir}}$ for $\bigcup_{i=1}^{w} \mathscr{C}_i^{\text{dir}}$ for $\text{dir} \in \{\text{out}, \text{in}\}$.

For every part the tail of every of its outgoing and the head of every of its ingoing hyperarcs is a subset of the part.

See figures 1.1 and 1.2 for a visualization of a partitioned hypergraph. The latter also shows configurations.

For the second class, the graph based hypergraphs, we use that vehicle rotation planning without regularity conditions can be described on a directed graph. Regularity, which can be described by hyperarcs, as we will see later, can be formulated using sets of arcs of the directed graph.

Graph based hypergraphs have a strong connection to directed graphs. The hyperassignment problem can be efficiently solved on directed graphs since it is just an assignment problem in this case. We can try to use this when investigating HAP on graph based hypergraphs.

Definition 1.3.2 (graph based hypergraph)**.** Let $D = (V, A)$ be a directed hypergraph define

$$B := \{a \in A : a \text{ is an arc}\}.$$

D is called **graph based hypergraph** if for each hyperarc $a \in A$ there exists a set $B(a) \subseteq B$ of arcs with pairwise disjoint tails and pairwise disjoint heads such that

$$T_a = \bigcup_{b \in B(a)}^{\cdot} T_b$$

and

$$H_a = \bigcup_{b \in B(a)}^{\cdot} H_b.$$

In this case, we call the directed graph (V, B) the base graph of D.

The directed hypergraph in figure 1.1 is not graph based whereas figure 1.2 shows a graph based hypergraph.

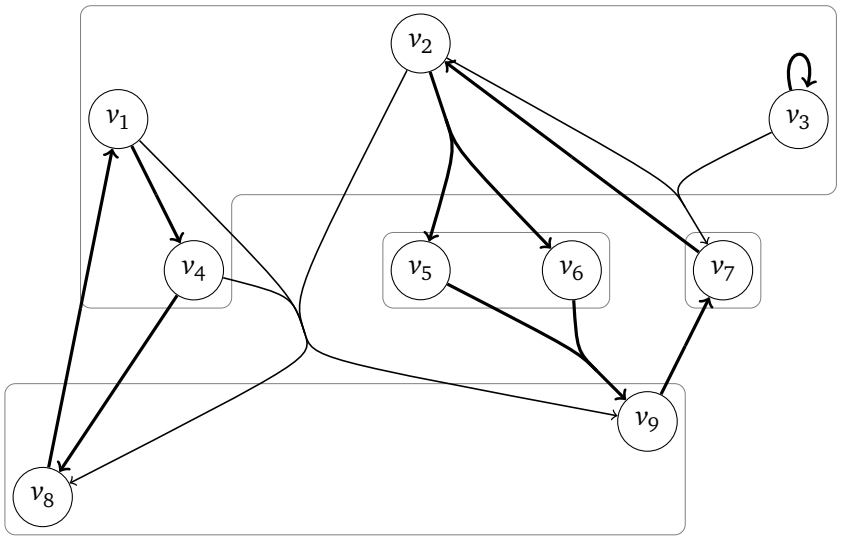


Figure 1.1: Directed hypergraph $D = (V,A)$ with vertices $V = \{v_1, \dots, v_9\}$ and hyperarcs indicated by the connections with arrows pointing into the direction of the head vertices of the hyperarc. v_1 has two outgoing hyperarcs $(\{v_1\}, \{v_4\})$ and $a = (\{v_1, v_2, v_4\}, \{v_8, v_9\})$, and one ingoing hyperarc $(\{v_8\}, \{v_1\})$. For $d \geq 4$, D is partitioned. A possible selection of parts is indicated by the gray boxes. Because of hyperarcs a and $(\{v_2, v_3\}, \{v_7\})$, D is not partitioned for $d \leq 3$. The thicker hyperarcs form a hyperassignment in D.

1.4 Further Notions

We introduce the notion of hyperassignments. They are the object of our study. The idea is to generalize assignments, which are formed by pairs of one element sets, to pairs of sets which can also contain more than more element. A formal definition is as follows.

Definition 1.4.1 (circulation, hyperassignment)**.** Let $D = (V,A)$ be a directed hypergraph. $Z \subseteq A$ is called *circulation* in D if for every $v \in V$

$$|\delta_Z^{\text{out}}(v)| = |\delta_Z^{\text{in}}(v)|$$

for all $v \in V$.

A circulation $H \subseteq A$ in D called *hyperassignment* if for each $v \in V$

$$|\delta_H^{\text{out}}(v)| = 1.$$

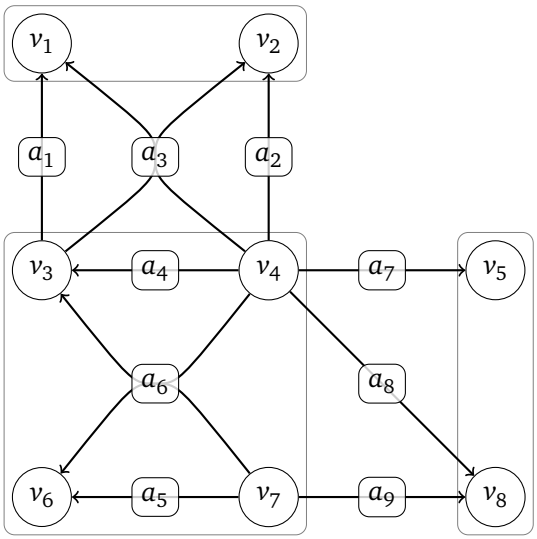


Figure 1.2: $D = (V,A)$ with $V = \{v_1, \dots v_8\}$, $A = \{a_1, \dots a_9\}$ is a graph based hypergraph. For $d = 4$, it is also partitioned with the parts $P_1 = \{v_1, v_2\}$, $P_2 = \{v_3, v_4, v_6, v_7\}$ and $P_3 = \{v_5, v_8\}$. P_1 has two ingoing configurations, $\{a_1, a_2\}$ and $\{a_3\}$. $\{a_7, a_8\}$ is not an ingoing configuration of P_3, because the tails of a_7 and a_8 intersect. The set of outgoing configurations of P_2 is empty, because v_6 has no outgoing hyperarcs.

A hyperassignment assigns to each vertex some set of following vertices given by the head of its unique outgoing hyperarc. The same set is assigned to all the other vertices in the tail of this hyperarc. By the definition of a hyperassignment, each vertex also has a unique ingoing hyperarc. This implies that each vertex has exactly one predecessor set.

Usually, assignments are considered on two sets with equal cardinality. In our case both sets are taken to be the vertex set of the hypergraph, which is no loss of generality.

In figure 1.1 an example of a hyperassignment in some directed hypergraph is given. In figure 1.3 the vertices of the directed hypergraph are doubled such that this hyperassignment can be viewed as assigning elements from one set to elements from another set.

For graph based hypergraphs, the hyperassignment can be broken down to an assignment as follows: Assign to each vertex the vertex in the head of the arc with this vertex in its tail in the set of arcs forming its outgoing hyperarc.

Definition 1.4.2 (undirected hypergraph, undirected graph)**.** A pair $U = (N, E)$

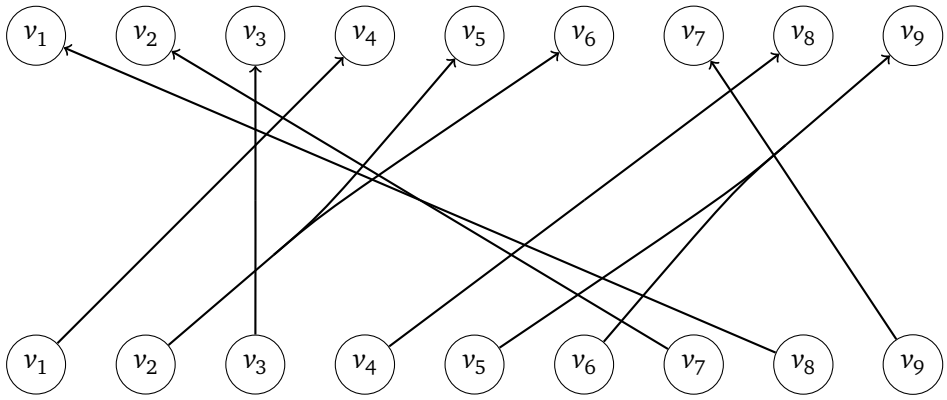


Figure 1.3: After doubling the vertices of the directed hypergraph from figure 1.1 the hyperassignment shown there can be viewed as assigning elements from one set to elements from another set.

of a finite *node* set N and a *hyperedge* set $E \subseteq 2^N$ of nonempty subsets of N is called *undirected hypergraph*. If $|e| = 2$ for all $e \in E$, the undirected hypergraph is called *undirected graph* with *edge* set E.

Definition 1.4.3 (partitioning)**.** A *partitioning* of an undirected hypergraph $U = (N, E)$ is a subset K of E such that each element of N occurs in exactly one member of K.

An example of an undirected hypergraph is $U = (N, E)$ with $N = \{1, \ldots, 4\}$ and $E = \big\{\{1,2\},\{1,3\},\{1,2,4\},\{2\},\{3,4\}\big\}$.

$\big\{\{1,2\},\{3,4\}\big\} \subseteq E$ is a partitioning of U. $\big\{\{1,3\},\{2\}\big\} \subseteq E$ is not a partitioning of U because $4 \in N$ does not appear in any subset. $\big\{\{1,2,4\},\{3,4\}\big\} \subseteq E$ is not a partitioning of U because 4 is contained in two sets.

Definition 1.4.4 (complement, subgraph, induced subgraph)**.** The *complement* of an undirected graph $U = (N, E)$ is the undirected graph with node set N and an edge set consisting of all the two element subsets of N without those in E.

An undirected graph (O, F) is called a *subgraph* of U if $O \subseteq N$, $F \subseteq E$. Further, the subgraph is an *induced* subgraph if F is maximal for O.

Definition 1.4.5 (conflict graph)**.** Let $D = (V, A)$ be a directed hypergraph. The *conflict graph* of D is an undirected graph with one node for each $a \in A$ and such that an edge $\{a, b\}$ connects $a, b \in A$ if and only if there exists a vertex $v \in V$ such that $v \in T_a \cap T_b$ or $v \in H_a \cap H_b$.

Two hyperarcs $a, b \in A$ which are connected by an edge in the conflict graph can never both appear in a hyperassignment in D.

Definition 1.4.6 (clique)**.** Let $U = (N, E)$ be an undirected graph. A *clique* in U is a set $Q \subseteq N$ of nodes of U such that for every two nodes $m, n \in Q$ there is an edge $\{m, n\} \in E$ connecting them. A clique in U is called *maximal clique* if there is no clique in U containing this clique and other nodes.

Figure 1.4 shows a conflict graph.

Definition 1.4.7 (clique graph)**.** An induced subgraph of the conflict graph of a partitioned hypergraph $D = (V, A)$ is called outgoing (ingoing) *clique graph* of D, if there is a fixed part such that the vertex set of the subgraph consists of all hyperarcs all whose tails (heads) are subsets of the part.

Definition 1.4.8 (hole, antihole)**.** A *hole* of size $s \geq 5$ is an undirected graph $U = (N, E)$ with $N = \{n_0, n_1, \ldots, n_{s-1}\}$ and

$$E = \left\{\{n_i, n_{(i+1) \mod s}\} : i \in \{0, 1, \ldots, s-1\}\right\}.$$

An *antihole* of size $s \geq 5$ is the complement of a hole of size s.
A hole or antihole is called odd if s is odd.

Definition 1.4.9 (perfect graph)**.** An undirected graph is called *perfect* if it does not contain an odd hole or antihole as an induced subgraph.

This is not the usual definition of a perfect graph, but the strong perfect graph theorem (see [CRST06]) implies that it is equivalent to the usual definition. The definition stated here is better applicable for our purpose.

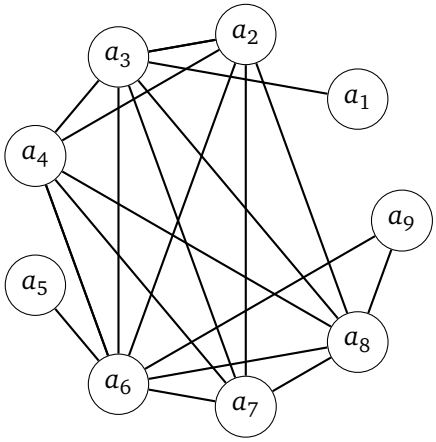


Figure 1.4: The conflict graph of the directed hypergraph from figure 1.2. a_2, a_3, a_4, a_7, a_8 form a clique in the conflict graph, they are all outgoing hyperarcs of v_4. a_6, a_8, a_9 also form a clique, but there is no vertex in the hypergraph such that they all would be contained in its set of out- or ingoing hyperarcs.

