



Optimale Umplanung in der kundenindividuellen Massenproduktion

Masterarbeit
in
Informatik

vorgelegt von
Markus Kruber

im März 2015

Priv.-Doz. Dr. Walter Unger
Lehrstuhl für Informatik 1
Algorithmen und Komplexität
RWTH Aachen University

Prof. Dr. Marco Lübbecke
Lehrstuhl für Operations Research
RWTH Aachen University

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, dass alle Stellen der Arbeit, die wörtlich oder sinngemäß aus anderen Quellen übernommen wurden, als solche kenntlich gemacht sind und dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegt wurde.

Aachen, den 30. März 2015

Markus Kruber

Inhaltsverzeichnis

1	Einleitung und Motivation	1
2	Datengrundlage	3
2.1	Spezifikation	3
2.2	Zeitpunkte	4
2.3	Zusätzliche Informationen	5
3	Data Mining Methoden	7
3.1	Clustering	7
3.1.1	Partitionierende Clusterverfahren	7
3.1.2	Dichtebasierte Clusterverfahren	9
3.2	Assoziationsanalyse	11
3.2.1	Häufige Attribute	12
3.2.2	Assoziationsregeln [A93]	14
3.3	Klassifikation	16
3.3.1	Entscheidungsbäume [Q86]	16
3.3.2	Entropy [E48]	18
3.3.3	Verzweigung	19
3.3.4	Diskussion	20
4	Datenanalyse	22
4.1	Statistische Analyse	22
4.2	Qualitätsmaß	24
4.3	Entscheidungsbäume	26
5	Das Auto Planungs Modifikations Problem	30
5.1	Konstruktion des Übergangsgraphen	30
5.2	Modellierung des APMP	34
5.3	Modell- und Grapherweiterungen	36
5.3.1	Ähnlichkeitsanforderung	36
5.3.2	Ähnlichkeitsbeschränkung für den Zielknoten	37
5.3.3	Multikriterielle Zielfunktion	38
5.3.4	Korrelation zwischen Attributen	40
5.3.5	Lager	41
5.4	Übersicht	42

Inhaltsverzeichnis

5.5	Alternative Formulierung	45
5.6	Resultate	48
5.6.1	Laufzeit	50
5.6.2	Einzelergebnisse	51
5.6.3	Gesamtergebnis	52
6	Komplexitätsanalyse	55
6.1	Komplexitätsklassen P und NP	55
6.2	Komplexität des Auto Planungs Modifikations Problems	56
6.2.1	Das Hyper Binary Integer Programming Problem	57
6.2.2	Reduktion auf APMP	61
6.3	Einschränkungen an das APMP	64
7	Zusammenfassung & Ausblick	67
	Literaturverzeichnis	71

1 Einleitung und Motivation

Die in der Automobilindustrie vorhandene hohe Produktvielfalt und gleichzeitige *Just-In-Time* Produktion machen eine flexible Reaktion auf Kundenanfragen zu einem relevanten Problem. Die Produktionsplanung ist aufgrund der nahezu lagerlosen Fertigung und den langen Lieferzeiten von einigen Autoteilen für mehrere Wochen im Voraus geplant. Insgesamt umfasst die Produktion Autos, die aus einer Kundenbestellung resultieren, und abhängig von Markt und Modell zu einem Anteil von 80%-90% aus Autos, die nicht bereits ab Werk verkauft sind. Ein noch unverkauftes, geplantes Auto wird freie Order bezeichnet und bieten den Handlungsspielraum für eventuelle Veränderungen.

Durch Markt- und Datenanalysen wird seitens der Automobilindustrie versucht eintreffende Kundenwünsche vorherzusagen. Ein unmögliches Vorhaben für Tausende Konfigurationen, die sich der Kunde zum Teil selber zusammenstellen kann. Um dennoch flexibel auf Kundenwünsche reagieren zu können, wird in dieser Arbeit die Idee verfolgt, durch kleinere Veränderungen an freien Ordnern das vom Kunden gewünschte Auto zu einem möglichst frühen Zeitpunkt der Produktionsplanung hinzuzufügen. Der resultierende Ein- und Ausbau von Ausstattungsmerkmalen findet dabei als Tausch mit anderen geplanten Autos statt, sodass es keine Notwendigkeit für eine Lagerhaltung gibt. Durch eine erfolgreiche Umplanung kann ein ursprünglich geplantes, aber noch unverkauftes Auto an den Kunden verkauft und gleichzeitig die Wartezeit für den Kunden reduziert werden.

Der Automobilhersteller Ford besitzt für die Organisation seiner Produktion eine Datenbank, genannt Orderbank, in welcher alle relevanten Daten, wie zum Beispiel der Produktionszeitpunkt oder die Liste der zu verbauenden Autoteile, gespeichert sind. Konfiguriert ein Kunde beim Händler oder privat sein Wunschauto, wird dieses zuerst im Lager der Händler oder in der Orderbank innerhalb der freien Orders gesucht und falls gefunden zugeteilt. Aufgrund der erwähnten Produktvielfalt tritt dieser Fall selten auf, sodass eine neue Order für die Orderbank generiert werden muss. Damit verbunden ist, in Abhängigkeit von den Lieferzeiten der Autoteile und freien Produktionskapazitäten, eine hohe Wartezeit für den Kunden.

Die konkrete Problemstellung sieht vor, basierend auf einer vorhandenen Produktionsplanung, Umplanungen vorzunehmen, sodass neben dem Wunschauto bezüglich der Verkaufbarkeit hochwertige Autos entstehen, ohne dass Autoteile gelagert

1 Einleitung und Motivation

oder aus dem Lager entnommen werden müssen. Zusätzlich herrschen bei den möglichen Veränderungen, die im Zuge einer Umplanung vorgenommen werden, Restriktionen, da ähnliche Autos, zu den ursprünglich geplanten, gebaut werden sollen und nicht alle Kombinationen von Autoteilen möglich sind.

Insgesamt soll das angeforderte Auto unter mehreren zu optimierenden Aspekten realisiert werden. Im Allgemeinen soll der Produktionszeitpunkt möglichst früh sein, um die Wartezeit auf das Auto zu minimieren. Gleichzeitig soll die Veränderung innerhalb der Orderbank minimiert werden. Damit sind sowohl die Anzahl der zu verändernden Autos, als auch der Grad der Veränderung an einem spezifischen Auto mitinbegriffen. Final gilt es durch Marktanalysen die Qualität eines Autos in Form seiner Verkaufbarkeit zu bestimmen und somit im Falle einer Umplanung die Qualität der Autos zu maximieren.

Außerdem sind an einen möglichen Lösungsprozess weitere, laufzeittechnische Anforderungen gestellt. Eine Antwort in Form einer Umplanung inklusive eines Lieferdatums müssen in wenigen Sekunden berechenbar sein. Genau wie der Preis gehört das Lieferdatum zu den Daten, die dem Kunden für eine fundierte Entscheidung nach der Konfiguration seines Autos bereitgestellt werden müssen. Um die Vielzahl an realistischen Einschränkungen und die Laufzeitrestriktionen vereinbaren zu können, werden, falls möglich, einzelne Prozesse als Preprocessing ausgelagert.

In Kapitel 2 werden die von Ford bereitgestellten Daten erklärt und deren Vor- und Nachteile aufgelistet. Kapitel 3 gibt einen Einblick in das weitreichende Thema Data Mining und beschränkt sich dabei auf Methoden, die in Kapitel 4 auf die vorhandenen Daten angewendet werden können. Mittels der gewonnenen Kenntnisse und linearer ganzzahliger Programmierung werden in Kapitel 5 Modelle aufgestellt, deren Lösungen Antwort auf das Auto Planungs Modifikations Problem geben. Neben der empirisch getesteten Laufzeit des Lösungsprozesses werden in Kapitel 6 die Komplexität des Problems und eingeschränkter Versionen bestimmt. Abschließend werden in Kapitel 7 eine Zusammenfassung über die vollständige Arbeit gegeben, vorhandene Restriktionen im Bezug zur Realität aufgelistet und ein Eindruck vermittelt, durch welche Ergänzungen und Zusatzinformationen weitere Verbesserungen möglich sind.

Im Verlauf der Arbeit wird der Prozess der Umplanung des Öfteren als Umbau bezeichnet. Damit ist zu keiner Zeit der Umbau eines bereits gebauten Autos gemeint, sondern die Änderung von freien Orders innerhalb der Produktionsplanung.

2 Datengrundlage

Innerhalb des Unternehmens Ford werden Autos aufgrund von verschiedenen Anforderungen auf mehrere Arten kodiert und dargestellt. Im Laufe dieser Arbeit wird lediglich eine Kodierung, die sogenannten *SALES-Codes*, verwendet und betrachtet.

Es existieren zwar deutlich detailreichere Kodierungen, allerdings ist im Verlauf dieser Arbeit zu erkennen, dass die SALES-Codes alle Informationen für eine adäquate Behandlung der Problemstellung zur Verfügung stellen.

Bei den SALES-Codes handelt es sich um Produktions- und Verkaufsdaten von allen Autos, die Ford in den letzten Jahren produziert hat und in den nächsten Wochen zu produzieren geplant hat. Für jedes Auto existiert genau ein Eintrag in der Datenbank, der durch die *VIN* (engl. vehicle identification number) eindeutig gekennzeichnet ist.

Welche Teile in einem Auto verbaut sind bzw. werden wird hauptsächlich durch dessen Spezifikation beschrieben.

2.1 Spezifikation

Die Spezifikation legt einen Großteil der Gestalt und der Teile fest, die in dem jeweiligen Auto verbaut sind bzw. werden. Insgesamt hat die Spezifikation mindestens eine Länge von neun Positionen, wobei jede Position der Kodierung aus drei Zeichen besteht, die sich aus Buchstaben und Zahlen zusammensetzen. Um ein valides Auto zu kodieren, müssen diese Position spezifiziert sein. Die ersten sieben Positionen gehören der sogenannten *Entity* an. Tabelle 2.1 gibt eine Übersicht und Beschreibung der einzelnen Positionen.

Die nächsten beiden Positionen (8 & 9) legen die Außenfarbe und die Polsterung des Autos fest. Auf die letzte Position folgend, werden falls vorhanden optionale *Features* kodiert. Optional deshalb, weil ein Auto produziert werden kann, das keine Features enthält. Als Beispiel für ein optionales Features seien ein Sonnendach oder ein vollwertiges Ersatzrad genannt.

2 Datengrundlage

Code	Position	Beschreibung	Beispiele
Modell	1	Klassifizierung von Autos der Marke Ford	Fiesta, Focus, Kuga, Mondeo, etc.
Jahr	2	Modelljahr	-
Karosserie	3	-	3-Türer, 5-Türer, etc.
Modellvariante	4	Untervariante für jedes Modell.	Ambiente, Trend, Titanium, etc.
SVP	5	engl. special value package	SYNC Edition, Sport Edition, etc.
Motor	6	Kodiert die Motorisierung und die Leistung	Benzin, Diesel, Gas
Getriebe	7	-	5-gang, 6-gang, Automatik

Tabelle 2.1: Kodierungen der Entity innerhalb des Spezifikationsvektors.

Zu jeder Position innerhalb der Spezifikation existiert in der Datenbank eine zugehörige Spalte, in der der jeweiligen Code genauer beschrieben wird. Auf diese Weise lässt sich eine Zuordnung von den 3-stelligen Codes innerhalb der Spezifikation zu einer detaillierteren Beschreibung bilden.

Problematisch wird das Bilden der Zuordnung bei der Wiederverwendung von bestimmten Codes. Dies geschieht, falls ein bestimmtes Teil durch ein neueres Teil ersetzt wird. Der Code wird erneut verwendet, während sich die Beschreibung ändert. Zur Lösung des Problems gilt es die Zeit in die Betrachtung miteinzubeziehen. Die Zuordnung ist für einen festgelegten Zeitpunkt eindeutig.

Beispiel 1. Die in diesem Beispiel aufgeführte Spezifikation besteht neben den neun notwendigen Codes aus weiteren fünf Codes, die optionale Features kodieren. In diesem fünf türigen, schwarzen Ford Fiesta ist unter anderem ein zusätzliches Sound-Paket verbaut.

Entity
optionale Features

00C YYV R85 8W5 8Y9 8ZB 9M5 9NA 9QF
9VB 9VG HOE HO2 KWD

2.2 Zeitpunkte

Auf dem Weg vom Einfügen der Bestellung eines Autos in die Datenbank bis zu dessen Auslieferung an den Kunden existieren mehrere nennenswerte Zeitpunkte, deren Zusammenhang in Abbildung 2.1 dargestellt ist. Eine Beschreibung der

einzelnen Zeitpunkte ist in Tabelle 2.2 zu finden.

Die Lage des *Contract dates* innerhalb der Zeit und im Speziellen in Relation zum aktuellen Betrachtungszeitraum ist von essentieller Bedeutung. Von besonderem Interesse sind die in Definition 1 eingeführten, als *BTS*-Autos (engl. build to stock) bezeichneten, Autos, da diese noch nicht an einen Kunden verkauft sind und somit Umplanungen ermöglichen.