Chapter 2

HAP—The Hyperassignment Problem

Summary of this chapter *First of all, we introduce the hyperassignment problem (*HAP*) and prove the correctness of a canonical ILP formulation. We show how* HAP *can be seen as a minimum cost flow problem. Then, we present vehicle rotation planning as our practical application for* HAP*. Afterwards, we state* SPP*, the set partitioning problem, which will be shown to have many connections to* HAP*. In the end, we give an overview of further ILPs for* HAP*, which will be discussed later on.*

2.1 Introduction of HAP

The main problem investigated in this thesis is the hyperassignment problem in directed hypergraphs.

Problem 2.1.1 (HAP)**.**

Input: A pair (D, c_A) consisting of a directed hypergraph $D = (V, A)$ and a cost function $c_A : A \to \mathbb{R}$.

Output: A minimum cost hyperassignment in D w. r. t. c_A, i. e., a hyperassignment H^* in D such that

$$c_A(H^*) = \min\{c_A(H) : H \text{ is a hyperassignment in } D\},$$

or the information that no hyperassignment in D exists if this is the case.

We will be concerned with different ILP formulations for HAP. Here we

give a canonical formulation and prove its correctness.

$$\begin{aligned}
\underset{x\in\mathbb{R}^A}{\text{minimize}} \quad & \sum_{a\in A} c_A(a)x_a && && \text{(HAP)}\\
\text{subject to} \quad & \sum_{a\in\delta^{\text{in}}(v)} x_a - \sum_{a\in\delta^{\text{out}}(v)} x_a = 0 && \forall v\in V && \text{(i)}\\
& \sum_{a\in\delta^{\text{out}}(v)} x_a = 1 && \forall v\in V && \text{(ii)}\\
& x \geq 0 && && \text{(iii)}\\
& x \in \mathbb{Z}^A && && \text{(iv)}
\end{aligned}$$

Definition 2.1.2. Let

$$P_{\text{LP}}(\text{HAP}) := \{x \in \mathbb{R}^A : \text{(HAP) (i)–(iii)}\}$$

be the polyhedron associated with the LP relaxation of (HAP).

The variable x_a for $a \in A$ indicates whether the hyperarc a is contained in the hyperassignment. a is in the hyperassignment if and only if $x_a = 1$. Otherwise x_a is equal to 0.

The idea of the formulation is the following. We state the requirement that a solution of HAP is a hyperassignment as a so-called flow conservation constraint (HAP) (i), which ensures that the solution is a circulation, and (HAP) (i), which means that every vertex has exactly one outgoing hyperarc in the hyperassignment.

Lemma 2.1.3. *Given a directed hypergraph $D = (V,A)$ and a cost function $c_A : A \to \mathbb{R}$, there is a bijection between the feasible solutions of* (HAP) *and hyperassignments in D. The optimum value of* (HAP) *is equal to the cost of the minimum cost hyperassignment in D w. r. t. c_A if it exists and to ∞ otherwise.*

Proof. First of all, for every hyperassignment H in D $x = (x_a)_{a\in A}$ with

$$x_a = \begin{cases} 1 & \text{if } a \in H \\ 0 & \text{otherwise} \end{cases}$$

is a feasible solution of (HAP).

On the other hand, constraints (HAP) (ii)–(iv) imply that every feasible solution x of (HAP) lies in $\{0,1\}^A$. Let $H := \{a \in A : x_a = 1\}$. Then H is a hyperassignment in D. Indeed, the so called flow conservation constraint (HAP) (i)

corresponds to the requirement that for every vertex there is the same number of outgoing hyperarcs as ingoing hyperarcs, i. e., H is a circulation. (HAP) (ii) guarantees that the circulation is a hyperassignment.

The values of the minimized functions in (HAP) and the one used in HAP are preserved under the bijection. Finally, (HAP) and HAP use the same cost function under this bijection. □

Although we will not deal much with the multi-commodity version of HAP, we state it here for completeness.

Problem 2.1.4 ($\mathsf{HAP}_{\text{m-c}}$)**.**

Input: A pair $(D,(c_{A,i})_{i\in\{1,\dots,k\}})$ consisting of a directed hypergraph $D = (V,A)$ and cost functions $c_{A,i} : A \to \mathbb{R}$ for $i \in \{1,\dots,k\}$ for k commodities.

Output: A family $(H_i^*)_{i\in\{1,\dots k\}}$ of k pairwise disjoint circulations such that their union is a hyperassignment in D, minimizing

$$\sum_{i=1}^{k} c_{A,i}\left(H_i\right),$$

or the information that no such family exists if this is the case.

This is the usual multi-commodity generalization of flow problems applied to HAP.

2.2 HAP as a Minimum Cost Flow on Directed Hypergraphs

The well-known minimum cost flow problem on directed graphs can be generalized to directed hypergraphs. Instances of HAP can be formulated as instances of this problem.

For a directed hypergraph $D = (V,A)$ construct a new directed hypergraph $D' = (V',A')$ with

$$\begin{aligned} V' &= \{s,t\} \cup (V \times \{0,1\}), \\ A' &= \{a' : a \in A\} \cup \left\{ (\{s\},\{(v,0)\}), (\{(v,1)\},\{t\}) : v \in V \right\} \end{aligned}$$

by doubling the vertices and transforming each hyperarc $a \in A$ of D to a corresponding hyperarc $a' = (T_{a'}, H_{a'})$ with

$$T_{a'} = \left\{(v,0) : v \in T_a\right\}$$

and

$$H_{a'} = \left\{(v,1) : v \in H_a\right\}$$

of D'. Then, for every hyperassignment H in D the set of the corresponding hyperarcs in A' for the hyperarcs in H together with all out- and ingoing hyperarcs of D' of s and t, respectively, corresponds to an integral flow of $|V|$ units from s to t in D' with capacity equal to one on all hyperarcs, and vice versa.

To solve the minimum cost flow problem on directed hypergraphs, a hypergraph network simplex algorithm was introduced in [CGS92]. In contrast to the problem on graphs, for integral input optimal integral solutions do not have to exist. The generalization of the network simplex algorithm can produce non-integral solutions. Special cases where the set of feasible solutions is integral can be found in [JMRW92], but these usually do not apply to the instances gotten from HAP. However, the restriction that the heads of the hyperarcs may consist only of one element usually imposed on directed hypergraphs when speaking about flow problems is not a limitation—other hyperarcs can be split into two by adding a vertex.

2.3 HAP in Vehicle Rotation Planning

HAP arises in vehicle rotation planning for long distance passenger railways, which is a fundamental problem in rail transport. A regularity requirement in this application is the motivation to study HAP in this thesis and the reason why we will concentrate on the partitioned case. We first describe the vehicle rotation planning problem in general, and discuss regularity second.

Suppose a weekly repeating schedule with all trips that a railway company wants to provide is given. A trip is characterized by its departure weekday, departure time, departure location, arrival location and its duration.

Every trip has to be serviced by a vehicle. Between arrival and departure there may be several stops but the vehicle must not change during the trip.

After servicing a trip, the vehicle does a deadhead trip, possibly having a distance of zero, from the arrival location of the trip to the departure location of the next trip it services. This deadhead trip has some duration. Afterwards, when the weekday and departure time of the next trip arrives, the vehicle services this trip.

A vehicle rotation plan is an assignment of each trip to another trip. This assignment tells every vehicle which trip it has to service next after servicing a trip. Since the schedule is periodic, the sequence of trips for every vehicle is periodic, too. The period is a positive integral multiple of a week.

The cost of a pair of trips in the assignment depends on the duration and distance of a deadhead trip and the break between the trip before and after the deadhead. The last criterion is relevant, because the longer the breaks are the

more vehicles the railway company needs to service all trips.

The aim is to find an assignment with minimum cost. Thus, vehicle rotation planning as explained so far can be formulated as an assignment problem in a directed graph. The vertices of the graph are the trips and there is an arc from every trip to every trip.

An additional criterion which determines the quality of a vehicle rotation plan is the regularity: The more the operation of vehicles on different weekdays resembles each other, the easier is, e. g., personnel planning and adjustment of the plan if unforeseen events occur.

Let us group all trips which differ only be the departure weekday. Such a set is called a train. It is characterized by the departure time, the departure location, the arrival location, the duration of the trips it consists of, and a validity. The validity is the set of all weekdays on which the trips of the train start.

Given an assignment, we can count for each train the number of unequal deadheads in the trips assigned to the trips of the train. Two deadheads are unequal if the trains the next trips belong to are different or the breaks have a different length. Otherwise they are equal. The less the number of unequal deadheads in the vehicle rotation plan, the higher the regularity.

Since regularity simplifies the operation of vehicles, a high regularity is desirable. To take this criterion into account in vehicle rotation planning, we can model it as a hyperassignment problem. Therefore, we add hyperarcs. All trips in the tail and all trips in the head of a hyperarc belong to the same train, respectively. Further, for every hyperarc there is a set of arcs such that the disjoint union of their tails is the tail or the hyperarc, the disjoint union of their heads is the head of the hyperarc and the deadhead trips represented by these arcs are all equal. The cost of the hyperarc is less than the sum of the costs of the arcs. The difference is the bonus for regularity.

The resulting hypergraph is graph based and partitioned. Its parts are the trains. The maximum part size is the number of weekdays, i. e., seven.

If different types of vehicles with possibly different deadhead trip costs are involved and we have given a set of possible vehicle types for every trip, the problem translates into a $\mathsf{HAP}_{\text{m-c}}$.

Requirements such as maintenance of trains after a given kilometer limit will not be considered in this thesis.

2.4 The Set Partitioning Problem

The set partitioning problem in undirected hypergraphs (SPP) has many connections to the hyperassignment problem (HAP), as we will see in the next chapter.

Problem 2.4.1 (SPP)**.**

Input: A pair (U, c_E) consisting of an undirected hypergraph $U = (N, E)$ and a cost function $c_E : E \to \mathbb{R}$.

Output: A minimum cost partitioning of U w. r. t. c_E, i. e., a partitioning K^* of U such that

$$c_E(K^*) = \min\{c_E(K) : K \text{ is a partitioning of } U\},$$

or the information that no partitioning exists if this is the case.

The following is an ILP formulation for the set partitioning problem. We will prove its correctness in the next lemma.

$$\begin{aligned}
\underset{x \in \mathbb{R}^E}{\text{minimize}} \quad & \sum_{e \in E} c_E(e) x_e && && \text{(SPP)}\\
\text{subject to} \quad & \sum_{e \in E : n \in e} x_e = 1 && \forall n \in N && \text{(i)}\\
& x \geq 0 && && \text{(ii)}\\
& x \in \mathbb{Z}^E && && \text{(iii)}
\end{aligned}$$

The variable x_e for $e \in E$ indicates whether the set e is contained in the partitioning. e is in the partitioning if and only if $x_e = 1$. Otherwise x_e is equal to 0.

By the definition of a partitioning every $n \in N$ is contained in exactly one set in the partitioning. This requirement is described by (SPP) (i).

Lemma 2.4.2. *Given an undirected hypergraph $U = (N, E)$ and a cost function $c_E : E \to \mathbb{R}$, there is a bijection between the feasible solutions of* (SPP) *and partitionings of U. The optimum value of* (SPP) *is equal to the cost of the minimum cost partitioning of U w. r. t. c_E if it exists and to ∞ otherwise.*

Proof. Similar to the proof of Lemma 2.1.3, the constraints imply that $x \in \{0, 1\}^E$, we want a set $e \in E$ to be contained in a partitioning if and only if $x_e = 1$ in the corresponding solution to (SPP), (SPP) (i) corresponds to the requirement that every element of X is contained in exactly one set of a cover, and the minimized functions are preserved under the bijection. □

For an investigation of SPP see [Bor98]. Here we only want to remark that SPP is $\mathcal{NP}$-hard.

2.5 Overview of Further ILP Formulations of HAP

This section gathers further ILPs for HAP, which we will consider later.

The first ILP works for arbitrary directed hypergraphs. It formulates HAP in terms of SPP. We will proof its correctness in the next chapter.

$$\begin{aligned}
\underset{x\in\mathbb{R}^A}{\text{minimize}} \quad & \sum_{a\in A} c_A(a)x_a && && \text{(HAP_SP)}\\
\text{subject to} \quad & \sum_{a\in\delta^{\text{dir}}(v)} x_a = 1 && \forall v\in V, \text{dir}\in\{\text{out},\text{in}\} && \text{(i)}\\
& x \geq 0 && && \text{(ii)}\\
& x \in \mathbb{Z}^A && && \text{(iii)}
\end{aligned}$$

The variables have the same meaning as in (HAP). The idea of this formulation is to state that feasible solutions have to be hyperassignments by requiring that every vertex of the directed hypergraph has exactly one outgoing and one ingoing hyperarc in the hyperassignment.

For the following two ILPs, we assume that D is partitioned. We will discuss them in chapter 4.

The next ILP can be stated for $\text{dir}\in\{\text{out},\text{in}\}$.

$$\begin{aligned}
\underset{x\in\mathbb{R}^A, y^{\text{dir}}\in\mathbb{R}^{\mathscr{C}^{\text{dir}}}}{\text{minimize}} \quad & \sum_{a\in A} c_A(a)x_a && && \text{(HAP_P}^{\text{dir}}\text{)}\\
\text{subject to} \quad & \sum_{C\in\mathscr{C}^{\text{dir}}:a\in C} y_C^{\text{dir}} = x_a && \forall a\in A && \text{(i)}\\
& \sum_{a\in\delta^{\text{rid}}(v)} x_a = 1 && \forall v\in V, \text{rid}\in\{\text{out},\text{in}\} && \text{(ii)}\\
& x, y^{\text{dir}} \geq 0 && && \text{(iii)}\\
& x \in \mathbb{Z}^A && && \text{(iv)}\\
& y^{\text{dir}} \in \mathbb{Z}^{\mathscr{C}^{\text{dir}}} && && \text{(v)}
\end{aligned}$$

Definition 2.5.1. Let

$$P_{\text{LP}}(\text{HAP\_P}^{\text{dir}}) := \{(x, y^{\text{dir}}) \in \mathbb{R}^A \times \mathbb{R}^{\mathscr{C}^{\text{dir}}} : (\text{HAP\_P}^{\text{dir}})\ \text{(i)–(iii)}\}$$

be the polyhedron associated with the LP relaxation of (HAP_P$^{\text{dir}}$). Further, let

$$\pi_x^{\text{P}^{\text{dir}}} : \mathbb{R}^A \times \mathbb{R}^{\mathscr{C}^{\text{dir}}} \to \mathbb{R}^A, \quad (x, y^{\text{dir}}) \mapsto x$$
$$\pi_{x^{P_i}}^{\text{P}^{\text{dir}}} : \mathbb{R}^A \times \mathbb{R}^{\mathscr{C}^{\text{dir}}} \to \mathbb{R}^{P_i}, \quad (x, y^{\text{dir}}) \mapsto (x_a)_{a \in \delta^{\text{dir}}(P_i)}$$

be mappings that produce projections onto the coordinates of all the hyperarc variables and the hyperarc variables for the outgoing or ingoing hyperarcs of one part P_i, respectively.

This is an extension of the formulation (HAP_SP). Additionally to the variables and inequalities from (HAP_SP) there are variables y_C^{dir} for $C \in \mathscr{C}^{\text{dir}}$ for every out- or ingoing configuration.

The idea of this formulation is to additionally include that every hyperassignment can be uniquely partitioned into out- or ingoing configurations. As we will see in chapter 4, this makes the set of feasible solutions of the LP relaxation smaller.

$C \in \mathscr{C}^{\text{dir}}$ is a subset of the hyperassignment, which means that all hyperarcs $a \in C$ are in the hyperassignment, if and only if $y_C^{\text{dir}} = 1$. Otherwise y_C^{dir} is equal to 0. This is ensured by (HAP_P$^{\text{dir}}$) (i).

The next ILP is very similar to (HAP_P$^{\text{dir}}$). It includes that hyperassignment can be uniquely partitioned both into out- and ingoing configurations whereas the last ILP used only out- or ingoing configurations.

$$\underset{x \in \mathbb{R}^A, y^{\text{out}} \in \mathbb{R}^{\mathscr{C}^{\text{out}}}, y^{\text{in}} \in \mathbb{R}^{\mathscr{C}^{\text{in}}}}{\text{minimize}} \quad \sum_{a \in A} c_A(a) x_a \tag{HAP_P}$$

$$\text{subject to} \quad \sum_{C \in \mathscr{C}^{\text{dir}} : a \in C} y_C^{\text{dir}} = x_a \qquad \forall a \in A, \text{dir} \in \{\text{out}, \text{in}\} \tag{i}$$

$$\sum_{a \in \delta^{\text{dir}}(v)} x_a = 1 \qquad \forall v \in V, \text{dir} \in \{\text{out}, \text{in}\} \tag{ii}$$

$$x, y^{\text{out}}, y^{\text{in}} \geq 0 \tag{iii}$$

$$x \in \mathbb{Z}^A \tag{iv}$$

$$y^{\text{dir}} \in \mathbb{Z}^{\mathscr{C}^{\text{dir}}} \qquad \forall \text{dir} \in \{\text{out}, \text{in}\} \tag{v}$$

Definition 2.5.2. Let

$$P_{\text{LP}}(\text{HAP\_P}) := \{(x, y^{\text{out}}, y^{\text{in}}) \in \mathbb{R}^A \times \mathbb{R}^{\mathscr{C}^{\text{out}}} \times \mathbb{R}^{\mathscr{C}^{\text{in}}} : \text{(HAP\_P) (i)–(iii)}\}$$

be the polyhedron associated with the LP relaxation of (HAP_P). Further, let

$$\pi_x^{\text{P}} : \mathbb{R}^A \times \mathbb{R}^{\mathscr{C}^{\text{out}}} \times \mathbb{R}^{\mathscr{C}^{\text{in}}} \to \mathbb{R}^A, \quad (x, y^{\text{out}}, y^{\text{in}}) \mapsto x$$

be a mapping that produces a projection onto the coordinates of all the hyperarc variables.

The last ILP works only for graph based hypergraphs where the $B(a)$ for $a \in A \setminus B$ are pairwise disjoint. It will be discussed in section 3.6.

In addition to the x-variables, which have the same meaning as before, we add variables z_b for $b \in B$. $z_b = 1$ for $b \in B$ means that either b or the hyperarc $a \in A$ with $b \in B(a)$ is in the hyperassignment. We describe HAP on the graph base using the z-variables and use coupling constraints between the x- and z-variables.

$$\begin{aligned}
\underset{x\in\mathbb{R}^A, z\in\mathbb{R}^B}{\text{minimize}} \quad & \sum_{a\in A} c_A(a)x_a && && \text{(HAP_GB)}\\
\text{subject to} \quad & \sum_{b\in\delta_B^{\text{in}}(v)} z_b - \sum_{b\in\delta_B^{\text{out}}(v)} z_b = 0 && \forall v \in V && \text{(i)}\\
& \sum_{b\in\delta_B^{\text{in}}(v)} z_b = 1 && \forall v \in V && \text{(ii)}\\
& \sum_{b\in B(a)} z_b \geq |B(a)| \cdot x_a && \forall a \in A\setminus B && \text{(iii)}\\
& x_b = z_b - \sum_{a\in A\setminus B : b\in B(a)} x_a && \forall b \in B && \text{(iv)}\\
& z \geq 0 && && \text{(v)}\\
& x \in \mathbb{Z}^A && && \text{(vi)}\\
& z \in \mathbb{Z}^B && && \text{(vii)}
\end{aligned}$$

Definition 2.5.3. Let

$$P_{\text{LP}}(\text{HAP\_GB}) := \{(x,z) \in \mathbb{R}^A \times \mathbb{R}^B : \text{(HAP\_GB) (i)–(v)}\}$$

be the polyhedron associated with the LP relaxation of (HAP_GB). Further, let

$$\pi_x^{\text{GB}} : \mathbb{R}^A \times \mathbb{R}^B \to \mathbb{R}^A, \quad (x,z) \mapsto x$$

be mapping that produces a projection onto the coordinates of all the hyperarc variables.

Chapter 3

Complexity of HAP

Summary of this chapter *We begin with looking for indications for the complexity of* HAP*: A reformulation as* SPP*, which does not have obvious particular properties, large basis matrix determinants, and an arbitrarily large integrality gap in* (HAP)*. Then, we prove that* HAP *is $\mathcal{NP}$-hard, even if we impose several restrictions. We use a transformation from* SPP *and the 3-matching problem to* HAP*.*

3.1 HAP in Terms of SPP

We can formulate the hyperassignment problem as a set partitioning problem. This is shown in the next lemma.

The only restriction which holds for the resulting undirected hypergraph is that its nodes can be partitioned into two sets such that every hyperedge contains at least one node from each set. Since SPP is $\mathcal{NP}$-hard and the restriction does not seem to be strong, this suggests that HAP is also $\mathcal{NP}$-hard. We will proof this in section 3.4.

Lemma 3.1.1. *Given a directed hypergraph $D = (V,A)$ and a cost function $c_A : A \to \mathbb{R}$, define*

$$\begin{aligned} N &:= V \times \{0,1\}, \\ e(a) &:= \{(v,0) : v \in T_a\} \cup \{(v,1) : v \in H_a\} \text{ for } a \in A, \\ E &:= \{e(a) : a \in A\}, \\ c_E(e(a)) &:= c_A(a). \end{aligned}$$

Then there is a bijection between exact circulations H in D and partitionings K of

$U = (N, E)$ *such that*

$$c_A(H) = c_E(K)$$

if K is the image of H under the bijection.

(HAP_SP) *is a correct ILP formulation of* HAP *and the set of feasible solutions of its LP relaxation is the same as for* (HAP).

Proof. We will prove this using the ILPs. For the undirected hypergraph U defined above, (SPP) reads

$$\begin{aligned}
\underset{x\in\mathbb{R}^E}{\text{minimize}} \quad & \sum_{e\in E} c_E(x_e)x_e \\
\text{subject to} \quad & \sum_{e\in E: n\in e} x_k = 1 \qquad \forall n \in V \times \{0,1\} && \text{(i)} \\
& x \geq 0 && \text{(ii)} \\
& x \in \mathbb{Z}^E && \text{(iii)}
\end{aligned}$$

If we now split (i) into the equations for $n \in V \times \{0\}$ and $n \in V \times \{1\}$ observing that

$$\{e \in E : (v,0) \in e\} = \{e(a) \in E : v \in T_a\} = \{e(a) \in E : v \in \delta^{\text{out}}(a)\}$$

and

$$\{e \in E : (v,1) \in e\} = \{e(a) \in E : v \in H_a\} = \{e(a) \in E : v \in \delta^{\text{in}}(a)\}$$

we get

$$\sum_{e(a)\in E: v\in\delta^{\text{out}}(a)} x_{e(a)} = 1 \quad \forall v \in V, \qquad \text{(i)'}$$

$$\sum_{e(a)\in E: v\in\delta^{\text{in}}(a)} x_{e(a)} = 1 \quad \forall v \in V. \qquad \text{(i)"}$$

Now, using the bijective correspondence $b : \mathbb{R}^A \to \mathbb{R}^E, (x_a)_{a\in A} \mapsto (x_{e(a)})_{a\in A}$, we can see that for the defined undirected hypergraph (HAP) and (SPP) consist of exactly the same equations and inequalities after substituting (HAP) (ii) in (HAP) (i) for each $v \in V$.

This partitioning formulation of HAP is the one given in (HAP_SP). □

Remark 3.1.2. The previous lemma suggests that we may interpret HAP as optimizing over the intersection of the two set partitioning problems given by (HAP_SP)(i) for dir=in and dir=out.

If all hyperarcs of D are arcs, the formulation also implies that in this case HAP is a minimum cost bipartite perfect matching problem. In this case, the set of feasible solutions of the LP relaxation of (HAP_SP) is an integral polytope.

The transformation is also possible for the multi-commodity version of HAP. There we use

$$N := V \times \{0,\dots,k\},$$
$$e(a,i) := \{(v,0) : v \in T_a\} \cup \{(v,i) : v \in H_a\} \cup \{(v,j) : v \in T_a, j \neq i\}$$

for $a \in A$ and $i \in \{1,\dots,k\}$,

$$E := \{e(a,i) : a \in A, i \in \{1,\dots,k\}\},$$
$$c_E(e(a,i)) := c_{A,i}(a).$$

For $k = 1$ we get the transformation from above.

3.2 Determinants of Basis Matrices

The determinant of submatrices of coefficient matrices is an indicator for the complexity of ILPs. For example, if the coefficient matrix is totally unimodular, the LP relaxation is integral. In general, by Cramer's rule, the denominator of the variable values in a basic solution of a LP is (if the numerator and denominator are relatively prime) at most the determinant of the basis matrix.

For the LP relaxation of the (HAP) formulation of HAP, the denominator, and therefore also the determinant of basis matrices, can be arbitrarily large. This is even the case if one allows only hyperarcs with head and tail cardinality at most two. Am example of this is as follows.

Let s be a positive integer and consider the following directed hypergraph $D = (V,A)$ with $2s+1$ vertices and $3s$ hyperarcs with tail and head cardinality at most two. We want $V = \{u, v_i, w_i, : i \in \{0,\dots,s-1\}\}$ and $A = A_1 \cup A_2$ with

$$A_1 = \{(\{v_i, w_i\}, \{v_i, w_i\}) : i \in \{0,\dots,s-1\}\},$$
$$A_2 = \left\{\left(\{u, v_i\}, \{w_{(i+1) \mod s}, u\}\right), (\{w_i\}, \{u\}), (\{u\}, \{v_i\}) : i \in \{0,\dots,s-1\}\right\}.$$

The only feasible solution of the LP relaxation of (HAP) is $x_a = \frac{2s-1}{2s}$ for all $a \in A_1$ and $x_a = \frac{1}{2s}$ for all $a \in A_2$. Thus the determinant of the basis matrix is at least $2s$.