Definition 1. Zu einem Betrachtungszeitpunkt t wird ein Auto als BTS-Auto bezeichnet, falls eine zugehörige Order mit Order date $\leq t$ in der Datenbank existiert und der Vertragsabschluss zu einem späteren Zeitpunkt (Contract date $> t$) oder noch nicht stattgefunden hat.

Für alle anderen geplanten Autos existieren bereits Vertragsabschlüsse. Diese Autos werden im Folgenden als *BTO*-Autos (engl: build to order) bezeichnet.

2.3 Zusätzliche Informationen

Unabhängig davon, ob es sich um ein BTS- oder BTO-Auto handelt, existiert in der Datenbank zu jeder Order eine Händlernummer. Diese macht kenntlich zu welchem Händler die jeweilige Order gehört. Im Verlauf der Arbeit werden alle BTS-Autos als Menge betrachtet, in der Umbauten stattfinden können. Dies ist eine unrealistische Annahme, da, nach Auskunft von Ford, Händler im Allgemeinen nicht bereit sind mit anderen Händlern Autos oder Autoteile zu tauschen.

Für die Datenanalysen in Kapitel 4 ist ein weiteres Datenfeld von Bedeutung. Die Spalte *PEST* (engl. pan european sales type) gibt an, an welche Zielgruppe das Auto verkauft worden ist. Dabei wird es in den meisten Fällen an Privatpersonen

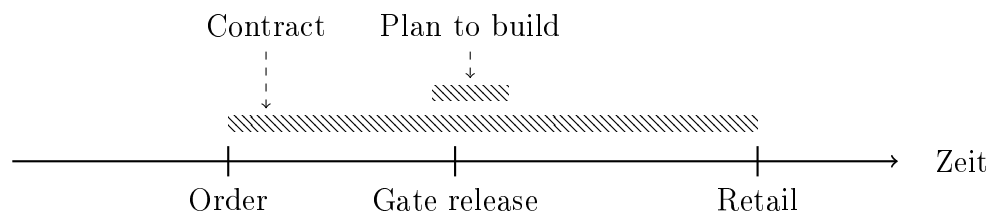


Abbildung 2.1: Zu sehen sind die für den Kontext fünf wichtigsten Zeitpunkte und ihre mögliche Lage innerhalb der Zeit. Die schraffierten Flächen geben den Zeitraum an, in welchem der jeweilige Zeitpunkt liegen kann.

2 Datengrundlage

Zeitpunkt	Beschreibung
Order date	Eingang der Bestellung in die Orderbank
Gate release date	Auto verlässt Werk
Retail date	Kundenübergabe
Contract date	Vertragsabschluss mit Kunden
Plan to build date	Voraussichtliches Produktionsdatum

Tabelle 2.2: Beschreibung der relevanten Zeitpunkte.

oder für den Hof des Händlers produziert. Ein beträchtlicher Teil wird zum Beispiel für Flottenkunden produziert. Diese Autos können sich stark in ihrer Spezifikation oder zumindestens ihrer Entity ähneln, was Analysen über die Verteilung von Spezifikationen stark beeinträchtigen kann.

Der monetäre Wert eines Autos ist bei der Interpretation späterer Ergebnisse in Abschnitt 5.6 von Bedeutung. Dieser ist in der Datenbank in der Spalte *inventory value* zu finden.

3 Data Mining Methoden

In diesem Kapitel wird ein Einblick in das weitreichende Thema Data Mining gegeben. Die unüberschaubare Menge an Daten macht eine Anwendung von Methoden aus dem Bereich Data Mining unabdingbar. Die Schwierigkeit dabei liegt jedoch in der Auswahl der richtigen Methoden, um Antworten auf spezifische Fragestellungen zu erhalten. Im Anschluss werden deshalb mehrere Algorithmen aus verschiedenen Bereichen vorgestellt, auf die während der Arbeit und im Kontext der genauen Problemstellung verwiesen wird.

3.1 Clustering

Unter Clustering versteht man das Aufspüren von Ähnlichkeitsstrukturen innerhalb unbekannter Daten. Datenobjekte, die nach einem bestimmten Maß ähnlich zueinander sind, sollen in Gruppen, sogenannten Clustern, zusammengefasst werden. Gleichzeitig sollen Datenobjekte, die unähnlich sind, in verschiedenen Clustern liegen.

Clusteringalgorithmen benutzen keinerlei Vorabinformationen über Cluster, in welche die Daten eingeteilt werden sollen. Daher eignen sich diese Methoden besonders als Vorbereitung für weitere Data Mining Methoden und zur ersten Übersichtsgewinnung über unbekannte Daten.

3.1.1 Partitionierende Clusterverfahren

Wie dem Namen bereits abzuleiten ist basiert diese Art des Clusterings auf dem Partitionieren von Datenobjekten in eine vorgegebene Anzahl an Cluster. Eine gegebene Partition wird in einem Clusteringschritt durch Austauschen von Elementen zwischen den Clustern verändert/verbessert.

k-Means-Algorithmus [M67]

Als Eingabeparameter erhält dieser Algorithmus lediglich die Anzahl der Cluster $k \in \mathbb{N}$ und versucht die Distanz zwischen den Datenpunkten $x \in X$ und dem nächstgelegenen Mittelpunkt m_i mit $i \in \{1, \dots, k\}$ zu minimieren. Mathematisch

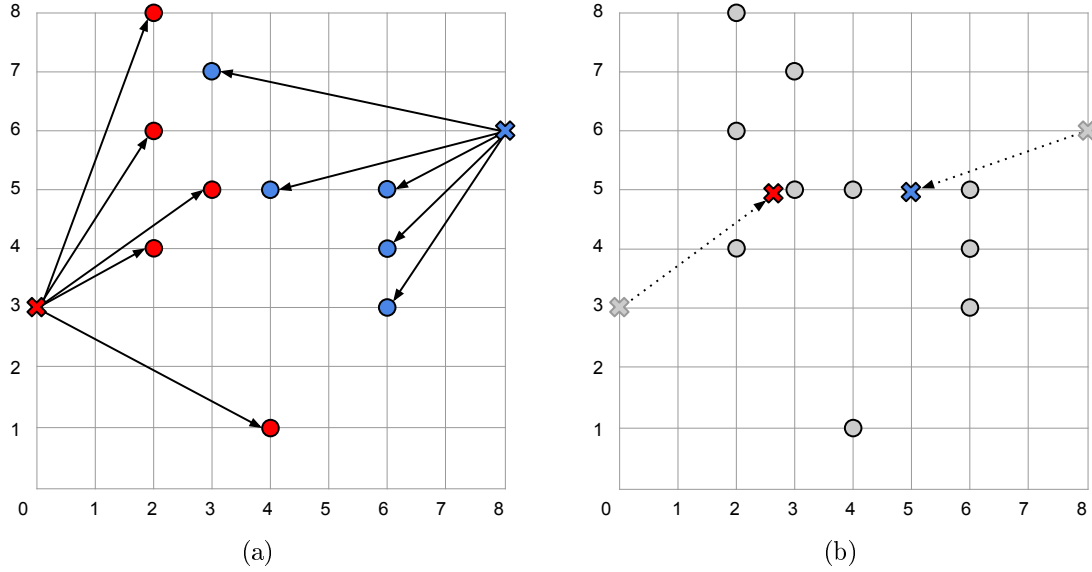


Abbildung 3.1: Sich wiederholende Distanzberechnung, Zuordnung und Aktualisierung.

gilt es demnach folgenden Ausdruck zu minimieren:

$$\sum_{i=1}^k \sum_{x_j \in S_i} \|x_j - m_i\|^2,$$

wobei S_i den zu m_i gehörigen Cluster beschreibt. Da die Suche nach einer optimalen Lösung NP-schwer ist, wird die Heuristik von Lloyd [L82] vorgestellt, welche ein lokales Minimum findet.

1. **Initialisierung:** Wähle k zufällige Clustermittelpunkte $m_1^0, \dots, m_k^0$.
2. **Distanzberechnung:** Berechne die Distanz von jedem Clustermittelpunkt m_i^t zu jedem Datenobjekt x_j .
3. **Zuordnung:** Ordne jeden Datenpunkt zu dem Cluster S_i des nächsten Clustermittelpunktes m_i hinzu.

$$S_i^t = \{x_j : \|x_j - m_i^t\|^2 \leq \|x_j - m_{i^*}^t\|^2 \ \forall i^* \in \{1, \dots, k\}\}$$
4. **Aktualisierung:** Verschiebe den neuen Mittelpunkt m_i^t auf den Mittelpunkt des Clusters S_i^t :

$$m_i^{t+1} = \frac{1}{|S_i^t|} \sum_{x_j \in S_i^t} x_j$$

Die Schritte zwei bis vier werden solange wiederholt, bis sich keine Zuordnung eines Datenobjektes zu seinem Cluster mehr ändert. In Abbildung 3.1 ist eine solche Wiederholung dargestellt. Die Distanzberechnung und Zuordnung zum nächsten Clustermittelpunkt sind Abbildung 3.1 (a) zu sehen, die darauffolgende Neuberechnung der Position der Clustermittelpunkte in Abbildung 3.1 (b).

Der Algorithmus terminiert sehr schnell und findet ein lokales Optimum, wobei dieses maßgeblich von der Initialisierung der Clustermittelpunkte abhängt. Diese Schwäche versucht eine der vielen Varianten, der k-Means++ Algorithmus [A07], durch die nicht zufällige Wahl der Clustermittelpunkte zu beheben. Desweiteren reagiert der Algorithmus sensitiv auf Ausreißer. Eine weitere Einschränkung ist dadurch gegeben, dass sich die Mittelpunkte lediglich für Daten berechnen lassen, für die ein Mittelwert sinnvoll definiert ist. Die Daten, beschrieben in Kapitel 2, gehören nicht dazu, was die Notwendigkeit des folgenden Algorithmus verdeutlicht.

k-Mode-Algorithmus [H97]

Der k-Mode-Algorithmus funktioniert nach der gleichen Struktur wie der k-Means-Algorithmus, jedoch wird die Berechnung des Mittelwertes durch den Modus und die euklidische Distanz durch eine Distanz auf kategorialen Variablen, beispielsweise die Hammingdistanz, ersetzt.

Der Modus einer diskreten Zufallsvariable beschreibt die Ausprägung mit der größten Wahrscheinlichkeit. Wie in Abbildung 3.2 zu sehen, ist es nicht mehr nötig, dass ein Mittelwert auf den Daten definiert ist. Zu bemerken ist, dass der Modus im Allgemeinen keinem der Datenobjekte entspricht. Im Kontext der Daten aus Kapitel 2 beschreibt der Modus demnach möglicherweise ein Auto, welches nicht baubar ist. Dies stellt keinen Konflikt bezüglich der Anforderungen an den Algorithmus dar.

Viel weiter von der Realität entfernt ist die Annahme, dass sich die Ähnlichkeit durch eine Distanzfunktion, wie der Hammingdistanz, beschreiben lässt. Diese impliziert, dass die Ähnlichkeit zwischen den Attributen gleichmäßig verteilt ist, was nicht der Realität entspricht.

3.1.2 Dichtebasierte Clusterverfahren

Eine der größten Einschränkungen partitionierender Clusteralgorithmen ist, dass diese nur konvexe Cluster finden können. Dichtebasierte Verfahren versuchen Cluster zu formen, in denen eine hohe Dichte an Datenobjekten herrscht. Gleichzeitig werden Cluster durch Gebiete mit niedriger Dichte getrennt.

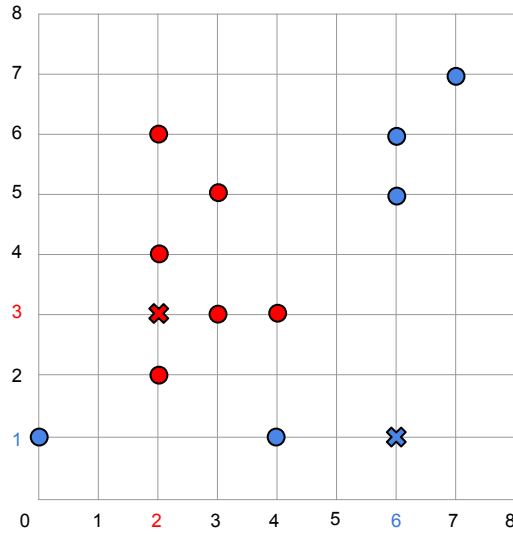


Abbildung 3.2: Aktualisierung der Clustermittelpunkte innerhalb der zugehörigen Cluster auf den Modus.

DBSCAN Algorithmus [E96]

Der DBSCAN (engl. *Density Based Spatial Clustering of Applications with Noise*) Algorithmus erhält als Eingabeparameter eine Distanz ϵ , welche mit Hilfe einer Distanzfunktion dist die Nachbarschaft

$$N_\epsilon(x) := \{x' \in X \mid \text{dist}(x, x') \leq \epsilon\}$$

eines Datenobjektes $x \in X$ beschreibt. Durch den zweiten Eingabeparameter wird eine Schranke für die minimale Anzahl an Nachbarn minPkt festgelegt, um innerhalb des Algorithmus von einer hohen Dichte sprechen zu können.