An upper bound on the modulus of the determinant is

$$\prod_{a \in A} |T_a|$$

if the hypergraph $D = (V,A)$ can be extended to a graph based hypergraph by adding arcs (this is also true for the hypergraph of the example). This is

the case if head and tail cardinalities are equal for each hyperarc. Since every column of the basis matrix can be represented as the sum of columns for the corresponding arcs and basis matrices of (HAP) for directed graphs are totally unimodular, i. e., have determinant with modulus 0 or 1, we can apply the multilinearity of the determinant until we get only such matrices and obtain the bound.

3.3 Integrality Gap

The gap between the optimum solution of the LP relaxation of (HAP) and the minimum cost hyperassignment can be arbitrarily large. An example is given in figure 3.1.

3.4 HAP Is $\mathcal{NP}$-hard

HAP is $\mathcal{NP}$-hard, as will be shown now. Our first proof uses a transformation from the decision problem version of SPP to HAP and works even if we only allow directed hypergraphs as HAP input with tail and head cardinality at most three for all hyperarcs.

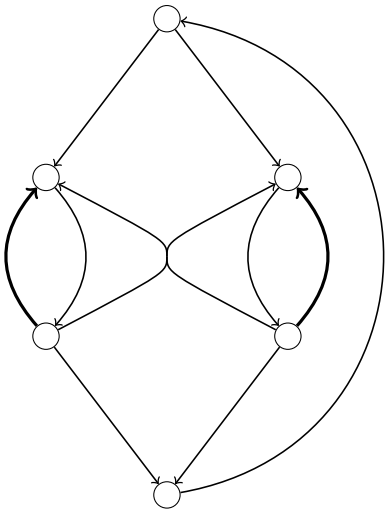
Theorem 3.4.1. *Given a directed hypergraph $D = (V,A)$ satisfying $|T_a| = |H_a| \leq 3$ for all $a \in A$ and a cost function $c_A : A \to \mathbb{R}$, HAP with input (D, c_A) is $\mathcal{NP}$-hard.*

Proof. The problem to decide whether a partitioning of an undirected hypergraph $U = (N,E)$ with $|N| = 3s, s \in \mathbb{N}$ and $|e| = 3$ for all $e \in E$ exists is $\mathcal{NP}$-complete (see [GJ79], page 53).

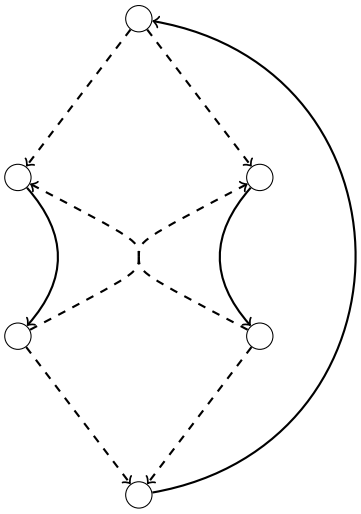
Construct the directed hypergraph $D = (V,A)$ with $V = N$ and $A = \{(e,e) : e \in E\}$. This can be done in polynomial time and the resulting hypergraph satisfies $|T_a| = |H_a| \leq 3$ for all $a \in A$. Choose $c_A : A \to \mathbb{R}, c_A \equiv 0$. Then HAP with input (D, c_A) returns a hyperassignment with cost 0 if and only if a partitioning of U exists. □

The theorem leaves the case $|T_a| = |H_a| = 2$ open (for $|T_a| = |H_a| = 1$ see Remark 3.1.2). For $|T_a| = |H_a| = 2$ we get an $\mathcal{NP}$-hard problem, but here we need a slightly more complicated transformation.

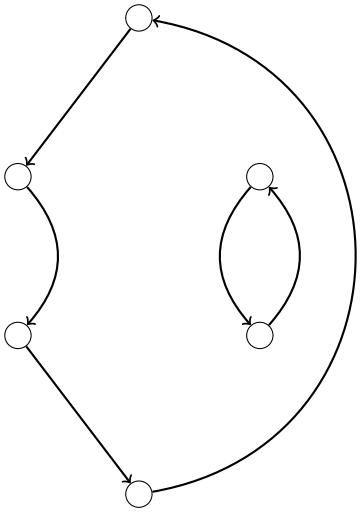
Theorem 3.4.2. *Given a directed hypergraph $D = (V,A)$ satisfying $|T_a| = |H_a| \leq 2$ for all $a \in A$ and a cost function $c_A : A \to \mathbb{R}$, HAP with input (D, c_A) is $\mathcal{NP}$-hard.*



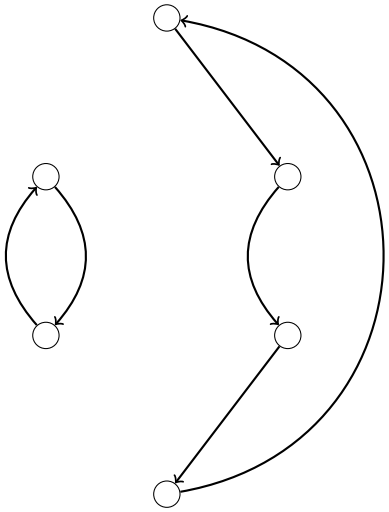
(a) A (graph based, for $d \geq 2$ partitioned) directed hypergraph. The thicker hyperarcs have cost $g > 0$, the others have cost 0.



(b) The optimum solution of the LP relaxation of (HAP). Solid hyperarcs have the value 1, dashed hyperarcs the value 0.5. The optimum value is 0.



(c) Circulation in the hypergraph. It has cost g.



(d) The other circulation in the hypergraph. It also has cost g.

Figure 3.1: Example of an instance of HAP with only two circulations and arbitrary large integrality gap g for (HAP).

Proof. The 3-dimensional matching problem, which given an undirected hypergraph $U = (N \uplus O \uplus P, E)$, $|N| = |O| = |P|$,

$$|e \cap N| = |e \cap O| = |e \cap P| = 1 \quad \forall e \in E$$

asks whether a partitioning of U exists, is $\mathcal{NP}$-complete (see [GJ79], page 46).

Construct the directed hypergraph $D = (V, A)$ with

$$V = N \cup O \cup \big\{\{p\} \times \{0,1\} : p \in P\big\}$$

and

$$\begin{aligned} A &= A_1 \cup A_2, \\ A_1 &= \Big\{\big((e \cap N) \cup (e \cap O), \{(e \cap P, 0), (e \cap P, 1)\}\big) : e \in E\Big\}, \\ A_2 &= \Big\{\big(\{(e \cap P, 0)\}, e \cap N\big), \big(\{(e \cap P, 1)\}, e \cap O\big) : e \in E\Big\}. \end{aligned}$$

This can be done in polynomial time and the resulting hypergraph satisfies $|T_a| = |H_a| \leq 2$ for all $a \in A$. Choose $c_A : A \to \mathbb{R}, c_A \equiv 0$. Then HAP with input (D, c_A) returns a hyperassignment with cost 0 if and only if a partitioning of U exists. The chosen hyperarcs from A_1 correspond to the $e \in E$ in the partitioning of U. □

3.5 HAP for Partitioned Hypergraphs

The resulting directed hypergraphs in the transformations from above are not partitioned. We will prove next that HAP for partitioned hypergraphs is also $\mathcal{NP}$-hard.

Theorem 3.5.1. *Given a partitioned hypergraph $D = (V, A)$ satisfying $|T_a| = |H_a| \leq 3$ for all $a \in A$, $d \leq 3$ and a cost function $c_A : A \to \mathbb{R}$, HAP with input (D, c_A) is $\mathcal{NP}$-hard.*

Proof. Again, we use a transformation from the 3-dimensional matching problem on the undirected hypergraph $U = (N \uplus O \uplus P, E)$, $|N| = |O| = |P|$,

$$|e \cap N| = |e \cap O| = |e \cap P| = 1 \quad \forall e \in E.$$

Construct the directed hypergraph $D = (V, A)$ with

$$V = (E \times \{N, O, P\}) \cup (N \cup O \cup P)$$

and

$$A = \Big\{ \big(e \cap N, \{(e,N)\}\big) : e \in E \Big\}$$
$$\cup \Big\{ \big(e \cap O, \{(e,O)\}\big) : e \in E \Big\} \cup \Big\{ \big(e \cap P, \{(e,P)\}\big) : e \in E \Big\}$$
$$\cup \Big\{ \big(\{(e,N)\}, e \cap N\big) : e \in E \Big\}$$
$$\cup \Big\{ \big(\{(e,O)\}, e \cap O\big) : e \in E \Big\} \cup \Big\{ \big(\{(e,P)\}, e \cap P\big) : e \in E \Big\}$$
$$\cup \Big\{ \big(\{(e,N),(e,O),(e,P)\}, \{(e,N),(e,O),(e,P)\}\big) : e \in E \Big\}.$$

This can be done in polynomial time. For $d = 3$, D is partitioned. The parts are $\{(e,N),(e,O),(e,P)\}$ for $e \in E$ and all the one element subsets of $N \cup O \cup P$. All hyperarcs $a \in A$ satisfy $|T_a| = |H_a| \leq 3$.

Now, we prove that there exists a hyperassignment in D if and only if there exists a partitioning of U. This proves the theorem. Just define $c_A : A \to \mathbb{R}, c_A \equiv 0$. Then HAP with input (D, c_A) returns a hyperassignment with cost 0 if and only if a partitioning of U exists.

Indeed, assume that K is a partitioning of U. Then

$$H = \Big\{ \big(e \cap N, \{(e,N)\}\big) : e \in K \Big\}$$
$$\cup \Big\{ \big(e \cap O, \{(e,O)\}\big) : e \in K \Big\} \cup \Big\{ \big(e \cap P, \{(e,P)\}\big) : e \in K \Big\}$$
$$\cup \Big\{ \big(\{(e,N)\}, e \cap N\big) : e \in K \Big\}$$
$$\cup \Big\{ \big(\{(e,O)\}, e \cap O\big) : e \in K \Big\} \cup \Big\{ \big(\{(e,P)\}, e \cap P\big) : e \in K \Big\}$$
$$\cup \Big\{ \big(\{(e,N),(e,O),(e,P)\}, \{(e,N),(e,O),(e,P)\}\big) : e \in E \setminus K \Big\}.$$

is a hyperassignment in D. K being a partitioning guarantees that $|\delta_H^{\text{out}}(v)| = |\delta_H^{\text{in}}(v)| = 1$ for $v \in N \cup O \cup P$. For the other vertices the requirement is satisfied automatically for every subset K of E by the construction.

On the other hand, if H is a hyperassignment in D, by construction for every $e \in E$ holds either

$$\big(\{(e,N),(e,O),(e,P)\}, \{(e,N),(e,O),(e,P)\}\big) \in H$$

or

$$\big(e \cap N, \{(e,N)\}\big), \big(e \cap O, \{(e,O)\}\big), \big(e \cap P, \{(e,P)\}\big) \in H$$

because $\{(e,N),(e,O),(e,P)\}$ has no other ingoing hyperarcs. Define K such that $e \in E$ is in K in the second case and e is not in K otherwise. Since H

is a hyperassignment and there are no other ingoing hyperarcs for $N \cup O \cup P$ than those from the second case, we get for each node of U exactly one $e \in E$ containing it. Thus, K is a partitioning of U.

□

For the case where the input is a partitioned hypergraphs and $d = 2$ the complexity remains open. For partitioned hypergraphs with $d = 1$, again all hyperarcs are arcs and HAP is solvable in polynomial time.

3.6 HAP for Graph Based Hypergraphs

The directed hypergraphs constructed in the proofs of Theorems 3.4.1, 3.4.2 and 3.5.1 are not graph based. But because all hyperarcs have equal tail and head cardinalities one can add arcs to the hypergraphs such that they become graph based and assign costs > 0 to the arcs. Then the proofs are still correct and therefore the results also hold for graph based hypergraphs.

Still, one can try to reduce HAP for graph based hypergraphs to the problem for the base graph which has an integral LP relaxation and then add the hyperarcs which are not arcs in an ILP formulation.

In (HAP_GB), we give an example of how this can be done.

(HAP_GB) (i) and (HAP_GB) (ii) describe HAP for the base graph in the way it is done in (HAP) by using the z-variables. $z_b = 1$ for $b \in B$ means that either b or the hyperarc $a \in A$ with $b \in B(a)$ is in the hyperassignment. Thus, the values for arcs in the variable vector z have a different meaning than in x. $z_b = 1$ for $b \in B$ does not necessarily mean that b is in the hyperassignment. The meaning of the x-variables is the same as in (HAP).