Es gibt drei verschiedene Klassen, in die Datenobjekte x von dem Algorithmus eingeteilt werden können:

1. **Kernpunkte:** $|N_\epsilon(x)| \geq \text{minPkt}$
2. **Randpunkte:** $|N_\epsilon(x)| < \text{minPkt} \wedge \exists x' \in X : x' \text{ ist Kernpunkt} \wedge x \in N_\epsilon(x')$
3. **Rauschpunkte:** $|N_\epsilon(x)| = 1$

Falls Datenobjekte in die erste Klasse eingeteilt werden, wird ein neuer Cluster erstellt und alle Punkte ihrer Nachbarschaft werden demselben Cluster zugeordnet. Diese Datenobjekte werden wiederum der oben beschriebenen Fallunterscheidung unterzogen und gegebenenfalls auch ihre, bisher unbetrachteten, Nachbarn zu dem Cluster hinzugefügt. Besitzt ein Datenobjekt nicht genügend Nachbarn, ist es entweder ein Randpunkt, falls es von einem Kernpunkt aus erreicht worden ist, oder

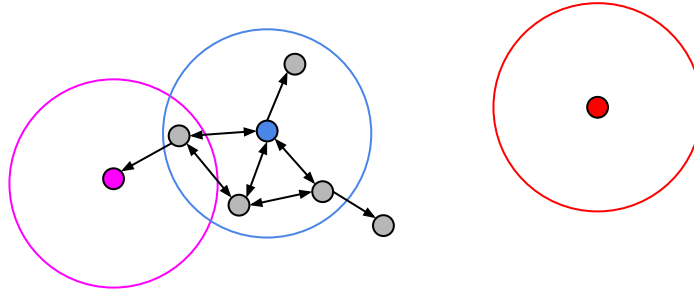


Abbildung 3.3: In blau ist ein Kernpunkt, in pink ein Randpunkt und in rot ein Rauschpunkt für $\text{minPkt} = 3$ zu erkennen.

andernfalls ein Rauschpunkt, falls es nicht in der Reichweite eines Kernpunktes liegt.

Abbildung 3.3 zeigt die beschriebene Vorgehensweise in einem Beispiel, in welchem der blaue Punkt als Kernpunkt klassifiziert wurde. Durch die schrittweise Erweiterung des Clusters wird unter anderem der pinke Punkt erreicht und als Randpunkt klassifiziert. Der rote Punkt besitzt nicht genügend bzw. gar keine Nachbarn und wird somit als Rauschpunkt erkannt. Als Ergebnis werden alle Punkte, mit Ausnahme des roten Punktes, der als Rauschpunkt klassifiziert wurde, in den gleichen Cluster eingeteilt.

Neben dem Vorteil beliebige Formen an Clustern erkennen zu können, ist vor allem die flexible Anzahl an Clustern für unbekannte Daten von Vorteil. Während bei den partitionierenden Clusteralgorithmen dem Anwender die Bestimmung des Eingabeparameters k überlassen wird, sind es beim DBSCAN Algorithmus ϵ und minPkt die manuell bestimmt werden müssen und somit Expertise verlangen. Aufgrund der Distanzberechnung zwischen allen Datenobjekten beträgt die Laufzeit $\mathcal{O}(|X|^2)$.

3.2 Assoziationsanalyse

Die Assoziationsanalyse, eingeführt in [A93], besitzt die populärste Anwendung im Gebiet der Warenkorbanalyse. In einem Supermarkt sollen beispielsweise Waren ermittelt werden, die häufig zusammen gekauft werden. Besonders von Interesse sind dabei überraschende oder unerwartete Regeln. Ein Beispiel bietet eine Assoziationsanalyse des Kaufhauses Walmart, in der herausgefunden wurde, dass Männer zu einer bestimmten Uhrzeit häufiger Bier und Windeln in Kombination kaufen. Diese Information kann zur Umsatzsteigerung, unter anderem durch gezielte Wer-

bung, genutzt werden.

Übertragen auf den vorhandenen Kontext sind es Autoteile oder Ausstattungsmerkmale, deren Korrelationen untereinander relevant sind. Impliziert das Vorhandensein eines Autoteils zu einer hohen Wahrscheinlichkeit das Vorhandensein eines anderen Autoteils, sollte diese starke Korrelation nicht durch eine Umplanung zerstört werden.

3.2.1 Häufige Attribute

Sei mit D die Datenbank bezeichnet. Jede Zeile von D wird als Transaktion T angesehen und beschreibt in unserer Anwendung ein Auto, welches durch die verwendeten Attribute $a \in \mathcal{A}$ (als Spalten anzusehen) eindeutig beschrieben wird. Von Interesse sind Teilmengen von Attributen $A \subseteq \mathcal{A}$ die häufig in der Datenbank vorkommen und somit eine Schlussfolgerung auf Muster oder Trends zulassen. Diese Eigenschaft wird in Definition 2 eingeführt.

Definition 2. Der Support [A93] beschreibt für eine gegebene Teilmenge $A \subseteq \mathcal{A}$ die relative Häufigkeit innerhalb von D und ist definiert als:

$$\text{supp}(A) = \frac{|\{T \in D \mid A \subseteq T\}|}{|D|}$$

Das minimale Vorkommen und somit die minimale Relevanz wird vom Anwender durch die Schranke minSupp festgelegt. Als häufige Attributmengen A werden die bezeichnet, für die gilt:

$$\text{supp}(A) \geq \text{minSupp}$$

Zur Bestimmung aller häufigen Attributmengen einer Datenbank könnte ein naiver Ansatz verfolgt und alle möglichen Kombinationen von Attributen berechnet und gezählt werden. Da dies der Potenzmenge über $\mathcal{A}$ entspricht, ist eine exponentielle Laufzeit zu erwarten.

Folgendes Lemma beschreibt Eigenschaften von häufigen Attributmengen, die das Erkennen dieser deutlich erleichtern.

Lemma 1. Das Apriori Prinzip [A93]:

- Jede nicht leere Teilmenge einer häufigen Attributmengen kommt ebenfalls häufig vor.
- Jede Obermenge einer nicht häufigen Attributmengen kommt ebenfalls nicht häufig vor.

3 Data Mining Methoden

Att ₁	Att ₂	Att ₃	Att ₄	Att ₅
1	1	0	1	0
1	1	0	0	1
0	1	0	1	1
1	0	1	0	0
0	1	0	1	1

Tabelle 3.1: Darstellung der Datenbank.

Attributmenge	Support
{1}	0.6
{2}	0.8
{3}	0.2
{4}	0.6
{5}	0.6

Tabelle 3.2: Support der Itemsets.

Der Apriori-Algorithmus [A93] macht sich diese Eigenschaften zunutze. Als erstes werden alle einelementigen Attributmengen, also alle Attribute, betrachtet und ihr Support berechnet. Aus den resultierenden häufigen Attributen werden, begründet durch Lemma 1, alle möglichen Attributmengen der Kardinalität zwei berechnet. Dieses Verfahren wird fortgeführt, bis keine häufigen Attributmengen mehr kombiniert werden können.

Die Frage, ob eine generierte Menge A häufig vorkommt, kann in zweiter Instanz durch die Berechnung ihres Supports bestimmt werden. Das Ausschließen kann bereits in einem ersten Schritt durch die Anwendung des Apriori-Prinzips geschehen. Sind nicht alle $(|A| - 1)$ -elementigen Teilmengen von A als häufig erkannt, kommt A in jedem Fall nicht häufig vor.

Beispiel 2. Sei $\text{minSupp} = 0.5$ und die Datenbank wie in Tabelle 3.1 gegeben. Bei der Berechnung des Supports (Tabelle 3.2) fällt auf, dass Att_3 nicht über die nötige Relevanz verfügt und somit nicht in die Generierung mehrelementiger Attributmengen mit einbezogen wird.

Innerhalb der zweielementigen Attributmengen kommen $\{2, 4\}$ und $\{2, 5\}$, mit einem Support von 0.6, häufig vor. Die Mengen unterscheiden sich lediglich in einem Attribut. Deshalb wird die dreielementige Attributmenge $\{2, 4, 5\}$ zwar erzeugt, jedoch bereits durch den ersten Test ($\text{supp}(\{4, 5\}) = 0.4$) als nicht häufig vorkommend erkannt.

Mit FP-Groth [H00] sei auf einen Algorithmus verwiesen, der auf die Generierung der Attributmengen vollständig verzichtet und besonders auf sehr großen Datenmengen höhere Performance erreicht. Aufgrund der Größenordnung der vorhandenen Daten und der Tatsache geschuldet, dass das Erkennen der häufigen Attributmengen später als Preprocess ausgeführt wird, ist der Apriori-Algorithmus vollkommen ausreichend. Durch die Vielzahl an Restriktionen bei der Kombination verschiedener Attribute, beschrieben in Abschnitt 4.1, terminiert dieser Algorithmus sehr schnell.

3.2.2 Assoziationsregeln [A93]

Im zweiten Schritt der Assoziationsanalyse werden Regeln der Form $A \Rightarrow B$ extrahiert, wobei $A, B \subseteq \mathcal{A}$ gilt. Wie bereits für die Attributmengen beschrieben, sind Assoziationsregeln lediglich ab einer bestimmten relativen Häufigkeit relevant, weshalb der Support ebenfalls für Assoziationsregeln definiert ist.

Definition 3. Der Support einer Assoziationsregel $A \Rightarrow B$ mit $A, B \subseteq \mathcal{A}$ und $A \cap B = \emptyset$ ist definiert als:

$$\text{supp}(A \Rightarrow B) = \frac{|\{T \in D \mid A \cup B \in T\}|}{|D|}$$

Es sind besonders die Regeln interessant, die eine hohe Vertraulichkeit besitzen, d.h. falls $A \in T$ ist, zu einer hohen Wahrscheinlichkeit ebenfalls $B \in T$ folgt. Diese Beziehung zwischen Attributmengen beschreibt das Maß Confidence, eingeführt in Definition 4. Die geforderte untere Schranke an Vertraulichkeit wird mit minConf bezeichnet.

Definition 4. Das Maß Confidence [A93] für zwei Itemsets $A, B \subseteq \mathcal{A}$ und $A \cap B = \emptyset$ ist definiert als:

$$\text{conf}(A \Rightarrow B) = \frac{\text{supp}(A \cup B)}{\text{supp}(A)}$$

Für die Extraktion der relevanten Assoziationsregeln wird sich das Wissen über häufig vorkommenden Attributmengen zu nutze gemacht. Damit eine Regel $A \Rightarrow B$ einem minSupp genügt, muss nach Definition 2 die Attributmenge $A \cup B$ ebenfalls häufig vorkommen. Aus Abschnitt 3.2.1 sind all diese Attributmengen bereits bekannt. Noch zu ermitteln ist die Aufteilung, der in einer häufig vorkommenden Attributmenge enthaltenden Attribute, auf die Mengen A und B . Es folgt eine Skizze des Algorithmus.

Algorithm 1 Extraktion von Assoziationsregeln aus häufigen Attributmengen.

Input: Menge der häufigen Itemsets $\mathcal{A}^*$

```

for all  $A \in \mathcal{A}^*$  do
  for all  $B \subset A \wedge B \neq \emptyset$  do
    if  $\text{conf}(B \Rightarrow (A - B)) \geq \text{minConf}$  then
      füge Regel hinzu
  
```

Output: Alle Assoziationsregeln $X \Rightarrow Y$ mit $\text{supp}(X \Rightarrow Y) \geq \text{minSupp}$ und $\text{conf}(X \Rightarrow Y) \geq \text{minConf}$.

Für eine häufig vorkommende Attributmenge A mit $|A| = m$ betrachtet der Algorithmus alle $2^m - 2$ und damit exponentiell viele Möglichkeiten der Aufteilung.

3 Data Mining Methoden

Kompensiert wird diese Laufzeit durch das seltene Auftreten von häufigen Attributmengen mit hoher Kardinalität. Die Berechnung der Vertraulichkeit im inneren der Schleifen kann durch das Ablesen des Supports geschehen, da gilt:

$$\text{conf}(B \Rightarrow (A - B)) = \frac{\text{supp}(B \cup (A - B))}{\text{supp}(B)} = \frac{\text{supp}(A \cup B)}{\text{supp}(B)}$$

Beispiel 3. Betrachtet man die Datenbank aus Beispiel 2, lässt sich unter anderem die folgende Regel ableiten:

$$\text{Att}_2 \Rightarrow \text{Att}_5 [0.6, 0.75]$$

Mit 60% liegt diese über dem geforderten minSupp und besitzt mit 75% eine hohe Vertraulichkeit. Eine weitere Regel soll die Schwäche der bisherigen Maße aufzeigen.

$$\text{Att}_1 \Rightarrow \text{Att}_2 [0.4, 0.667]$$

Mit über 66% besitzt die Regel eine relativ hohe Vertraulichkeit. Allerdings führt sie zu einer Missinterpretation, da der Support für Att_2 bereits 80% beträgt. Die gesuchte Regel wäre $\text{Att}_1 \Rightarrow \neg \text{Att}_2$. Die beiden Attribute sind negativ korreliert.