The connection between the x- and z-variables could be made by a constraint like

$$z_b = \sum_{a \in A : b \in B(a)} x_a,$$

but after substituting this in (HAP_GB) (i) and (HAP_GB) (ii), we would get exactly the constraints from (HAP).

A different possibility is given by (HAP_GB) (iii) and (HAP_GB) (iv). It can be applied only if the heads and tails of the hyperarcs $B(a)$ for $a \in A \setminus B$ are pairwise disjoint. (HAP_GB) (iii) says, when a hyperarc can be chosen, namely, if all the z-values for the arcs for this hyperarc from the graph base are chosen. (HAP_GB) (iv) calculates from this the values of the arcs. The sum in this inequality sums up at most one element.

However, the following lemma shows that this formulation does not lead to a good LP relaxation.

Lemma 3.6.1. *Let $D = (V,A)$ be a graph based hypergraph, such that the heads and tails of the hyperarcs $B(a)$ for $a \in A \setminus B$ are pairwise disjoint, and let $c_A : A \to \mathbb{R}$ be a cost function. Then $\pi_x^{GB}(P_{LP}(HAP\_GB)) \subseteq P_{LP}(HAP)$, i. e. the projection of the set of feasible solutions of the LP relaxation of* (HAP_GB) *to x is a subset of the set of feasible solutions of the LP relaxation of* (HAP). *There exist hypergraphs where it is a proper subset.*

Proof. For the first statement calculate the value of z in (HAP_GB) by using (HAP_GB) (iv) from a feasible solution of (HAP). These values satisfy (HAP_GB) (i) and (HAP_GB) (ii), because after substitution of the z_b they are (HAP) (i) and (HAP) (ii). (HAP_GB) (iii) is implied by the nonnegativity constraints for $x_a, a \in A \setminus B$ in (HAP), because after substitution of z_b from (HAP_GB) (iv) this constraint is just $x_a \geq 0$.

For an example where the sets are not equal, see figure 3.2. □

If we add the constraint $x_b \geq 0$ for $b \in B$ to (HAP_GB), the sets become equal again.

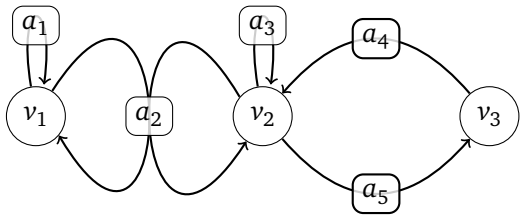


Figure 3.2: Graph based hypergraph $D = (V,A)$, $V = \{v_1, \dots v_3\}$, $A = \{a_1, \dots a_5\}$. $z_{a_1} = z_{a_4} = z_{a_5} = 1$, $z_{a_3} = 0$, $x_{a_1} = x_{a_3} = 0.5$, $x_{a_2} = -0.5$, $x_{a_4} = x_{a_5} = 1$ is a feasible solution of the LP relaxation of (HAP_GB) but its projection to x is infeasible for (HAP).

Chapter 4

Configuration ILP and its Advantages

Summary of this chapter *To begin with, we present computational results for* HAP *using instances arising from our application setting. They suggest that the elimination of fractional solutions of the LP relaxation* (HAP) *which violate so called clique inequalities is very important. To address this issue we propose a different ILP for* HAP *in partitioned hypergraphs using configurations whose LP relaxation automatically prohibits such solutions. In the end, we show that the novel formulation separates more than only cliques.*

4.1 Computational Results using (HAP)

Trying to solve practical instances of HAP using the ILP (HAP) showed that the separation of clique inequalities, which state that the sum of the variables corresponding to a clique in the conflict graph (see next section for an explanation) is less than or equal to one, is very important. The computational results (see figure 4.1) show that the gap between the LP solution and the ILP solution is very small if one adds enough clique inequalities, although it can be large without them. The LP bound improved by adding cliques by up to 20 % and the LP-IP gap reduced in many cases to a value of less than 1 %.

The instances are taken from a project with DB Fernverkehr AG, which deals with long distance passenger railway transport. These instances describe cyclic weekly schedules of the ICE 1 trains which operate in Germany.

Table 4.1: Computational results with real-world vehicle rotation planning problems using (HAP) and CPLEX 12.1.0. The LP-IP gap is given by $1 - \frac{L}{I}$, where L is the optimum value of the LP relaxation and I is the best integral solution found by CPLEX or Gurobi 3.0.0. The root gap is $1 - \frac{R}{I}$, where R is the optimum value of the LP relaxation before branching but after applying the cuts described in the last two columns. The root improvement is $\frac{R}{L} - 1$.

# rows ($2 \cdot \|V\|$)	# columns ($\|A\|$)	nonzeros	LP-IP gap	root gap	root improvement	# clique cuts	# other cuts
534	52056	140081	11.16 %	6.81 %	4.90 %	160	14
620	80477	236020	8.72 %	0.00 %	9.54 %	120	2
812	102375	216566	0.38 %	0.18 %	0.20 %	24	16
1128	267542	732134	4.59 %	0.26 %	4.55 %	263	0
1310	363513	1006024	7.85 %	0.22 %	8.28 %	378	2
1496	469932	1369224	18.70 %	1.86 %	20.71 %	809	0
1696	618348	1787078	5.17 %	0.16 %	5.28 %	925	0
1746	649525	1859898	7.52 %	4.88 %	2.86 %	563	0
1798	647650	1822718	13.60 %	0.95 %	14.65 %	537	0
1798	647650	1822718	13.35 %	0.62 %	14.69 %	604	0
2006	855153	2491372	5.76 %	0.68 %	5.39 %	1025	0
2260	1079535	3138752	9.89 %	2.03 %	8.73 %	954	0
2502	1290750	3680124	7.06 %	0.76 %	6.79 %	801	0
2620	1432355	4187296	9.05 %	1.15 %	8.68 %	1068	0
2624	1439453	4087042	14.17 %	5.23 %	10.41 %	951	0

4.2 Analysis of Cliques in the Conflict Graph

If Q is a clique in the conflict graph of a directed hypergraph D, then

$$\sum_{a \in Q} x_a \leq 1$$

is a valid inequality for every (integral) feasible solution of (HAP), because no two hyperarcs, which are both contained in a clique, can be simultaneously in a hyperassignment in D. Such an inequality is called clique inequality and can be used to separate fractional feasible solutions of the LP relaxation of (HAP), which violate this inequality.

Let us call the only hyperarc which is no arc in the hypergraph from figure 3.1 a_1 and the two outgoing arcs of its tail a_2 and a_3. Then the clique inequality

$$x_{a_1} + x_{a_2} + x_{a_3} \leq 1$$

does not hold for the fractional solution in figure 3.1 (b). There,

$$x_{a_1} + x_{a_2} + x_{a_3} = 0.5 + 0.5 + 0.5 = 1.5.$$

If we add this inequality to (HAP) we cannot get this solution as an optimum any more.

The aim is to find maximal cliques, because for cliques which are not maximal, the clique inequalities are usually no facets of the polytope defined by the set of feasible solutions of the LP relaxation.

We begin this section with a lemma for partitioned hypergraphs which states that it is enough to look for cliques in clique graphs. Every clique in the conflict graph of a partitioned hypergraph is a clique in a clique graph of the partitioned hypergraph. Since clique graphs, which are induced subgraphs of the conflict graph, are smaller than the whole conflict graph this makes it easier to find cliques.

Then we proceed with an analysis of clique graphs. We want to study whether there is a possibility to easily find maximal cliques. Since a maximal clique in a perfect graph can be found in polynomial time ([GLS88]), it is interesting to find out in which cases clique graphs are perfect. For this purpose we investigate the holes and antiholes of clique graphs. We concentrate on the situation where $d \leq 7$ because this is the important case for vehicle rotation planning.

Lemma 4.2.1. *Let $D = (V,A)$ be a partitioned hypergraph. Then, every clique in the conflict graph G of D is a subset of the nodes of some clique graph of D.*

Proof. Let Q be a clique in G containing some hyperarc a. Thus, every other hyperarc b in Q must have $t(b) = h(a)$ or $h(b) = h(a)$ Assume that Q contains some hyperarc b_t with $h(b_t) \neq h(a)$ and some hyperarc b_h with $t(b_h) \neq t(a)$. Now, b_t and b_h cannot be connected by an edge in G since $t(b_t) = t(a) \neq t(b_h)$ and $h(b_t) \neq h(a) = h(b_h)$, which contradicts the assumption that Q is a clique. Hence, the tails or heads of all the hyperarcs in Q lie in one part and therefore are all nodes of one clique graph. □

Lemma 4.2.2. *Let $D = (V,A)$ be a partitioned hypergraph and let G be an outgoing (ingoing) clique graph of D. Then, for each of the at most $|A|$ hyperarcs a with $|T_a| = 1$ ($|H_a| = 1$), a maximal clique in G containing a can be found in polynomial time.*

Proof. W. l. o. g. we will consider only an outgoing clique graph in the proof.

A maximal clique in G containing a can only contain hyperarcs of two types. The first type are hyperarcs from $S := \{b \in A : T_a \subseteq T_b\}$, which form a clique. The others are hyperarcs c with $H_c \cap H_a \neq \emptyset$. There are at most 2^{d-1} possibilities for their heads and at most 2^d possibilities for their tails (the parts from which to which they go are fixed) for the second type. Thus, the number of such hyperarcs is constant for constant d. For each possible subset of these hyperarcs (there is also a constant number of subsets) check whether it is a clique. If this the case, add all hyperarcs from S which are connected to all of these hyperarcs. The result is a clique. The clique with maximum cardinality from all such cliques is then a maximal clique.. □

In what follows, we will use that in perfect graphs a maximal clique can be found in polynomial time (see [GLS88]). Complements of perfect graphs are perfect by definition. Bipartite graphs are perfect, because all its holes and antiholes have an even size.

Lemma 4.2.3. *Let $D = (V,A)$ be a partitioned hypergraph and let G be an outgoing (ingoing) clique graph of D. Then for each of the at most $|A|$ hyperarcs a with $|T_a| = 2$ ($|H_a| = 2$), a maximal clique in G containing a can be found in polynomial time.*

Proof. We can proceed as in the previous proof with the only difference that now the subsets of hyperarcs of the second type with all the hyperarcs from S which are connected to all of them induce the complement of a bipartite graph instead of a clique. However, complements of bipartite graphs are perfect and we can find the maximal cliques there in polynomial time. □

Lemma 4.2.4. *Let $D = (V,A)$ be a partitioned hypergraph and let G be an outgoing (ingoing) clique graph of D for a part of size at most 6 such that all its outgoing (ingoing) hyperarcs in G have tail (head) cardinality at least 3. Then G is perfect.*

Proof. W. l. o. g. consider an outgoing clique graph. Every hyperarc with tail cardinality at least 4 will be contained in every maximal clique since it will have at least one tail vertex in common with any other hyperarc in G. For each hyperarc with tail cardinality 3 there is only one possibility for a tail of a hyperarc (exactly the vertices that do not appear in the tail of this hyperarc) that is not connected to this hyperarc in G. Therefore, the complement of G is bipartite. □

Lemma 4.2.5. *Let $D = (V,A)$ be a partitioned hypergraph and let G be an outgoing (ingoing) clique graph of D for a part of size 7 such that the tails of all hyperarcs in G have tail (head) cardinality at least 3. Then G does not contain odd holes and antiholes of size 5.*

Proof. Assume that we have an outgoing clique graph and that the hyperarcs $a_1, \dots, a_l, l \geq 5$ induce a hole in G. Since no three of them can be pairwise connected by an edge, every vertex in the considered part for the clique graph may be shared by only two of them. But then

$$\sum_{i \in \{1,\dots,l\}} |T_{a_i}| \leq 2 \cdot 7 = 14 < 15 \leq 3 \cdot l \leq \sum_{i \in \{1,\dots,l\}} |T_{a_i}|,$$

which is a contradiction.