Beispiel 3 macht auf das Problem beim Extrahieren von Assoziationsregeln auf negativ korrelierten Attributen aufmerksam. Zur Lösung dieses Problems wird ein drittes Maß, genannt *Lift* (Definition 5), eingeführt und verwendet. Das Maß gibt an, wie viel häufiger bzw. seltener Attribute gleichzeitig vorkommen, als dass sie statistisch unabhängig voneinander wären.

Definition 5. Der Lift [B97] einer Assoziationsregel $A \Rightarrow B$ ist definiert als:

$$\text{lift}(A \Rightarrow B) = \text{lift}(B \Rightarrow A) = \frac{\text{supp}(A \cup B)}{\text{supp}(A) \cdot \text{supp}(B)}$$

Die in Beispiel 3 genannte Regel $\text{Att}_1 \Rightarrow \text{Att}_2 [0.4, 0.667]$ besitzt einen lift von ca. 0.83. Die beiden Attribute kommen demnach statistisch seltener zusammen vor, als wenn sie unabhängig voneinander wären. Die korrekte Schlussfolgerung wäre die Regel $\text{Att}_1 \Rightarrow \neg \text{Att}_2 [0.2, 0.33]$. Diese besitzt jedoch weder das geforderte relative Vorkommen, noch eine nennenswerte Vertraulichkeit.

3.3 Klassifikation

In diesem Kapitel werden Entscheidungsbäume als Vertreter der Klassifikation eingeführt. Ziel der Klassifikation ist es unbekannte Datenobjekte anhand ihrer Attribute in vorher festgelegte Klassen einzuteilen. Die Regeln, die für eine solche Klassifikation nötig sind, werden in einem ersten Schritt aus Trainingsdaten, deren Klassenzugehörigkeit bekannt ist, abgeleitet. Das Vorhandensein und der Nutzen dieser Trainingsdaten ist der Hauptunterschied zu den in Abschnitt 3.1 vorgestellten Methoden des Clusterings.

3.3.1 Entscheidungsbäume [Q86]

Die Grundlage für die Konstruktion von Entscheidungsbäumen bilden die Trainingsdaten T . Jedes Trainingsobjekt $t \in T$ wird durch mehrere Attribute $A = \{A_1, \dots, A_k\}$ beschrieben und ist jeweils genau einer Klasse $c \in C$ zugeordnet. Die inneren Knoten eines Entscheidungsbaumes entsprechen Attributen, anhand derer die Trainingsdaten partitioniert werden. Dazu wird ein Attribut $A^* \in A$ ausgewählt, welches auf dem Pfad zur Wurzel bisher nicht ausgewählt wurde. Für kategoriale Attribute wird ein Zweig für jeden möglichen Wert $v \in \text{dom}_{A^*}$ gebildet, welcher alle Trainingsdaten beschreibt, die in Attribut A^* den Wert v besitzen. An dieser Stelle folgt entweder ein rekursiver Aufruf zur Konstruktion eines weiteren Entscheidungsbaumes auf den Trainingsdaten $T_{|A^*=v}$ oder die Erzeugung eines

Algorithm 2 Konstruktion eines Entscheidungsbaumes

Input: ID3(Trainingsdaten T , Klassen C , Attribute A)

```

Generiere Wurzelknoten
for all  $c \in C$  do
    if  $\forall t \in T : \text{class}(t) = c$  then
        return Wurzelknoten mit Beschriftung  $c$ 
if  $A = \emptyset$  then
    return häufigste Klasse in  $T$ 
else
    Wähle "bestes" Attribut  $A^* \in A$ 
    Beschrifte Wurzelknoten mit  $A^*$ 
    for all  $v \in \text{dom}_{A^*}$  do
        Erzeuge Zweig mit  $A^* = v$ 
         $T_{|A^*=v}$  Trainingsdaten unter der Bedingung  $A^* = v$ 
        if  $T_{|A^*=v} = \emptyset$  then
            return Blatt mit Beschriftung der häufigsten Klasse in  $T$ 
        else
            erzeuge Entscheidungsbaum ID3( $T_{|A^*=v}$ ,  $C$ ,  $A \setminus A^*$ )

```

3 Data Mining Methoden

mit $c \in C$ beschrifteten Blattknotens, abhängig von der Klassenzugehörigkeit von $T|_{A^*=v}$. Blattknoten stellen nach der mehrstufigen Anwendung von Regeln auf die Trainingsdaten die Entscheidung dar, in welche Klasse die Partition eingeteilt wird. Ziel der Konstruktion von Entscheidungsbäumen ist es, anhand der Attribute die Trainingsdaten so zu partitionieren $T = T_1 \dot{\cup} \dots \dot{\cup} T_m$, dass innerhalb einer Partionen T_i mit $i \in \{1, \dots, m\}$ eine möglichst starke Homogenität bezüglich der Klassenzugehörigkeit herrscht.

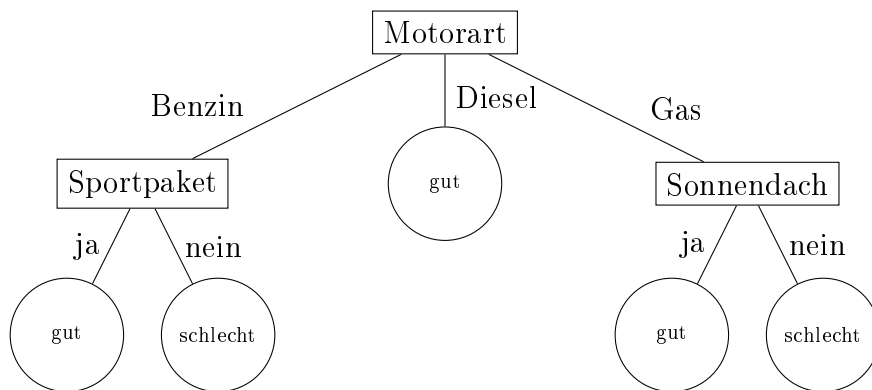
Algorithmus 2, eingeführt in [Q86], ist eine mögliche Umsetzung der beschriebenen Vorgehensweise. ID3 entscheidet an einem Knoten nach dem Prinzip eines Greedy-Algorithmus welches Attribut sich am besten zur Trennung der Daten eignet.

Beispiel 4. Die in der folgenden Tabelle aufgeführten Trainingsdaten beschreiben Automodelle, welche sich durch den Motor, ein ggf. verbautes Sportpaket und ein als Extra eingebautes Sonnendach unterscheiden lassen. Wie bereits beschrieben ist für Trainingsdaten die Klassenzugehörigkeit, in diesem Falle die Verkaufbarkeit, mit möglichen Werten *gut* und *schlecht*, bekannt.

ID	Motorart	Sportpaket	Sonnendach	Verkaufbarkeit
1	Benzin	nein	ja	schlecht
2	Benzin	nein	nein	schlecht
3	Diesel	nein	ja	gut
4	Gas	nein	ja	gut
5	Gas	ja	ja	gut
6	Gas	ja	nein	schlecht
7	Diesel	ja	nein	gut
8	Benzin	nein	ja	schlecht
9	Benzin	ja	ja	gut
10	Gas	ja	ja	gut
11	Benzin	ja	nein	gut
12	Diesel	nein	nein	gut
13	Diesel	ja	ja	gut
14	Gas	nein	nein	schlecht

Ein aus den oben aufgeführten Trainingsdaten konstruierter Entscheidungsbaum könnte wie folgt aussehen:

Bisher ungeklärt sind die Fragen, welches Attribut die Trainingsdaten am besten hinsichtlich der Homogenität partitioniert und wie nicht kategoriale Attribute verzweigt werden können. Die nächsten beiden Abschnitte liefern Antworten auf diese Fragen.



3.3.2 Entropy [E48]

Algorithmus ID3 wählt an den inneren Knoten ein Attribut aus, um die Trainingsdaten zu partitionieren. Ziel ist es, ein Attribut zu wählen, welches durch seine Charakteristik möglichst homogene Gruppen erschafft. Die *Entropy* ist ein Maß, welches diese Eigenschaft beschreibt.

Definition 6. Die Entropy H für eine Trainingsdatenmenge T , mit relativer Häufigkeit p_c für das Vorkommen einer Klasse $c \in C$, ist definiert als:

$$H(T) = - \sum_{c \in C} p_c \cdot \log_2(p_c)$$

Abbildung 3.4 verdeutlicht das Verhalten der Funktion für zwei Klassen. Je weniger die Trainingsobjekte gleichverteilt in den Klassen liegen, desto kleiner wird die Entropy. Im ungünstigsten Fall sind die Trainingsdaten zu jeweils 50% auf die

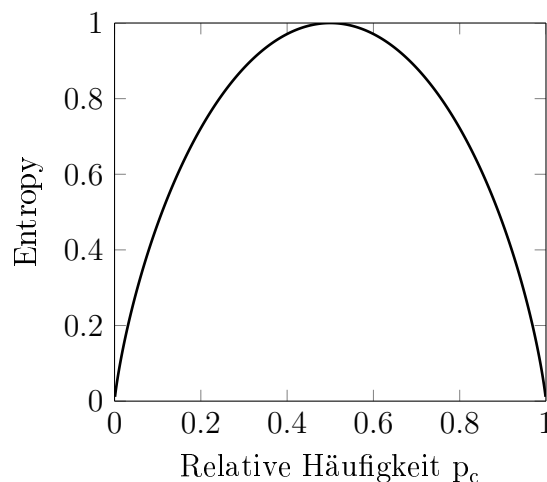


Abbildung 3.4: Geplottete Entropy Funktion für zwei Klassen.

Klassen aufgeteilt. Es ist keine Trennschärfe vorhanden und wie gewünscht nimmt die Entropy ihren maximalen Wert von eins an. Die Entropy verhält sich proportional zur Homogenität und somit wird versucht die Entropy zu minimieren.

An einem inneren Knoten des Entscheidungsbaumes ist die Gewinnung an Trennschärfe innerhalb der Daten von Interesse. Deshalb wird von der Entropy der ausgehenden Trainingsdaten $H(T)$ die gewichtete Entropy der Partitionen abgezogen. Als Resultat bleibt der sogenannte Informationsgewinn übrig.

Definition 7. Der Informationsgewinn für eine Trainingsmenge T und ein gewähltes Attribut A^* ist definiert als:

$$IG(T, A^*) = H(T) - \sum_{v \in \text{dom}_{A^*}} \frac{|T_{|A^*=v}|}{|T|} H(T_{|A^*=v})$$

Für die Beschriftung der inneren Knoten wird der Informationsgewinn aller Attribute, die auf dem Pfad zur Wurzel noch nicht zur Partition benutzt wurden, berechnet und das Attribut mit dem maximalen Informationsgewinn ausgewählt.

Beispiel 5. Als Fortsetzung von Beispiel 4 lässt sich der Informationsgewinn an der Wurzel für das Attribut Sportpaket wie folgt berechnen:

$$IG(T, \text{Sportpaket}) = 0.9403 - \left(\frac{7}{14} \cdot 0.9852 + \frac{7}{14} \cdot 0.5917 \right) = 0.1519$$

Der Informationsgewinn für alle Attribute wurde in Abbildung 3.5 errechnet. Ausgehend von der lokalen Sicht der Wurzel bringt die Unterscheidung nach der Motorart den größten Informationsgewinn und dient daher als erstes Attribut für die Partitionierung.

3.3.3 Verzweigung

Bisher wurde die Erzeugung von Zweigen lediglich für kategoriale Attribute betrachtet. Anstatt für jeden Wert aus der Domäne eines Attributes einen Zweig zu erzeugen, ist es möglich die Domäne zu partitionieren und die Partitionen als Kriterium an den Zweigen zu verwenden (siehe Abbildung 3.6). Dieses Vorgehen führt nicht zu einer Verbesserung der Trennschärfe, jedoch verhindert die Trennung von Trainingsdaten, die sich zwar in dem spezifischen Attribut unterscheiden, aber bezüglich ihrer Klassenzugehörigkeit ähnlich zueinander sind. Eine Aufteilung nach allen Werten der Domäne und eine spätere Gruppierung von homogenen Gruppen ist ein praktikabler Ansatz.

Deutlich mehr Entscheidungsgewicht liegt bei der Behandlung von kontinuierlichen Attributen vor. Diese müssen diskretisiert werden, entweder durch die Ermittlung

3 Data Mining Methoden

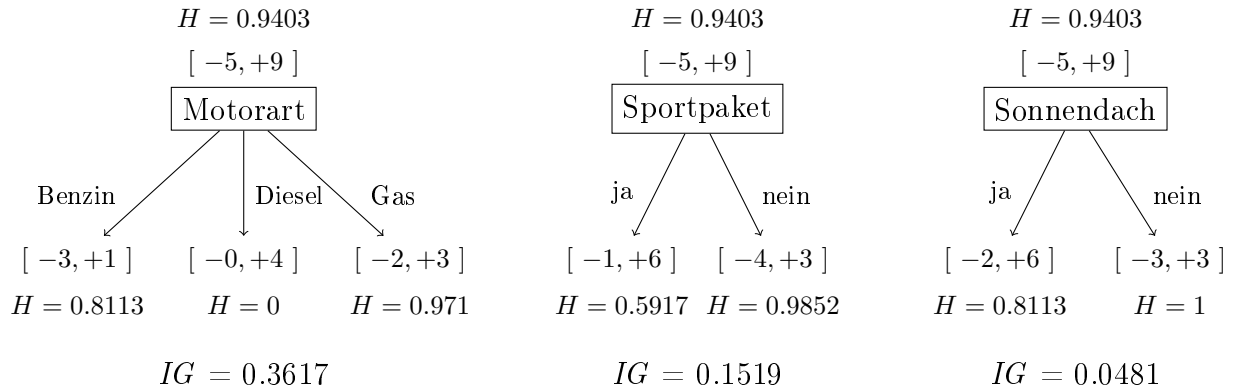


Abbildung 3.5: Die Bezeichnung $[-n, +m]$ gibt an, dass n Trainingsdaten innerhalb der Klasse *schlecht* liegen und m innerhalb der Klasse *gut*. Für jedes Attribut wird die Entropy H vor und nach der Verzweigung durch das Attribut berechnet. Final geht die Entropy jedes Zweiges gewichtet in die Berechnung des Informationsgewinns ein.

durch einen Trennpunkt, der alle größeren Daten von den kleineren unterscheidet, oder durch eine Intervalleinteilung (siehe Abbildung 3.7).

Generell gibt es unendlich viele Möglichkeiten ein kontinuierliches Intervall auf diese Weise zu diskretisieren. Ein pragmatischer Ansatz liegt darin, die endlich vorkommenden Werte aus den Trainingsdaten als mögliche Trennwerte zu betrachten. Dies ist in der Weiterentwicklung von ID3, dem Algorithmus C4.5 [Q93], umgesetzt.

3.3.4 Diskussion

Durch ihre Darstellung als Baum gelten Entscheidungsbäume als intuitiv für den Betrachter. Dabei ist bei der Konstruktion darauf zu achten einen möglichst kompakten Baum zu erzeugen. Dies wird erreicht, indem Verzweigungen mit einem zu geringen Informationsgewinn nicht durchgeführt werden und der Grad der Verzweigungen, wie in Abschnitt 3.3.3 beschrieben, reduziert wird.

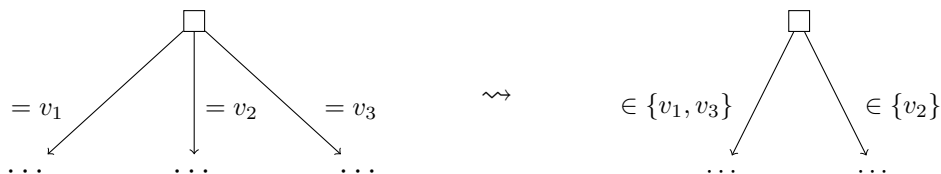


Abbildung 3.6: Gruppierung von kategorialen Daten.

3 Data Mining Methoden

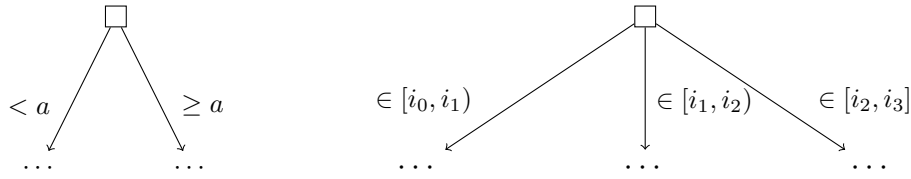


Abbildung 3.7: Einteilung von kontinuierlichen Daten.

Jeder Entscheidungsbaum lässt sich problemlos in disjunkter Normalform (DNF) ausdrücken. Auf einem Pfad von Wurzel zu Blattknoten werden alle Bedingungen, geformt durch die an den inneren Knoten gewählten Attribute, mit einer Konjugation versehen. Wird die Disjunktion über alle so entstandenen Klauseln gebildet, wird exakt der Entscheidungsbaum repräsentiert.

Beispiel 6. Der Entscheidungsbaum aus Beispiel 4 wird durch die folgende DNF beschrieben:

$$\begin{aligned}
 &(\text{Motorart} = \text{Benzin} \wedge \text{Sportpaket} = \text{ja}) && \vee \\
 &(\text{Motorart} = \text{Benzin} \wedge \text{Sportpaket} = \text{nein}) && \vee \\
 &(\text{Motorart} = \text{Diesel}) && \vee \\
 &(\text{Motorart} = \text{Gas} \wedge \text{Sonnendach} = \text{nein}) && \vee \\
 &(\text{Motorart} = \text{Gas} \wedge \text{Sonnendach} = \text{nein}) && \vee
 \end{aligned}$$

Die erwähnten Schwierigkeiten für kontinuierliche Attribute sind aufgrund der Datenbeschaffenheit, beschrieben in Abschnitt 2, zu vernachlässigen. Ein viel größeres Problem stellt die Klassenzugehörigkeit dar. Da Entscheidungsbäume nicht robust gegenüber Veränderungen, insbesondere bei der Klassenzuordnung der Trainingsdaten sind, nehmen kleine Änderungen in der Definition der Verkaufbarkeit großen Einfluss auf die Gestalt des Entscheidungsbaumes. Eine Vergleichbarkeit zweier Entscheidungsbäume ist unmöglich.

4 Datenanalyse

Dieses Kapitel dient durch die Anwendung von Data Mining Methoden auf die vorhandenen Daten als Zusammenführung von Kapitel 2 und 3. Ziel ist es eine Vorstellung davon zu vermitteln, welche Autos gebaut werden, wann ein Auto als verkaufbar gilt und welche Autoteile oft bzw. selten zusammen verbaut werden. Die Angaben beziehen sich auf Verkaufsdaten von Ford Fiestas, welche für den deutschen Markt von Juli 2013 bis Juni 2014 produziert worden sind.

4.1 Statistische Analyse

Insgesamt sind im betrachteten Jahr 46.090 Ford Fiestas produziert worden. Davon wurden ca. 57% auf Kundenwunsch gebaut und gehören somit der Kategorie der BTO-Autos an. Folglich bleiben 43% übrig, die für das Lager der Händler produziert wurden und somit nach den getroffenen Annahmen in Kapitel 1 während ihrer Planungsphase für einen Umbau zu Verfügung stehen. Durchschnittlich stehen einem Umplanungsansatz, der im folgenden Kapitel vorgestellt wird, pro Tag 54 geplante Autos zur Verfügung.

Innerhalb der Entity lassen sich die Modellvariante und SVP, wie in Abbildung 4.1 zu erkennen, substituieren. Dadurch stehen im Gesamten 14 verschiedene Untermodelle des Fiestas zur Auswahl. Bedingt durch die Auswahl von 14 Motoren, die sich in Benzin-, Diesel- und Gasmotoren aufteilen, 3 Getriebearten und 2 verschiedenen Karosserien, ist es theoretisch möglich 1176 verschiedene Entities zu konstruieren. Die tatsächliche Anzahl kann in dieser Arbeit aufgrund von fehlender Produktspezifikation nicht genannt werden. Jedoch wurden im betrachteten Zeitraum lediglich 166 verschiedene Entities gebaut. Dies entspricht 9.8% der möglichen Entities, was die Schlussfolgerung zulässt, dass viele Kombinationen von Autoteilen verboten sind.

Gestützt wird diese Theorie durch das in Abbildung 4.2 veranschaulichte relative Vorkommen von einem Motor in einer bestimmten Modellvariante. Lediglich 27.7% aller möglichen Kombinationen kommen vor. Wird zum Beispiel SERIES 30 genauer betrachtet, welche zu einer der günstigeren Modellvarianten des Fiestas gehört, fällt auf, dass deutlich mehr Motoren mit geringerer als mit hoher Leistung verbaut worden sind. Die Vermutung liegt nahe, dass Kunden, die sich für eine günstigere

4 Datenanalyse

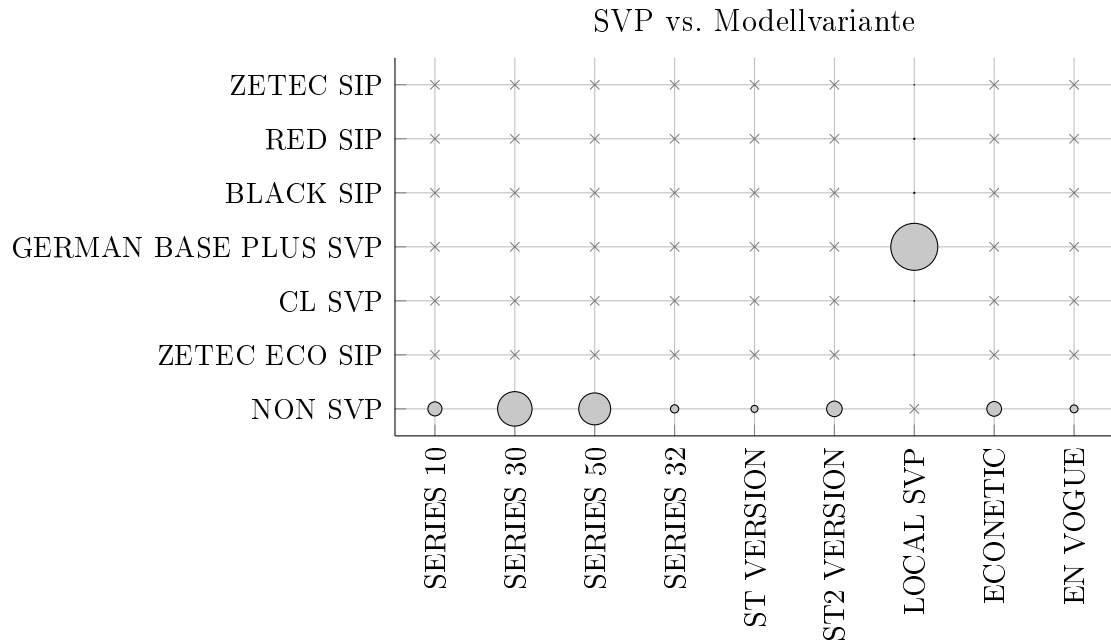


Abbildung 4.1: Die Kreise symbolisieren durch ihre Größe die relative Häufigkeit des gleichzeitigen Vorkommens einer SVP- und Modellvariante. Die Kreuze wurden gezeichnet, um wenig von gar keinem Vorkommen zu unterscheiden.

Modellvariante entscheiden, keinen hohen Wert auf die Motorisierung ihres Autos legen.

Werden in die theoretische Betrachtung die optionalen Features miteinbezogen, wächst die Anzahl der möglichen Kombinationen beträchtlich. Für 84 Features besteht die Wahl, ob diese eingebaut werden. Insgesamt lässt sich die Anzahl durch $1176 \cdot 2^{84}$ berechnen. Die resultierende 28-stellige Zahl steht zu den 6849 in der Realität gebauten unterschiedlichen Spezifikationen in keinerlei Verhältnis.

Wie zu erwarten ist, ist die Anzahl, der in einem BTO-Auto verbauten Features, im Durchschnitt 4.22 pro Auto, größer als die in einem BTS-Auto (3.37). Ein Auto, welches aus einem speziellen Kundenwunsch entstanden ist, besitzt im Durchschnitt mehr optionale Features als ein BTS-Auto. Bekannt ist in jedem Fall, dass sich Autos ohne jegliche optionalen Features, genannte tin-cans (engl. Blechdosen), im Allgemeinen schlecht verkaufen lassen. All diese Erkenntnisse, gewonnen aus der Datenanalyse oder aus der Expertise bekannt, sollten in den Prozess der Ermittlung einer Lösung miteinbezogen werden.