For the antihole, chose one hyperarc a_1. There must be two hyperarcs a_2, a_3 in the antihole not connected to a_1. Thus, three of the vertices from the part are contained in T_{a_1}, and the remaining four vertices must be either covered by T_{a_2} and T_{a_3}, or $T_{a_2} = T_{a_3}$ and one vertex remains. Now, we need two more hyperarcs a_4, a_5 for the antihole such that a_4 and a_5 are not connected and they are not connected to both a_2 and a_3. But this is impossible since we have at most five vertices in the part which are not contained in the tails of both a_2 and a_3 and we need at least $3 + 3 = 6 > 5$ vertices. □

Remark 4.2.6. Antiholes of size 7 or larger are possible in this setting. An example is given in figure 4.1

Lemma 4.2.7. *Let $D = (V,A)$ be a partitioned hypergraph and let G be an outgoing (ingoing) clique graph of D for a part of size 7 such that its outgoing (ingoing) hyperarcs have tail (head) cardinality at least 3. Then G does not contain antiholes of size 7 which contain a hyperarc with tail (head) cardinality 4.*

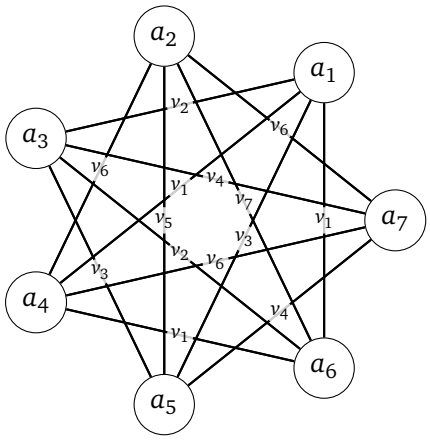


Figure 4.1: Let $P = \{v_1, \ldots, v_7\}$ be a part of some partitioned hypergraph and let $a_1, \ldots, a_7$ be the outgoing hyperarcs of P with pairwise disjoint heads and $T_{a_1} = \{v_1, v_2, v_3\}$, $T_{a_2} = \{v_5, v_6, v_7\}$, $T_{a_3} = \{v_2, v_3, v_4\}$, $T_{a_4} = \{v_1, v_6, v_7\}$, $T_{a_5} = \{v_3, v_4, v_5\}$, $T_{a_6} = \{v_1, v_2, v_7\}$, $T_{a_7} = \{v_4, v_5, v_6\}$. Then, the outgoing clique graph of P is an antihole of size 7. The vertex names on the edges give one reason for the edge.

Proof. Again, we regard an outgoing clique graph, and a hyperarc a_1 with tail cardinality 4. For an antihole of size 7 there must be two hyperarcs a_2, a_3 in the antihole not connected to a_1. The two hyperarcs must have equal tail sets since there are only three vertices left which are not covered by the tail of a_1. If a_4 and a_5 are other hyperarcs which cannot be connected to a_2 or a_3, respectively, they have to overlap in at least two vertices in the tail. But then the last two hyperarcs in the antihole have to overlap, since each of them can be connected to only one of the hyperarcs a_4 and a_5 in the conflict graph and there are only five vertices left which they can use. But this is not allowed. □

Remark 4.2.8. Antiholes of size 9 or larger are possible. Let $P = \{v_1, \ldots, v_9\}$ be a part of some partitioned hypergraph and let $a_1, \ldots, a_9$ be the outgoing hyperarcs of P with apart from H_{a_2} and H_{a_8}, H_{a_3} and H_{a_9} pairwise disjoint heads and

$$
\begin{aligned}
T_{a_1} &= \{v_1, v_2, v_3, v_4\},\\
T_{a_2} &= \{v_5, v_6, v_7\},\\
T_{a_3} &= \{v_2, v_3, v_4\},\\
T_{a_4} &= \{v_1, v_5, v_7\},
\end{aligned}
$$

$$T_{a_5} = \{v_2, v_4, v_6\},$$
$$T_{a_6} = \{v_1, v_3, v_7\},$$
$$T_{a_7} = \{v_4, v_5, v_6\},$$
$$T_{a_8} = \{v_1, v_2, v_3\},$$
$$T_{a_9} = \{v_5, v_6, v_7\}.$$

Then, the outgoing clique graph of P is an antihole of size 9.

Lemma 4.2.9. *Let $D = (V,A)$ be a partitioned hypergraph and let G be an outgoing (ingoing) clique graph of D for a part of size at most 7 such that its outgoing (ingoing) hyperarcs have tail (head) cardinality at least 3. Then every hyperarc in G with tail (head) cardinality 5 is contained in every maximal clique.*

Proof. There are at most seven vertices in the part and thus tails of a hyperarc with cardinality at least 5 will intersect the tails of every hyperarc with tail cardinality 3 or higher. □

We can summarize the previous lemmas as

Theorem 4.2.10. *Let $D = (V,A)$ be a partitioned hypergraph for $d \leq 7$. Then one of the following holds.*

1. *A maximal clique in the conflict graph of D can be found in polynomial time.*
2. *There is an odd antihole of size at least 9 in an outgoing (ingoing) clique graph of D which consists only of hyperarcs with tail (head) cardinality at least 3, one of which has size 4.*
3. *There is an odd antihole of size at least 7 in an outgoing (ingoing) clique graph of D which consists only of hyperarcs with tail (head) cardinality 3.*

4.3 Correctness of Configuration Formulations

We will now proof that (HAP_P$^{\text{dir}}$) and the stronger (HAP_P) is also a correct formulation for HAP. In the next section we will show that its LP relaxations are smaller than the one of (HAP) or the equivalent (HAP_SP).

The main idea is

Lemma 4.3.1. *Let $D = (V,A)$ be a partitioned hypergraph and dir $\in \{out, in\}$. Then, for every hyperassignment H of D, there exists a set $\mathscr{C}' \subseteq \mathscr{C}^{dir}$ of configurations such that for all $a \in A$ $|\{C \in \mathscr{C}' : a \in C\}| = |H \cap \{a\}|$, i. e., C' is a set of configurations whose disjoint union is exactly H.*

Proof. By the definition of a hyperassignment, $\mathscr{C}' := \{C'_i : i \in \{1,\dots,w\}\}$ with $C'_i = \delta^{\text{dir}}_H(P_i)$ is a subset of H and the required condition holds by construction. □

This easily implies

Theorem 4.3.2. *Given a directed hypergraph $D = (V,A)$ and a cost function $c_A : A \to \mathbb{R}$, there are bijections between the feasible solutions of* (HAP_P$^{\text{dir}}$) *and* (HAP_P)*, and hyperassignments of D. The optimum values of* (HAP_P$^{\text{dir}}$) *and* (HAP_P) *are equal to the cost of the minimum cost hyperassignment in D w. r. t. c_A if it exists and to ∞ otherwise.*

Proof. Since the projection $\pi_x^{\text{P}^{\text{dir}}}(x, y^{\text{dir}})$ or $\pi_x^{\text{P}}(x, y^{\text{out}}, y^{\text{in}})$ of every feasible solution (x, y^{dir}) or $(x, y^{\text{out}}, y^{\text{in}})$ of (HAP_P$^{\text{dir}}$) or (HAP_P), respectively, to its x-values is a feasible solution of (HAP_SP) and the objective functions are the same, the previous lemma implies that (HAP_P$^{\text{dir}}$) and (HAP_P) are a correct formulations for HAP: For $C \in \mathscr{C}'$ set $y^{\text{dir}}_C = 1$ if $C \in \mathscr{C}'$ and $y^{\text{dir}}_C = 0$ otherwise and use the same x-values as in (HAP_SP). □

Remark 4.3.3. The set of feasible solutions of (HAP_P) is the intersection of the feasible solutions of (HAPP$^{\text{out}}$) and (HAPP$^{\text{in}}$).

The sets of feasible solutions of the LP relaxations of (HAP_P$^{\text{out}}$) and (HAP_P$^{\text{in}}$) can be different. Figure 4.2 shows an example where (HAP_P$^{\text{out}}$) separates a fractional solution which cannot be separated by (HAP_P$^{\text{in}}$).

4.4 The Configuration Formulation Implies all Clique Inequalities

We will proof now that all clique inequalities are implied by (HAP_P), which is implied by the slightly more precise next theorem in connection with Remark 4.3.3.

Theorem 4.4.1. *Let $D = (V,A)$ be a partitioned hypergraph and $Q \subseteq A$ a clique in the outgoing or ingoing conflict graph of D. Then,*

$$\sum_{a \in Q} x_a \leq 1$$

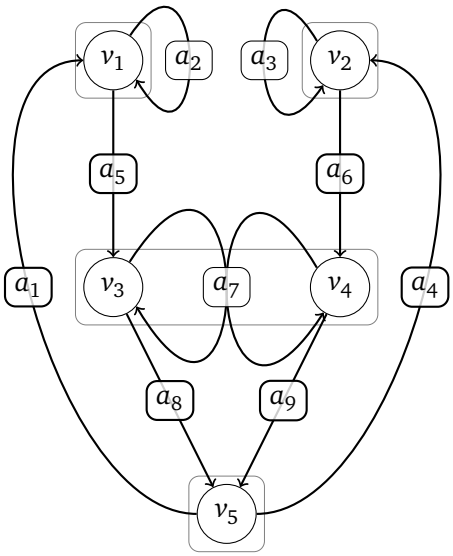


Figure 4.2: $D = (V,A)$, $V = \{v_1, \dots v_5\}$, $A = \{a_1, \dots a_7\}$ is a partitioned hypergraph with parts $\{v_1\}$, $\{v_2\}$, $\{v_3, v_4\}$, $\{v_5\}$. The LP relaxation of (HAP_P$^{\text{in}}$) for this directed hypergraph has the feasible solution with value 0.5 an all hyperarcs. (HAP_P$^{\text{out}}$) and (HAP_P) do not have it, because $\{v_3, v_4\}$ does not have any outgoing partitions containing a_8 or a_9 and therefore these two hyperarcs cannot have a positive value in a feasible solution of the LP relaxation of a formulation with outgoing configurations.

is a valid inequality for every feasible solution of the LP relaxation of (HAP_P^{out}) or (HAP_P^{in}), respectively.

Proof. By Lemma 4.2.1, $Q \subseteq \delta^{\text{dir}}(P_i)$ for some part P_i and the chosen direction.

First of all, observe that $|Q \cap C| \leq 1$ for every $C \in \mathscr{C}^{\text{dir}}$, because the tails and heads of two hyperarcs from $C \in \mathscr{C}^{\text{dir}}$ are disjoint, but never both the tails and heads of two hyperarcs from Q are disjoint.

Now, let v be some vertex in P_i. Then, by (HAP_P)(ii)

$$1 = \sum_{a \in \delta^{\text{dir}}(v)} x_a,$$

which is the sum of the right hand sides of (HAP_P)(i) for $a \in \delta^{\text{dir}}(v)$, and considering the left hand sides we get

$$= \sum_{C \in \mathscr{C}^{\text{dir}}} |\delta^{\text{dir}}(v) \cap C| \cdot y_C^{\text{dir}}$$

and, since every $a \in \delta^{\text{dir}}(v)$ is contained in P_i and exactly one such a is contained in every $C \in \mathscr{C}_i^{\text{dir}}$, we get

$$= \sum_{C \in \mathscr{C}_i^{\text{dir}}} y_C^{\text{dir}}$$

and the observation from above implies

$$= \sum_{a \in Q} \sum_{C \in \mathscr{C}_i^{\text{dir}} : a \in C} y_C^{\text{dir}} + \sum_{C \in \mathscr{C}^{\text{dir}} : C_i \cap Q = \emptyset} y_C^{\text{dir}}$$
$$\geq \sum_{a \in Q} \sum_{C \in \mathscr{C}_i^{\text{dir}} : a \in C} y_C^{\text{dir}}$$

by (HAP_P)(iii) and, finally, applying (HAP_P)(i) again

$$= \sum_{a \in Q} x_a.$$

□

4.5 Other Properties of the LP Relaxation of the Configuration Formulation

First of all, the set of feasible solutions of the configuration formulation (HAP_P) is not integral. For an example see figure 4.3

Moreover, the example shows that also the projections of set P_{LP}(HAP_P) of feasible solutions of the LP relaxation of (HAP_P) to the variables for all outgoing or ingoing hyperarcs of one part are not always integral. However, there are cases when they are integral.

Theorem 4.5.1. *Let $D = (V,A)$ be a partitioned hypergraph and let P_i be some part such that for every configuration $C \in \mathscr{C}_i^{dir}$ there is a (possibly non-integral) solution of the LP relaxation of* (HAP_P$^{\text{dir}}$) *where $y_C = 1$. Then,*

$$\pi^{P^{dir}}_{x^{P_i}}(P_{LP}(HAP\_P^{dir}))$$

is integral.