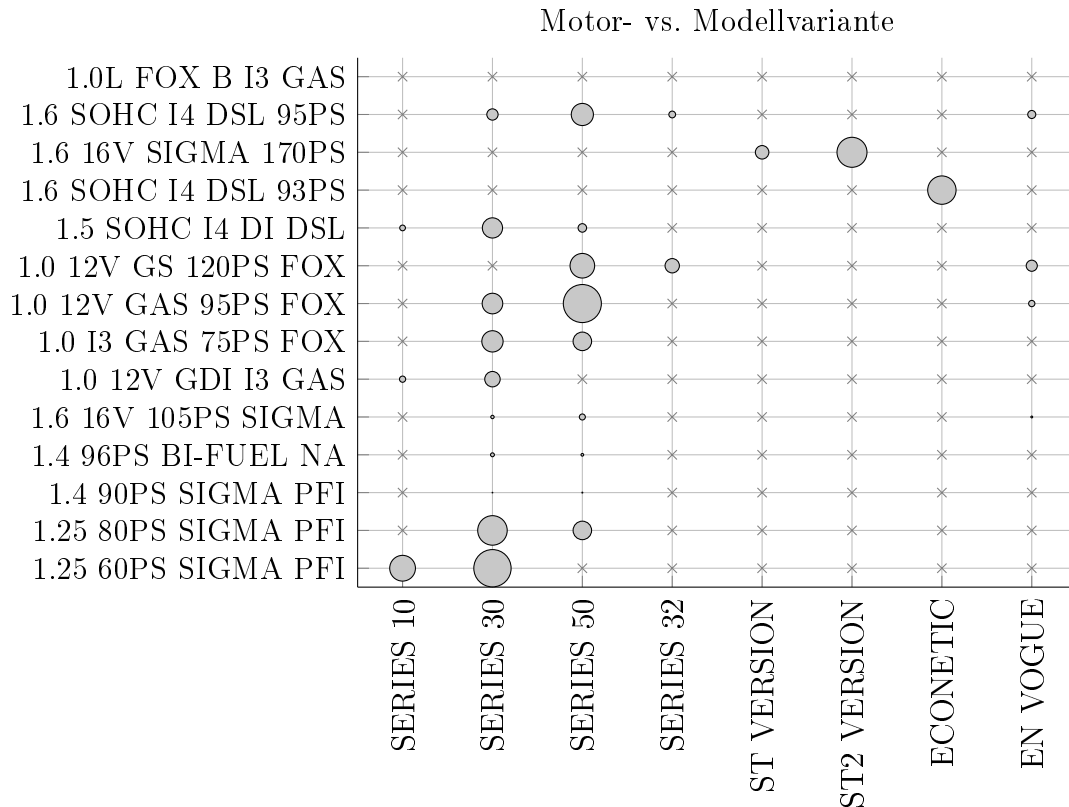


Abbildung 4.2: Relatives Vorkommen von Kombinationen aus Motor- und Modellvariante. Das Maximum wurde auf 1 normiert und ist durch den größten Kreis dargestellt.

4.2 Qualitätsmaß

Das Qualitätsmaß soll die Verkaufbarkeit eines Auto widerspiegeln. Besonders interessant ist das Qualitätsmaß für BTS-Autos, da BTO-Autos bereits an den Kunden verkauft sind und somit nicht vom Nachteil betroffen sein können, lange unverkauft zu bleiben. BTS-Autos sind Bestellungen, die von einem Händler in die Datenbank eingegeben worden sind und einer Spezifikation entsprechen, die nach Meinung des Händlers eine hohe Verkaufbarkeit besitzt. Werden diese Bestellungen durch einen Umbau verändert, sollten die entstehenden Autos ebenfalls ein hohes Qualitätsmaß besitzen. Offen bleibt die Frage, wie die Verkaufbarkeit einer Spezifikation gemessen werden kann.

Der wahrscheinlich aussagekräftigste Indikator für die gute Verkaufbarkeit eines Autos ist die Zeit, die das Auto unverkauft beim Händler steht. In Abbildung 4.3 ist die aggregierte Wartezeit, berechnet aus dem Zeitraum zwischen Produktionstag und dem Tag des Vertragsabschlusses, aller BTS-Autos in Wochen angegeben. Um

4 Datenanalyse

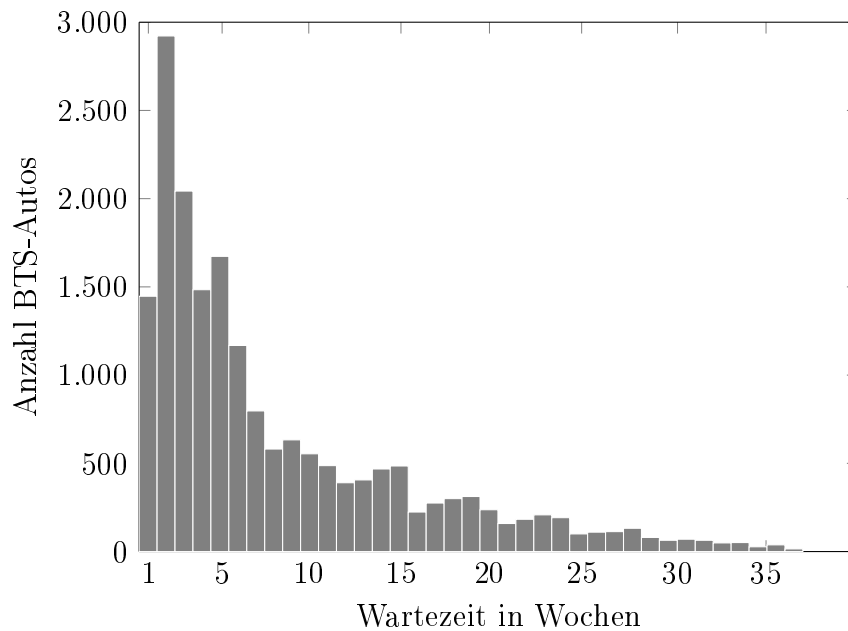


Abbildung 4.3: Die Verteilung der Wartezeit eines BTS-Autos.

vorhandene Ausreißer mit Wartezeiten von über 60 Wochen auszuschließen, wurde das minimale und maximale 1% der Messdaten ausgeschlossen.

Die Tatsache, dass in der zweiten Woche durchschnittlich die meisten Autos verkauft werden, hängt u.a. mit der Lieferung des Autos zum Händler zusammen. Um die Wartezeit zu berechnen, wird der Zeitraum zwischen Gate release date und Contract date betrachtet. In den meisten Fällen muss das Auto für einen Vertragsabschluss beim Händler stehen und somit muss eine gewisse Transportzeit eingerechnet werden.

Untersucht wird an dieser Stelle die Verkaufsgeschwindigkeit von Autos, die anhand ihrer Entity gruppiert sind. Auf Spezifikations-Level zu aggregieren ist wenig sinnvoll, da die Anzahl an gebauten Autos für eine Spezifikation oft zu gering ist, um eine fundierte Aussage treffen zu können.

Für eine gegebene Entity ist in Abbildung 4.4 genau eine Linie zu sehen. Dabei handelt es sich über die Zeit betrachtet um den kumulierten prozentualen Anteil an verkauften Autos, die der jeweiligen Entity entsprechen. Der durchschnittliche Verlauf der Kurven steht im Einklang mit der dargestellten Wartezeit in Abbildung 4.3. Zu Beginn besitzen die Kurven eine starke Steigung und implizieren somit den Verkauf der meisten Autos. Im weiteren Verlauf findet eine gewisse Sättigung statt. Während, wie in Abbildung 4.3 zu sehen, ein Großteil der Autos nach 15 Wochen verkauft wird, gibt es gewisse Entities, die nach diesem Zeitraum zu weniger als 50% verkauft worden sind. Diese Entities lassen sich im Durchschnitt schlechter verkaufen als andere.

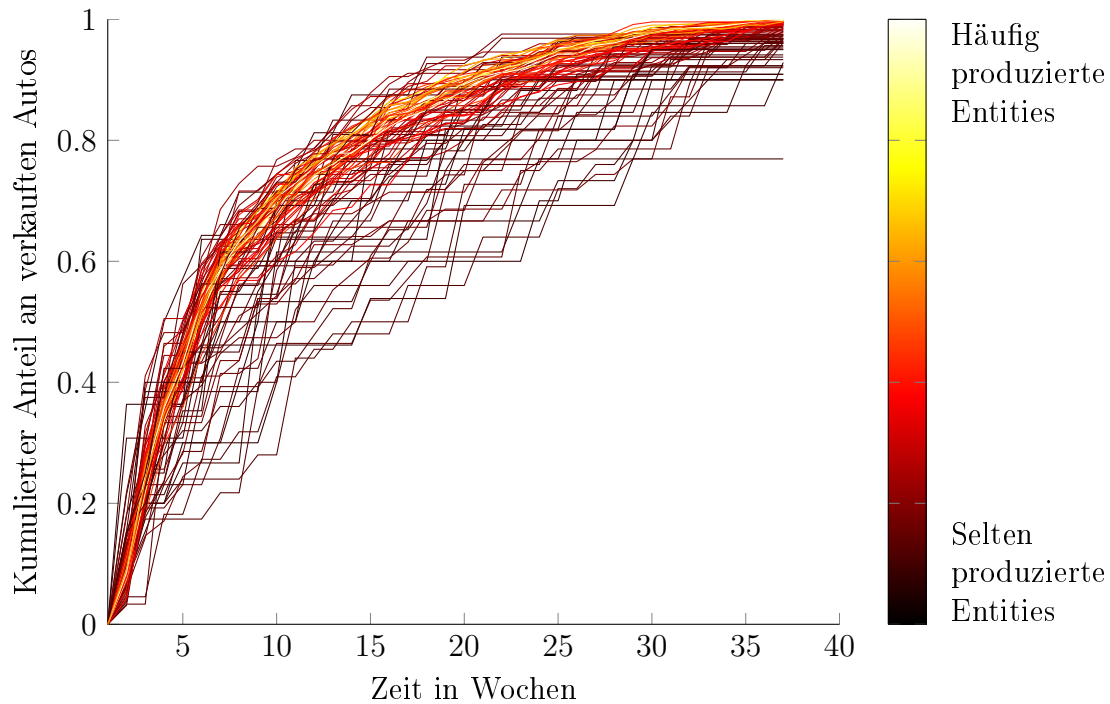


Abbildung 4.4: Zu sehen ist die Verkaufsgeschwindigkeit. Jede Linie symbolisiert eine Entity.

Zur Angabe einer möglichen Ursache für die unterschiedlichen Verkaufsgeschwindigkeiten, wurde jede Linie in Abhängigkeit der Anzahl der Autos, die ihrer Entity entsprechen und gebaut wurden, eingefärbt. Dunkelrote Linien gehören zu Entities mit wenig produzierten Autos und gelbe dementsprechend zu häufig produzierten. Sollten im späteren Verlauf Autos umgebaut werden, ist es nach obigem Ergebnis ratsam, Kombinationen von Autoteilen zu vermeiden, die zu wenig produzierten Entities führen. Diese besitzen teilweise eine hohe Wahrscheinlichkeit, dass Autos erst spät verkauft werden. Das zu einer Entity gehörige Qualitätsmaß ist proportional zu dem Integral der jeweiligen Kurve definiert worden. Je kleiner das Integral, desto schlechter ist die Verkaufbarkeit und somit das Qualitätsmaß abgeschätzt.

4.3 Entscheidungsbäume

Das im vorherigen Abschnitt eingeführte Qualitätsmaß reicht nicht aus, um die Anforderungen der Problemstellung zu erfüllen. Für Autos, die aufgrund einer Umplanung umgebaut werden, soll nicht nur eine gute Verkaufbarkeit, sondern auch ihre Ähnlichkeit zur ursprünglichen Spezifikation gewahrt werden. Der Begriff

der Ähnlichkeit ist in erster Linie subjektiver Natur und wird in diesem Abschnitt versucht mathematisch zu erfassen.

Grundlage für die Berechnung der Ähnlichkeit sind die in der Spezifikation kodierten Autoteile. Die Anzahl der unterschiedlichen Autoteile zweier Autos als Maß für ihre Unähnlichkeit zu verwenden, führt zu keiner realitätsnahen Beschreibung. Beispielsweise nimmt ein optionales Feature wie die dritte Kopfstütze auf der Rückbank weniger Einfluss auf die Gesamtgestalt des Autos als die Umplanung von einem 3- in einen 5-Türer. Es muss also mindestens eine Gewichtung der Autoteile angegeben werden, die den Einfluss auf die Gesamtgestalt des Autos bemisst.

Um sich von der Notwendigkeit eines realistisch definierten Distanzmaßes zu lösen, wurde die Idee der Entscheidungsbäume aufgegriffen. Mit Hilfe dieser wird versucht, das Problem der Beschreibung einer Ähnlichkeitsnachbarschaft für ein Auto ohne ein Ähnlichkeitsmaß anzugehen.

Aus Abschnitt 4.2 ist die Qualität für jedes Auto bekannt. Mittels Entscheidungsbäumen wird versucht anhand von Attributen Spezifikationen bzw. Entities in Gruppen einzuteilen, sodass diese im Durchschnitt ein Qualitätsmaß besitzen, welches stark von dem der anderen Gruppen abweicht. Am Ende bilden die entstandenen Gruppen die Ähnlichkeitsnachbarschaften der Autos. Muss ein Auto umgebaut werden, passiert dies ausschließlich in der ihm zugeordneten Gruppe.

Abbildung 4.5 zeigt einen auf diese Weise entstandenen Entscheidungsbaum. Um diesen präsentieren zu können, wurde die maximale Höhe des Baumes auf zwei eingeschränkt und lediglich zwei Klassen gebildet, die angeben, ob sich ein Auto gut oder schlecht verkaufen lässt. Zu sehen ist, dass in erster Instanz durch die Motorauswahl partitioniert wurde. Trotz der Möglichkeit nach Karosserie oder Getriebe zu unterscheiden, wurde im Fall einer weiteren Verzweigung nach der Modellvariante entschieden. Höchstwahrscheinlich nehmen die Modellvariante und der Motor größeren Einfluss auf die Gesamtgestalt des Autos als andere Teile der Entity.

An den Blattknoten wurden mehrere Informationen abgetragen. Als erstes die Anzahl der Entities, die sich durch die Entscheidungen, getroffen von Wurzel bis zum Blatt, charakterisieren lassen. Desweiteren sind die Anzahl der Autos, die der jeweiligen Entity entsprechen und gebaut wurden, und die prozentualen Anteile der beiden Klassen angegeben. Die eingeteilten Klassen weisen eine hohe Heterogenität auf. Ggf. würde die weitere Einteilung nach anderen Attributen diese Eigenschaft noch verstärken.

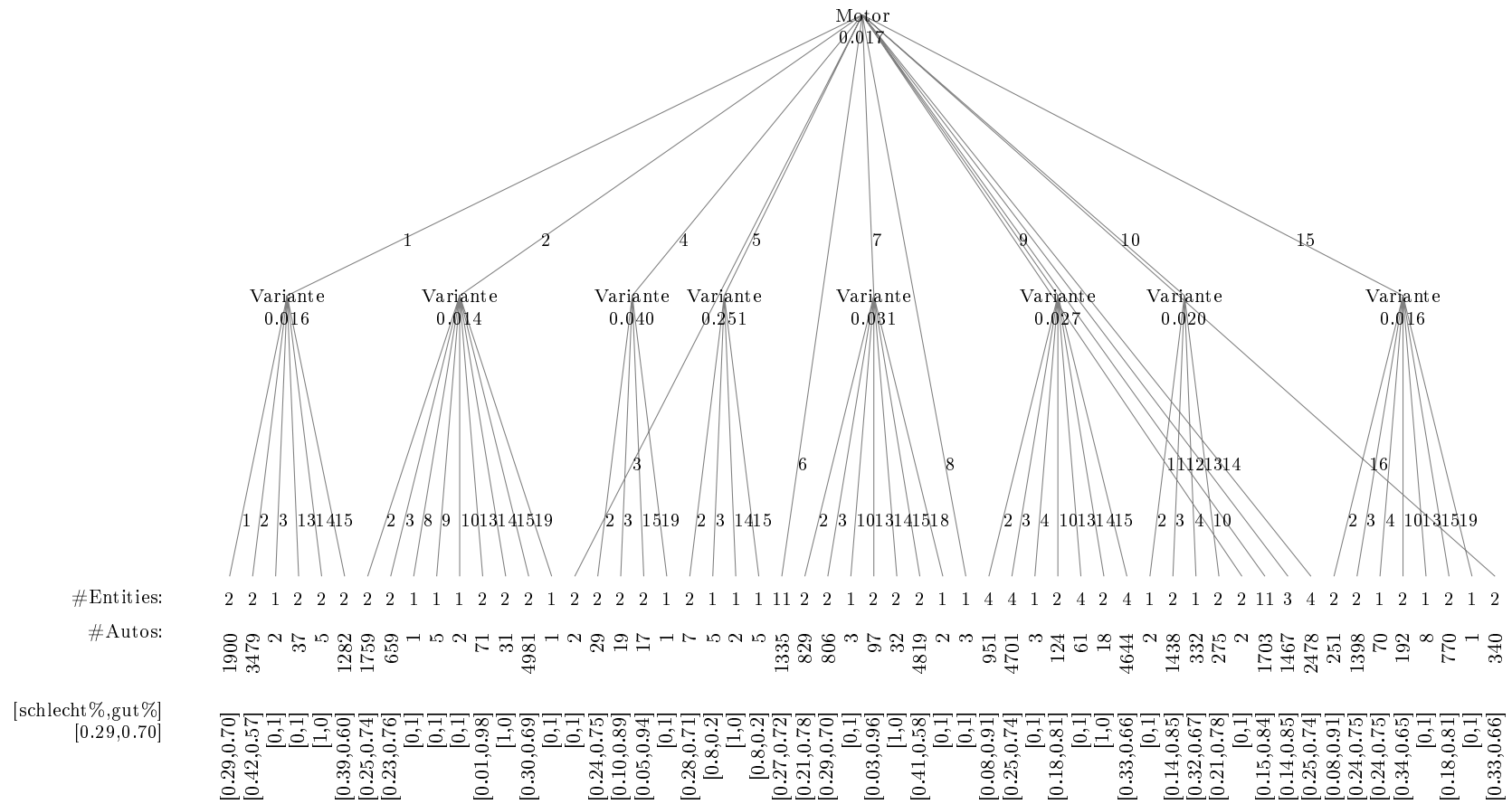


Abbildung 4.5: Ein auf die Entitykodierung der Ford Fiesta Daten angewendeter Entscheidungsbaum.

Die größte Schwäche von Entscheidungsbäumen wurde bereits im theoretischen Teil (Abschnitt 3.3.4) benannt. Entscheidungsbäume reagieren sensibel auf die Veränderung der Trainingsdaten. Wird der Betrachtungszeitraum geändert, verändert sich die Gestalt des Baumes radikal. Einen noch größeren Einfluss besitzt das Qualitätsmaß, auf Basis dessen der Entscheidungsbaum definiert ist. Gesucht ist eine robuste und nachvollziehbare Einteilung von Gruppen, die ähnlich zueinander sind und verifiziert werden können. Trotz der nachträglichen Gruppierung von ähnlichen Klassen war es nicht möglich, annähernd gleichbleibende Resultate zu erlangen. Da Entscheidungsbäume die Anforderungen nicht automatisiert erfüllen können, wurde für den weiteren Verlauf auf eine manuell definierte Distanzfunktion zwischen den Attributen zurückgegriffen.

5 Das Auto Planungs Modifikations Problem

Die anfangs in Kapitel 1 vorgestellte Problemstellung wird in diesem Kapitel erneut aufgegriffen und formal eingeführt. Um die aus den Daten extrahierten Informationen nutzen zu können und realitätsrelevante Einschränkungen zu respektieren, wird ein spezieller Übergangsgraph konstruiert, der einen Großteil dieser Informationen enthält und somit implizit gewährleistet. Das auf dem Übergangsgraphen operierende, ganzzahlige lineare Programm (ILP) sucht nach möglichen Umbauten, um ein gewünschtes Auto zu einem möglichst frühen Zeitpunkt zu bauen. Das ILP wird im späteren Verlauf an einigen Stellen erweitert bzw. eingeschränkt, damit vor allem realitätsnahe Lösungen produziert werden können.

Ein laufzeittechnisch wichtiger Aspekt ist die Auslagerung möglichst vieler Teilprozesse als Preprocess. Die nachfolgenden Prozesse sind, falls möglich, dahingehend optimiert.

5.1 Konstruktion des Übergangsgraphen

Als Grundlage für die weitere Modellierung wird ein gerichteter Übergangsgraph $G = (V, E, \text{spec})$ aus den Daten erzeugt.

Hierbei repräsentiert ein Knoten $v \in V$ eine Spezifikation, kodiert durch einen Binärvektor der Länge k entsprechend der maximalen Anzahl an Attributen, eines baubaren, im Folgenden als valide bezeichneten, Autos.

In anderen Kodierungsformen existieren Regeln, die alle validen Spezifikationen festlegen. Da, wie in Kapitel 2 beschrieben, diese Art der Beschreibung bei den SALES-Codes nicht vorhanden ist, muss die Knotenmenge V auf andere Weise konstruiert werden. Eine naheliegende Möglichkeit ist die Betrachtung aller produzierten Autos der letzten $|T_{\text{valid}}|$ Tage ausgehend vom aktuellen Betrachtungspunkt. Die Autos wurden bereits gebaut und können somit als valide angesehen werden. Offensichtlich handelt es sich bei V um eine echte Teilmenge aller möglichen Spezifikationen, was das später beschriebene Modell zwar kleiner macht, aber die Suche nach einer zulässigen Lösung erschwert.

Sei mit $V_{\text{BTS}} \subseteq V$ die Teilmenge beschrieben, die alle Knoten enthält, die nach

5 Das Auto Planungs Modifikations Problem

Definition 1 aus Kapitel 2 ein BTS-Auto repräsentieren.

Welche Attribute in dem zu repräsentierenden Auto verbaut sind, wird durch die Funktion spec angegeben, die in Definition 8 eingeführt wird.

Definition 8. Die Funktion $\text{spec}: V \rightarrow \{0, 1\}^k$ bildet jeden Knoten auf seinen binär kodierten Spezifikationsvektor ab. Die i -te Stelle $\text{spec}(v)_i$ ist für alle $v \in V$ und $i \in \{1, \dots, k\}$ wie folgt definiert:

$$\text{spec}(v)_i = \begin{cases} 1 & , \text{ falls Attribut } i \text{ in Auto repräsentiert durch } v \text{ verbaut ist} \\ 0 & , \text{ sonst.} \end{cases}$$

Eine gerichtete Kante $(v_1, v_2) \in E$ mit $v_1, v_2 \in V$ beschreibt die Veränderung von Spezifikation v_1 zu Spezifikation v_2 . Statt jeden Knoten mit allen anderen Knoten zu verbinden, werden nur Knoten verbunden, deren Spezifikationen ähnlich zueinander sind. Das bedeutet, dass die Autos, die von den Knoten repräsentiert werden, sich in maximal $\text{maxDiff} \in \mathbb{N}$ Attributen unterscheiden.

$$E = \{(v_1, v_2) \in V \times V \mid \|\text{spec}(v_1) - \text{spec}(v_2)\|_1 \leq \text{maxDiff}\} \quad (5.1)$$

Jeder Kante $e \in E$ kann ein Vektor $\text{spec}(e) \in \{-1, 0, 1\}^k$ zugeordnet werden, der die Differenz zwischen den Spezifikationen der Knoten beschreibt.

Definition 9. Für alle Kanten $(v_1, v_2) = e \in E$ ist der Vektor $\text{spec}(e) \in \{-1, 0, 1\}^k$ definiert als die komponentenweise Subtraktion der Spezifikation der Start- und Zielknoten.

$$\text{spec}(e)_i = \text{spec}(v_2)_i - \text{spec}(v_1)_i \quad \forall i \in \{1, \dots, k\}, \forall e \in E$$

Zu interpretieren ist eine -1 an der i -ten Stelle im Vektor durch das Vorhandensein von Attribut i in Auto v_1 , jedoch nicht in Auto v_2 . Attribut i würde demnach ausgebaut werden. Eine 1 verhält sich analog dazu. Eine 0 an der i -ten Stelle sagt lediglich etwas darüber aus, dass keine Veränderung bei diesem Attribut vorhanden ist. Ob v_1 und v_2 Attribut i besitzen oder nicht lässt sich daraus nicht schlussfolgern, kann aber aus $\text{spec}(v_1)_i$ abgelesen werden.

Beispiel 7 zeigt die Berechnung eines Spezifikationsvektors für eine Kante, ausgehend von den Spezifikationen des Start- und Zielknotens.

Beispiel 7. Seien zwei Knoten v_1 und v_2 und ihre Spezifikation gegeben.

$$\text{spec}(v_1) = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \text{spec}(v_2) = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

5 Das Auto Planungs Modifikations Problem

Für eine Schranke $\text{maxDiff} \geq 2$ existiert in G die Kante (v_1, v_2) .

$$\text{spec}((v_1, v_2)) = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$$

Die Kante symbolisiert folglich den Einbau von Attribut 2 und den Ausbau von Attribut 3. Entscheidet sich das im nächsten Abschnitt eingeführte Modell für die Nutzung der Kante, wird Auto v_2 anstelle von v_1 gebaut.

Um für einen Übergangsgraphen einen validen Umbau definieren zu können, wird vorbereitend die Menge der eingehenden sowie ausgehenden Kanten eines Knotens in Definition 10 definiert.

Definition 10. Für einen Übergangsgraphen $G = (V, E, \text{spec})$ sei mit $\delta^-(v)$ für alle $v \in V$ die Menge der eingehenden Kanten in v und mit $\delta^+(v)$ die ausgehenden Kanten aus v beschrieben.

Durch das Verhältnis der Anzahl der ein- und ausgehenden Kanten kann ein valider Umbau in Definition 11 beschrieben werden. Im Allgemeinen müssen sich in einem Knoten die Anzahl der eingehenden wie ausgehenden Kanten entsprechen. Ausgenommen davon sind unter anderem Knoten der Menge V_{BTS} , da diese umgebaut werden können und somit mehr ausgehende Kanten besitzen können. Den zweiten Spezialfall bilden alle Knoten, die nach dem Umbau geplant sind zu bauen. Insbesondere der Knoten $v^* \in V$, der das Wunschauto repräsentiert. Diese Knoten besitzen mehr eingehende als ausgehende Kanten.

Definition 11. Ein valider Umbau auf einem Übergangsgraphen $G = (V, E, \text{spec})$ entspricht einer Auswahl an Kanten $E' \subseteq E$, sodass für alle Knoten $v \in V$ die Differenz von ausgehenden zu eingehenden Kanten der Menge E' beschränkt ist. Formal muss gelten:

$$|\delta^+(v) \cap E'| - |\delta^-(v) \cap E'| \leq \begin{cases} 0 & , \text{ falls } v \notin V_{\text{BTS}} \\ 1 & , \text{ falls } v \in V_{\text{BTS}}. \end{cases} \quad (5.2)$$

Gebaut wird ein Auto, falls sein repräsentativer Knoten Ungleichung 5.2 nicht mit Gleichheit erfüllt. Das Ziel ist, die Kantenwahl so zu treffen, dass diese Eigenschaft für einen Knoten v^* erfüllt ist, welcher dem gewünschten Auto eines Kunden entspricht. Beispiel 8 zeigt den Unterschied zwischen einem validen und nicht validen Umbau.