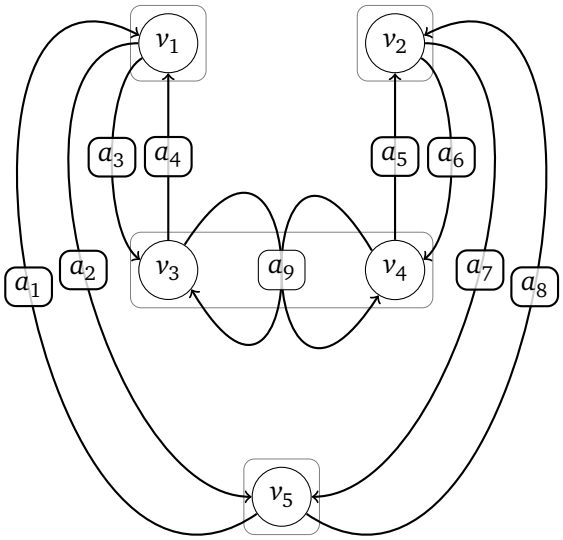


Figure 4.3: $D = (V,A)$, $V = \{v_1, \dots v_5\}$, $A = \{a_1, \dots a_9\}$ is a partitioned hypergraph with parts $\{v_1\}$, $\{v_2\}$, $\{v_3, v_4\}$, $\{v_5\}$. The LP relaxation of (HAP_P) (and also the LP relaxation of (HAP)) has only one feasible solution for this directed hypergraph. It has value 0.5 an all hyperarcs: Because of v_3 and v_4, $x_{a_3} = x_{a_4} = x_{a_5} = x_{a_6}$. Then, because of v_1 and v_2, $x_{a_1} = x_{a_2} = x_{a_7} = x_{a_8}$. Finally, because of v_5, we get the value 0.5 on all hyperarcs, and it is possible to find a set configurations such that the variables of these configurations have value 0.5 and the solution is feasible for the LP relaxation of (HAP_P).

Proof. Let (x, y^{dir}) be a feasible solution of the LP relaxation of (HAP_P$^{\mathrm{dir}}$). For $C \in \mathscr{C}_i^{\mathrm{dir}}$, define $x(C) \in \mathbb{R}^{\delta^{\mathrm{dir}}(P_i)}$ by

$$x(C)_a = \begin{cases} 1 & \text{if } a \in C \\ 0 & \text{otherwise} \end{cases}.$$

Then, by (HAP_P$^{\mathrm{dir}}$) (i)

$$(x_a)_{a \in \delta^{\mathrm{dir}}(P_i)} = \sum_{C \in \mathscr{C}_i^{\mathrm{dir}}} y_C^{\mathrm{dir}} \cdot x(C). \tag{4.1}$$

Now, if for every $C \in \mathscr{C}_i^{\mathrm{dir}}$ there is a feasible solution $(x', y'^{\mathrm{dir}}) \in P_{\mathrm{LP}}(\mathrm{HAP\_P}^{\mathrm{dir}})$ of the LP relaxation of (HAP_P$^{\mathrm{dir}}$) with $y'^{\mathrm{dir}}_C = 1$, i. e., $y'^{\mathrm{dir}}_{C'} = 0$ for $C' \neq C$ and $(x'_a)_{a \in \delta^{\mathrm{dir}}(P_i)} = x(C)$, then by (4.1) $\pi^{\mathrm{P}^{\mathrm{dir}}}_{x^{P_i}}(x, y^{\mathrm{dir}})$ for $(x, y^{\mathrm{dir}}) \in P_{\mathrm{LP}}(\mathrm{HAP\_P}^{\mathrm{dir}})$ is

a linear combination of integral projections $\pi^{\mathrm{P^{dir}}}_{x^{P_i}}(x', y'^{\mathrm{dir}})$ of feasible solutions of (HAP_P$^{\mathrm{dir}}$) to $\{x_a : a \in \delta^{\mathrm{dir}}(P_i)\}$ and thus $\pi^{\mathrm{P^{dir}}}_{x^{P_i}}(P_{\mathrm{LP}}(\mathrm{HAP\_P^{dir}}))$ is integral. □

Still, the configuration formulation implies much more than the clique inequalities. It implies all valid inequalities one can know from looking at the outgoing or all the ingoing hyperarcs of one part.

Theorem 4.5.2. *Let $D = (V,A)$ be a partitioned hypergraph, let P_i be some part and let*

$$(dir, rid) \in \{(out, in), (in, out)\}.$$

Then, $\pi^{P^{dir}}_{x^{P_i}}(P_{LP}(HAP\_P^{dir}))$ is a subset of the convex hull X of

$$\{x \in \mathbb{Z}^{\delta^{dir}(P_i)} : \sum_{a \in \delta^{dir}(v)} x_a = 1 \quad \forall v \in P_i,$$
$$\sum_{a \in \delta^{rid}(v) \cap P_i} x_a \leq 1 \quad \forall v \in V \setminus P_i\}.$$

Proof. The proof works similarly to the proof of the last theorem. (4.1) implies that every point in $\pi^{\mathrm{P^{dir}}}_{x^{P_i}}(P_{\mathrm{LP}}(\mathrm{HAP\_P^{dir}})$ is the linear combination of points $x(C) \in X$, since $x(C)$ has only integral coordinates 0 or 1 and satisfies by the definition of a configuration the equalities and inequalities defining X. □

The last theorem implies Theorem 4.4.1, which we stated separately because of its importance for the practical application.

Figure 4.4 shows an example of a fractional solution of the LP relaxation of (HAP) which cannot be separated by a clique inequality but cannot appear as a solution of the LP relaxation of (HAP_P).

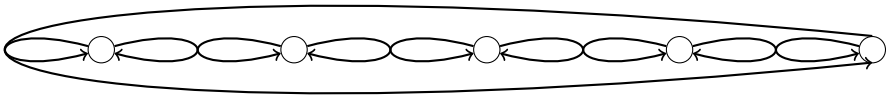


Figure 4.4: For $d \geq 5$ partitioned hypergraph with five vertices, all in one part, and five hyperarcs. The fractional solution where all hyperarcs have value 0.5 is feasible for the LP relaxation of (HAP), but not feasible for the LP relaxation of (HAP_P) although it cannot be separated by a clique inequality.

Chapter 5

Solving the Configuration LP

Summary of this chapter *Firstly, we formulate the pricing subproblem for* (HAP_P)*, which one has to solve to find the nonbasic configuration variable with the least reduced cost for some basic solution. We propose two algorithms to solve it. One of them solves at first a shortest path problem to find a structure, a so-called relaxed configuration, which in general is not necessarily a configuration, but already is a configuration for special instances arising from vehicle rotation planning problems. If relaxed configurations are not enough, the algorithm has to be applied many times. The other algorithm we propose can only be applied to a special case and employs minimum cost flow techniques in a directed graph.*

5.1 Pricing Subproblem

The number or configurations and thus also the number of variables in the configuration ILP (HAP_P) can be (although it is at most $2w|A|^d$—for each of the two directions and each part at most d hyperarcs from A can be selected—and thus polynomial for constant d) very high. Therefore, a possibility to solve the linear programming relaxation of the configuration ILP without really enumerating all configurations is useful. We want to find a fast algorithm which, given a basic feasible solution of (HAP_P), finds the nonbasic variable with the least reduced cost. This can be used when solving the LP relaxation of (HAP_P) with a column generation algorithm.

Let $(x, y^{\text{out}}, y^{\text{in}})$ be some basic feasible solution of the linear programming relaxation of (HAP_P) and let

$$\pi \in \mathbb{R}^{A\times\{\text{out,in}\}} \times \mathbb{R}^{V\times\{\text{out,in}\}}$$

be the corresponding solution of the dual problem, such that $\pi_{(a,\text{dir})}$ is the dual

variable for constraint (HAP_P) (i) for a and dir and $\pi_{(v,\text{dir})}$ is the dual variable for constraint (HAP_P) (ii) for v and dir.

The coefficient of $y_C^{\text{dir}}, C \in \mathscr{C}^{\text{out}} \cup \mathscr{C}^{\text{in}}$ in the objective function of (HAP_P) is 0. y_C^{dir} has always coefficient 1 in the constraints (HAP_P) (i) for $a \in C$ and dir, where it only appears. Thus, the reduced cost of $y_C, C \in \mathscr{C}^{\text{dir}}$ is given by

$$-\sum_{a \in C} \pi_{(a,\text{dir})}.$$

Therefore, the task to find the outgoing or ingoing configuration variable y_C^{dir}, $C \in \mathscr{C}^{\text{dir}}$, dir $\in \{\text{out}, \text{in}\}$ with the least reduced cost means to select the following: a set of hyperarcs with pairwise disjoint tails and heads such that the disjoint union of their tails or heads is a part, which has the minimum cost w. r. t. the cost function $c'_A(a) := \pi_{(a,\text{out})}$ or $c'_A(a) := \pi_{(a,\text{in})}$, respectively.

We will do this for each of the two directions separately and select the configuration variable with the least reduced cost from these two solutions afterwards. For an easier notation, we will assume that we are looking for outgoing configurations. Of course, it works in the same way for the other direction.

5.2 Algorithm using Shortest Paths

5.2.1 Shortest Path Formulation for a Relaxation

To begin with, we will consider a relaxed situation by omitting the requirement that the heads of the hyperarcs in a configuration have to be pairwise disjoint. So, what we are searching for now is a subset of the hyperarcs such that the disjoint union of their tails is

$$P_i = \{v_i^1, \ldots, v_i^{d_i}\}$$

for some $i \in \{1, \ldots, w\}$. Let us call such set of hyperarcs a relaxed configuration.

This can be done by solving a shortest path problem in the directed graph $G = (X, J)$. The set of vertices of this graph is

$$X = \left\{(i, \{v_i^1\} \cup Y) : i \in \{1, \ldots, w\}, Y \subseteq \{v_i^2, \ldots, v_i^{d_i}\}\right\} \cup \{s, t\}.$$

It describes, besides a source s and sink t, the set of all possible hyperarc tail unions $\left\{\{v_i^1\} \cup Y : Y \subseteq \{v_i^2, \ldots, v_i^{d_i}\}\right\}$, which can arise if we successively add a hyperarc which contains the not already covered vertex with the least upper index from the part which we want to cover in its tail to get to a relaxed

configuration. The first coordinate is the index of the part we are building successively and the second coordinate is the set of upper vertex indices of those vertices which are already covered by hyperarc tails. Since in the beginning of this procedure one always has to select the hyperarc which covers some v_i^1, this vertex is always inside this set. It holds that

$$|X| = \sum_{i=1}^{w} 2^{d_i - 1} + 2 \leq 2^{d-1} w + 2.$$

The arcs of G are

$$J = \{\Big((i, \{v_i^1, \ldots, v_i^{d_i}\}), t\Big) : i \in \{1, \ldots, w\}\} \cup \\ \Big\{\big(s, (i, T_a)\big) : i \in \{1, \ldots, w\}, a \in \delta^{\text{out}}(P_i), v_i^1 \in T_a\Big\} \cup \bigcup_{a \in A} J(a)$$

where

$$J(a) = \Big\{\big((i, J_1), (i, J_2)\big) : a \in \delta^{\text{out}}(P_i), J_2 = J_1 \cup T_a, v_i^k \in T_a\Big\},$$

with $k = \min\{k \in \{1, \ldots, d_i\} : v_i^k : v_i^k \notin J_1\}$. Firstly, we have the w arcs from vertices in G for complete relaxed configurations to t. They could be omitted, if we just identify these vertices with t, but this would make the notation more complicated.

Each of the other arcs can be associated with some hyperarc $a \in A$ of the partitioned hypergraph D. They describe from which vertex to which vertex in G we can get if we add the associated hyperarcs using the procedure from above.

This structure implies that G does not contain any directed cycles and the set of corresponding hyperarcs $a \in A$ for arcs in an s-t-path in G corresponds to a relaxed configuration. If we use $c'_A(a)$ as the weight for the arcs which are associated to hyperarc a, we get the cost w. r. t. c'_A of the relaxed configuration as the cost of the path. The problem to find the relaxed configuration with the cost w. r. t. c'_A then translates into a shortest path problem in G, which can be solved in $O(|X| \cdot |J|)$ using for example the Bellman-Ford algorithm.

We already have calculated $|X|$. For $|J|$, this is more complicated, since the number of arcs associated to a, depends on its tail cardinality and the upper index of the first vertex in its tail. A worst case value for $|\mathrm{J}|$ is $|X| \cdot |A|$ (for each hyperarc $a \in A$ we can have at most one element in $|\mathrm{J}|$ starting in some vertex from $|\mathrm{X}|$). For the expected value see the next section.

5.2.2 Expected Complexity

For $i \in \{1, \ldots, w\}$, let us assume that each of the $2^{d_i} - 1$ possible tails of outgoing hyperarcs of P_i appears with the same probability and calculate the expected number E_i of associated arcs in G for such a hyperarc.

How many different tails can there be if the tail cardinality $|T_a|$ and the vertex with the minimum upper index k_a of vertices in the tail are fixed? We can choose the remaining $|T_a| - 1$ vertices in the tail from $d_i - k_a$ vertices in the part with higher upper index than k_a, so there are $\binom{d_i - k_a}{|T_a|-1}$ possibilities.

And how many associated arcs are there for such a hyperarc? If $k_a = 1$, it is just one—we can chose the hyperarc in our procedure only as the first hyperarc to add. Otherwise J_1 has to contain all the $k_a - 1$ vertices with upper index less than k_a, and for each of the other but the T_a vertices which are in the hyperarc tail, we can decide whether we want it to be in J_1 or not. So, we have $2^{d_i - (k_a - 1) - |T_a|}$ associated arcs.

There are $2^{d_i - 1}$ different tails which contain v_i^1, this is the case where $k_a = 1$.

Thus, we get

$$
\begin{aligned}
E_i &= \frac{1}{2^{d_i} - 1} \left(2^{d_i - 1} + \sum_{k_a=2}^{d_i} \sum_{|T_a|=1}^{d_i - k_a + 1} \binom{d_i - k_a}{|T_a| - 1} 2^{d_i - (k_a - 1) - |T_a|} \right) \\
&= \frac{1}{2^{d_i} - 1} \left(2^{d_i - 1} + \sum_{k_a=2}^{d_i} \sum_{j=0}^{d_i - k_a} \binom{d_i - k_a}{j} 2^{d_i - (k_a - 1) - (j+1)} \right) \\
&= \frac{1}{2^{d_i} - 1} \left(2^{d_i - 1} + \sum_{k_a=2}^{d_i} \sum_{j=0}^{d_i - k_a} \binom{d_i - k_a}{j} 2^{(d_i - k_a) - j} \cdot 1^j \right) \\
&= \frac{1}{2^{d_i} - 1} \left(2^{d_i - 1} + \sum_{k_a=2}^{d_i} 3^{d_i - k_a} \right) \\
&= \frac{1}{2^{d_i} - 1} \left(2^{d_i - 1} + 3^{d_i} \cdot \left(\sum_{k_a=0}^{d_i} \left(\frac{1}{3} \right)^{k_a} - \frac{1}{3} - 1 \right) \right) \\
&= \frac{1}{2^{d_i} - 1} \left(2^{d_i - 1} + 3^{d_i} \cdot \left(\frac{1 - \left(\frac{1}{3} \right)^{d_i + 1}}{\frac{2}{3}} - \frac{4}{3} \right) \right)
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{2^{d_i}-1}\left(2^{d_i-1} + 3^{d_i}\cdot\left(\frac{3}{2}\cdot\left(1-\left(\frac{1}{3}\right)^{d_i+1}\right)-\frac{4}{3}\right)\right)\\
&= \frac{1}{2^{d_i}-1}\left(2^{d_i-1} + 3^{d_i}\cdot\left(\frac{1}{6}-\frac{1}{3}\cdot 3^{-d_i}\right)\right)\\
&= \frac{1}{2^{d_i}-1}\left(2^{d_i-1} + \frac{1}{2}\cdot 3^{d_i-1}-\frac{1}{3}\right)\\
&\in O\left(1.5^{d_i}\right)
\end{aligned}
$$

For $d_i = 7$, which is the highest possible value in the vehicle rotation planning application, we get $E_i \approx 3.37$. Then, the expected running time of the Bellman-Ford algorithm is in $O(3.37 \cdot 64 \cdot |A| \cdot w)$.

5.2.3 Solving the Pricing Subproblem

There are types of partitioned hypergraphs where relaxed configurations are automatically configurations. This is, for example, the case for the directed hypergraph from vehicle rotation planning, if we allow a trip only as the next trip of a vehicle if it starts less than 24 hours after the end of the current trip. Then, no two outgoing hyperarcs of one part with disjoint tails have a vertex in their heads in common, since the arrival time of two trips in one train differs by some positive number of full days. Often it is enough to do vehicle rotation planning with this restriction, because in most cases only such hyperarcs appear in an optimum solution.

If we do not have such a situation and want to get configurations and not possibly relaxed configurations (the same formulation with relaxed configurations does not imply the clique inequalities), we can modify the method from above.

Since this is the important case for vehicle rotation planning, we describe it for hypergraphs for which the heads and tails of all hyperarcs have the same cardinality. For each of the $|V| \cdot (|V|-1) \cdot \ldots \cdot (|V|-d+1) \in O(|V|^d)$ possible d-tuples L of pairwise different vertices of the hypergraph, we apply the procedure from above with the requirement that the head of a hyperarc which covers vertices $v_i^{l_1}, \ldots, v_i^{l_r}$ with its tail is $\{L_{l_1}, \ldots, L_{l_r}\}$.

Thus, we can solve the pricing subproblem by applying the method from above $O(|V|^d)$ times (with less hyperarcs than before) and selecting the solution with the least cost in $O(|V|^d \cdot 2^{d-1} \cdot w \cdot 1.5^d \cdot |A|) = O(|V|^d \cdot w \cdot |A|)$ for constant d if the head and tail cardinalities of the hyperarcs are equal. This is for $d = 7$ a very high exponent.

5.3 Algorithm for a Special Case using Minimum Cost Flows

We will present a different algorithm now. It works only if the hyperarcs satisfy a special condition. The condition will be explained later.

Let P_i be a part and let $C \in \mathscr{C}_i^{\text{out}}$ be some outgoing configuration of P_i. Define $T(C) := \{T_a : a \in C\}$ to be the set of all tails of the hyperarcs in the configuration. Possible values of $T(C)$ are exactly all sets of disjoint subsets of P_i whose union is P_i. There are $B_{|P_i|} \leq B_d$ such sets where B_n is the n-th Bell number ([Sta00]). The first Bell numbers are ([Slo10])

$$\begin{aligned} B_0 &= 1, \\ B_1 &= 1, \\ B_2 &= 2, \\ B_3 &= 5, \\ B_4 &= 15, \\ B_5 &= 52, \\ B_6 &= 203, \\ B_7 &= 877. \end{aligned}$$

We want to solve the pricing subproblem for each possible $T(C)$ separately now. Given some $T = T(C)$, we will show that the pricing subproblem can be formulated as a minimum cost flow problem on at most $|V| + d + 2$ vertices with at most $|\delta^{\text{out}}(P_i)|$ arcs, a flow value of at most d and capacities all equal to one. $|T|$ is at most d.

Basically, we will group the vertices such that each outgoing hyperarc of P_i becomes an arc and the hyperflow problem, which is a possible formulation of the pricing subproblem, becomes a flow. The hyperarcs used for the underlying hyperflow problem will be the ones whose tails are elements of T. Let us call this set of hyperarcs A_T.

We now explain when the algorithm is applicable. This is the case if it is possible to introduce an equivalence relation $\sim$ on A_T such that $H_a \cap H_b \neq \emptyset$ implies $a \sim b$ and $a \sim b$ implies $T_a \cap T_b \neq \emptyset$ or $H_a \cap H_b \neq \emptyset$.

Let V_T be the set of the equivalence classes under this relation. $|V_T|$ is at most $|V|$. Now, for every hyperarc in $a \in A_T$ there is exactly one element in T equal to T_a and exactly one element $V_T(a)$ in V_T which contains hyperarcs that have nonempty head intersection with a. Two hyperarcs from A_T can be used simultaneously in a configuration if and only if their corresponding elements in T and V_T are both different.

Therefore, a configuration C with the minimum reduced costs for a given P_i and $T = T(C)$ is the set of all hyperarcs a such that

$$(T_a, V_T(a)) \in E_{A_T} := \{(T_a, V_T(a)) : a \in A_T\} \subseteq T \times V_T$$

has value one in a optimal solution to the minimum cost flow problem in the directed graph

$$\left(T \cup V_T \cup \{s,t\}, E_{A_T} \cup \{(s,v) : v \in T\} \cup \left\{(v,t) : v \in V_T\right\}\right)$$

with all arc capacities equal to one, costs $c'_A(a)$ for e_a and all other arc costs zero sending $|T|$ units of flow from s to t.

Thus, if the hypergraph satisfies the condition from above, we can solve the pricing subproblem by solving at most $w \cdot B_d$ minimum cost flow problems on $O(|V| + d)$ vertices each and at most $B_d \cdot |A|$ arcs altogether and selecting the configuration with the least reduced cost from all these. If the condition is satisfied only for some parts, we can use this algorithm for these parts and the one from the previous section for the others.

Chapter 6

Summary

Summary of this chapter *We sum up the theoretical results of this thesis and give practical conclusions.*

6.1 Theoretical Results

We introduced a novel concept of hyperassignments in directed hypergraphs that is motivated by regularity requirements in vehicle rotation planning. Such vehicle rotation planning problems can be modeled as hyperassignment problems (HAP).

We studied the complexity of HAP. We began with transformations from HAP to SPP and vice versa. We proved that HAP is $\mathcal{NP}$-hard, even with several restrictions like graph based hypergraphs with equal head and tail cardinality at most two, and partitioned hypergraphs with equal head and tail cardinality at most three. We also showed that the determinants of the basis matrices and the integrality gap in the canonical ILP formulation of HAP can become arbitrarily large.

Furthermore, we were concerned with cliques in the conflict graphs associated with partitioned hypergraphs. We gave a characterization of the subgraphs of the conflict graph, which contain all maximal cliques. We then introduced a novel extended ILP formulation of HAP, proved its correctness, and showed that all clique inequalities hold for the feasible solutions of its LP relaxation.

Finally, we proposed efficient algorithms to solve the pricing subproblem for the extended formulation without resorting to an enumeration of the large number of variables there.

6.2 Practical Conclusions

Tests with real data implied that adding all clique inequalities to the canonical ILP formulation of HAP leads to very small gaps between the optimum value of the ILP and the LP, while they can be large otherwise. Our novel extended formulation using configurations implies all the clique inequalities and also other cuts which separate fractional solutions of the LP relaxation of our first formulation.

At least for the practically relevant case where we allow the break between subsequent trips of a vehicle to be at most one day long, we can deal well with the large number of variables in the configuration formulation. We gave a pricing algorithm for this case which consists of solving two shortest path problems on $64w + 2$ vertices.

List of Figures

Bibliography

[ADS84] Gabriele Ausiello, Alessandro D'Atri, and Domenico Saccà. Minimal representations of directed hypergraphs and their application to database design. In *Algorithm design for computer system design*, pages 125–157. Springer-Verlag New York, Inc., New York, NY, USA, 1984.

[Bor98] Ralf Borndörfer. *Aspects of Set Packing, Partitioning, and Covering*. Shaker Verlag, Aachen, 1998. Ph.D. thesis, Technische Universität Berlin.

[CDP04] Sanjay Chawla, Joseph G. Davis, and Gaurav Pandey. On local pruning of association rules using directed hypergraphs. In *International Conference on Data Engineering*, page 832, 2004.

[CGS92] Riccardo Cambini, Giorgio Gallo, and Maria G. Scutellà. Minimum cost flows on hypergraphs. Technical report, Department of Computer Sciences, University of Pisa, 1992.

[Chv83] Vašek Chvátal. *Linear Programming*. W. H. Freeman, 1983.

[CRST06] Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The strong perfect graph theorem. *Annals of Mathematics*, 164:51–229, 2006.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman, January 1979.

[GLS88] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988.

[Grö09] Martin Grötschel. Graphen- und Netzwerkalgorithmen, 2009. Lecture notes, `http://www.zib.de/groetschel/teaching/SS2009/ADMI-Skript.pdf`.

[Grö10] Martin Grötschel. Lineare und Ganzzahlige Programmierung, 2010. Lecture notes, `http://www.zib.de/groetschel/teaching/WS0910/skriptADMII-WS0910neu.pdf`.

[GS98] Giorgio Gallo and Maria Scutellà. Directed hypergraphs as a modelling paradigm. *Decisions in Economics and Finance*, 21:97–123, 1998.

[JMRW92] Robert G. Jeroslow, Kipp Martin, Robert L. Rardin, and Jinchang Wang. Gainfree Leontief substitution flow problems. *Math. Program.*, 57(3):375–414, 1992.

[KNO+03] L. Krishnamurthy, J. Nadeau, G. Ozsoyoglu, M. Ozsoyoglu, G. Schaeffer, M. Tasan, and W. Xu. Pathways database system: an integrated system for biological pathways. *Bioinformatics*, 19(8):930–937, May 2003.

[MN98] Patrice Marcotte and Sang Nguyen. Hyperpath formulations of traffic assignment problems. In Patrice Marcotte and Sang Nguyen, editors, *Equilibrium and Advanced Transportation Modelling*, pages 175–199. Kluwer Academic Publishers, 1998.

[MS07] Amadís Antonio Martínez Morales and María Esther Vidal Serodio. A directed hypergraph model for RDF. In *KWEPSY*, 2007.

[RSC97] Mysore Ramaswamy, Sumit Sarkar, and Ye-Sho Chen. Using directed hypergraphs to verify rule-based expert systems. *IEEE Transactions on Knowledge and Data Engineering*, 9(2):221–237, 1997.

[Slo10] Neil J. A. Sloane. The on-line encyclopedia of integer sequences, 2010. Sequence A000110, `http://www.research.att.com/~njas/sequences/A000110`.

[Sta00] Richard P. Stanley. *Enumerative Combinatorics: Volume 1*. Cambridge University Press, June 2000.

[WLW08] Gang Wu, Juan-Zi Li, and Kehong Wang. System Π: a hypergraph based native RFD repository. In *WWW*, pages 1035–1036, 2008.

Index