5 Das Auto Planungs Modifikations Problem

Beispiel 8. Seien in diesem Beispiel die Knoten der Menge V als Kreise dargestellt und die Teilmenge V_{BTS} durch graue Knoten unterschieden. Die eingezeichneten Kanten entsprechen bereits den ausgewählten Kanten, die einen Umbau kodieren. Auf der linken Seite ist ein valider Umbau zu sehen. Der graue Knoten repräsentiert ein BTS-Auto und kann deshalb eine ausgehende Kante mehr besitzen, als er eingehende Kanten besitzt. Für den mittleren Knoten gilt, dass sich die Anzahl der ein- und ausgehenden Kanten entspricht. Der rechte Knoten besitzt mehr eingehende Kanten. Das zugehörige Auto war nicht geplant zu bauen, wird aber nach dem durch die Kanten dargestellten Umbau gebaut. Der Umbau auf der rechten Seite ist nicht valide, weil ein Knoten der nicht V_{BTS} angehört mehr ausgehende als eingehende Kanten besitzt.



Ein solcher Übergangsgraph wird für jeden Tag in dem Suchzeitraum T_{such} benötigt, an dem das Modell nach einem möglichen, validen Umbau suchen soll.

Definition 12. Sei mit T_{such} der Zeitraum beschrieben, in dem nach einem validen Umbau gesucht wird.

Gesucht wird innerhalb eines Produktionstages, da bei einer Betrachtung von mehreren Tagen, die Umplanung eine Lagerung von Autoteilen zufolge hätte. Dies würde der Problemstellung widersprechen. Im Folgenden wird diese Einschränkung *Attributerhaltung* genannt und in Definition 13 eingeführt.

Definition 13. Die Attributerhaltung gilt genau dann, falls für alle Attribute i gilt, dass diese genauso oft ein- wie ausgebaut werden.

$$\sum_{e \in E'} \text{spec}(e)_i = 0 \quad \forall i \in \{1, \dots, k\}$$

Nach der ausformulierten Beschreibung des Auto Planungs Modifikations Problems in Kapitel 1 kann das Problem an dieser Stelle formal eingeführt werden.

AUTO PLANUNGS MODIFIKATIONS PROBLEM

Eingabe: Übergangsgraph $G = (V, E, \text{spec})$, Menge $V_{\text{BTS}} \subseteq V$ und Knoten $v^* \in V$
 Frage: Existiert ein valider Umbau auf G mit V_{BTS} , sodass die Attributerhaltung gewährleistet ist und v^* gebaut wird.

Abbildung 5.1: Das Auto Planungs Modifikatin Problem, kurz APMP.

5.2 Modellierung des APMP

Die Hauptaufgabe des Modells besteht darin, Kanten auszuwählen, sodass das gesuchte Auto, repräsentiert durch Knoten v^* , gebaut wird und weder Attribute gelagert, noch aus dem Lager entnommen werden müssen. Dazu wird für jede Kante $(v_1, v_2) = e \in E$ eine Entscheidungsvariable $x_e \in \{0, 1\}$ eingeführt, die genau dann 1 ist, falls die Kante ausgewählt wurde und somit der Umbau von v_1 zu v_2 stattfindet.

Jeder Knoten, der die Bedingung in Definition 11 nicht mit Gleichheit erfüllt, repräsentiert ein Auto, das nach dem Umbau für den Bau geplant ist. Eine positive ganzzahlige Entscheidungsvariable $y_v \in \mathbb{Z}_{\geq 0}$ für alle $v \in V$ stellt diesen Zustand dar. Dadurch ist es möglich die nach dem Umbau resultierenden Autos anhand des in Abschnitt 4.2 eingeführten Qualitätsmaßes zu bewerten.

An dieser Stelle werden die beiden Entscheidungsvariablen verknüpft. Die ausgewählte Kantenmenge E' , kodiert durch Entscheidungsvariablen $x_e = 1$ für alle $e \in E'$, nimmt Einfluss auf die geplant zu bauenden Autos, welche durch $y_v \geq 1$ festgelegt werden. Um diesen Zusammenhang in einer Nebenbedingung ausdrücken zu können, wird eine Indikatorfunktion $\text{bts}(v) \forall v \in V$ definiert.

Definition 14. Die Indikatorfunktion $\text{bts}(v) : V \rightarrow \{0, 1\}$ ist definiert durch:

$$\text{bts}(v) = \begin{cases} 1 & , \text{ falls } v \in V_{\text{BTS}} \\ 0 & , \text{ ansonsten.} \end{cases}$$

Durch folgende Nebenbedingung lässt sich der Zusammenhang zwischen den beiden Entscheidungsvariablen und der Menge der BTS-Autos ausdrücken:

$$\sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e + y_v = \text{bts}(v) \quad \forall v \in V. \quad (5.3)$$

Falls das durch v dargestellte Auto weder geplant war noch gebaut werden soll, muss sich die Anzahl der eingehenden und ausgehenden Kanten entsprechen. Soll

5 Das Auto Planungs Modifikations Problem

das Auto stattdessen nach den Umbauten produziert werden, muss nach Konstruktion $y_v = 1$ gesetzt werden. Dies wird dadurch gewährleistet, dass es mindestens eine eingehende Kante mehr geben muss, als es ausgehende Kanten gibt. Der resultierende negative Wert (-1) in der zweiten Summation wird durch y_v ausgeglichen. Im umgekehrten Fall, dass es sich bei dem Auto um ein BTS-Auto handelt und die rechte Seite somit 1 entspricht, erfolgt der Ausgleich entweder durch eine ausgehende Kante (erste Summation) oder durch die Entscheidungsvariable y_v . Im letzten Fall wird das Auto sowohl geplant als auch produziert. Es wurden alle vier Fälle, möglich durch die Belegung der Entscheidungsvariable y_v und den Fall eines geplanten, wie ungeplanten Autos, betrachtet.

Als nächstes muss die Kantenwahl weiter eingeschränkt werden. Insgesamt sollen die ausgewählten Kanten der Attributerhaltung genügen. Das heißt, dass für jedes Attribut die Anzahl der Kanten, in denen das Attribut ausgebaut wird, der Anzahl Kanten entsprechen muss, in denen das Attribut eingebaut wird. Für das i -te Attribut mit $i \in \{1, \dots, k\}$ lässt sich dies durch den berechneten Spezifikationsvektor an den Kanten des Übergangsgraphen ausdrücken:

$$\sum_{e \in E} \text{spec}(e)_i \cdot x_e = 0 \quad \forall i \in \{1, \dots, k\}. \quad (5.4)$$

Im aktuellen Zustand ist es dem Modell möglich die Nebenbedingungen 5.3 und 5.4 durch die Auswahl keiner Kante zu erfüllen. Es würde somit kein Umbau stattfinden. Erzwungen wird dieser durch den Bedarf an einem speziellen Knoten v^* . Das repräsentierte Auto soll, falls möglich, gebaut werden.

$$y_{v^*} = 1. \quad (5.5)$$

Im einfacheren Fall gilt $\text{bts}(v^*) = 1$. Dann ist es nach Ungleichung 5.3 nicht nötig Kanten auszuwählen. Falls das Auto nicht geplant ist, muss es mindestens eine eingehende Kante in den Knoten v^* geben. In diesem Fall zwingt Ungleichung 5.4 das Modell dazu weitere Kanten auszuwählen, um die Attributerhaltung zu gewährleisten.

Definition 15. Das ganzzahlige lineare Programm *ILP1* besteht aus Ungleichungen 5.3, 5.4 und 5.5.

subject to $\sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e + y_v = \text{bts}(v)$	$\forall v \in V$
$\sum_{e \in E} \text{spec}(e)_i \cdot x_e = 0$	$\forall i \in \{1, \dots, k\}$
$y_{v^*} = 1$	
$x_e \in \{0, 1\}$	$\forall e \in E$
$y_v \in \mathbb{Z}_{\geq 0}$	$\forall v \in V$

Für eine gegebene Instanz des APMPs findet ILP1, welches in Definition 15 spezifiziert worden ist, genau dann eine zulässige Lösung, wenn es sich um eine Ja-Instanz des APMPs handelt.

5.3 Modell- und Grapherweiterungen

Im momentanen Zustand des Graphen und des Modells entspricht eine gefundene Lösung einem validen Umbau. Als nächster Schritt werden weitere Anforderungen an Graph und Modell gestellt, um eine gewisse Realitätsnähe zu gewährleisten.

5.3.1 Ähnlichkeitsanforderung

Um den Ähnlichkeitsanforderungen zu genügen, wird der Übergangsgraph G in Abhängigkeit vom Suchtag $t \in T_{\text{such}}$ weiter eingeschränkt. Aufgrund der Konstruktion von E können durch die Auswahl einer Kante höchstens maxDiff Autoteile umgebaut werden. In der aktuellen Formulierung ist jedoch die Auswahl eines Pfades möglich, wodurch sich die maximale Beschränkung der unterschiedlichen Teile umgehen lässt. Verhindert wird dies durch die Einschränkung von G auf die Knotenmenge V_{BTS} und deren direkte Nachbarschaft.

Definition 16. Für einen Übergangsgraphen G ist die Nachbarschaft $N(v)$ für $v \in V$ definiert als:

$$N(v) = \{v' \in V \mid \exists (v, v') \in E\} \quad (5.6)$$

Der auf diese Weise entstehende Graph wird mit G_t mit $t \in T_{\text{such}}$ bezeichnet. Abbildung 5.2 zeigt die Konstruktion ausgehend von einem Übergangsgraphen und der Knotenmenge V_{BTS} .

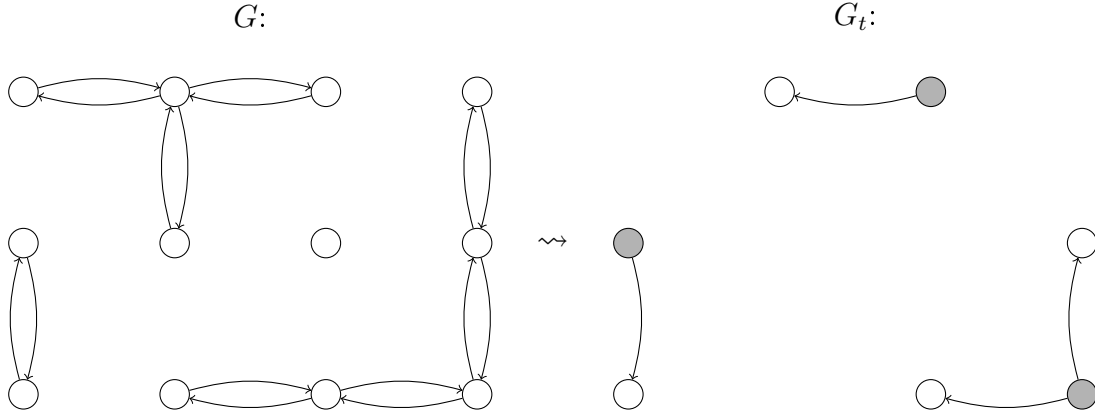


Abbildung 5.2: Auf der linken Seite ist ein möglicher Übergangsgraph G zu sehen. Auf der rechten Seite ist der durch V_{BTS} und deren Nachbarschaft induzierte Graph G_t zu sehen. Definiert wird dieser durch die grauen Knoten, welche BTS-Autos repräsentieren. Zusätzlich sind alle Nachbarschaftsknoten enthalten.

Diese starke Einschränkung an den Graphen G_t hat den Nachteil, dass der Knoten eines angefragten Autos oftmals nicht in G_t liegt. Das Modell kann demnach keine zulässige Lösung finden. Abschnitt 5.3.2 liefert eine Lösung für dieses Problem.

5.3.2 Ähnlichkeitsbeschränkung für den Zielknoten

Die Ähnlichkeitsbeschränkung durch die Kanten ist vor allem deshalb eingeführt worden, um die, durch die Attributerhaltung erzwungenen Umbauten möglichst gering zu halten. Diese Forderung ist unnötig für das gewünschte Auto, da dieses aufgrund der Problemstellung nach dem Umbau ein BTO-Auto ist und somit verkauft ist.

Deshalb kann der Knoten v^* , bei gegebener Anfrage, dem Graphen G_t hinzugefügt werden. Desweiteren wird die Kantenmenge E durch Kanten erweitert, die einen Umbau von allen Knoten $v \in V_{\text{BTS}}$ zu v^* möglich machen.

$$\tilde{E} = E \cup \{(v, v^*) \mid \forall v \in V_{\text{BTS}}\}$$

Diese Erweiterung wird genau dann zu einem Problem, falls das Modell aufgrund der Attributerhaltung mehrere der neu eingeführten Kanten auswählt. Verhindert wird dies durch eine weitere Nebenbedingung, genannt *Mehrfachnutzung*.

$$\sum_{(v, v^*)=e \in \tilde{E}} x_e \leq 1 \quad \forall v \in V_{\text{BTS}}; \quad (5.7)$$

5 Das Auto Planungs Modifikations Problem

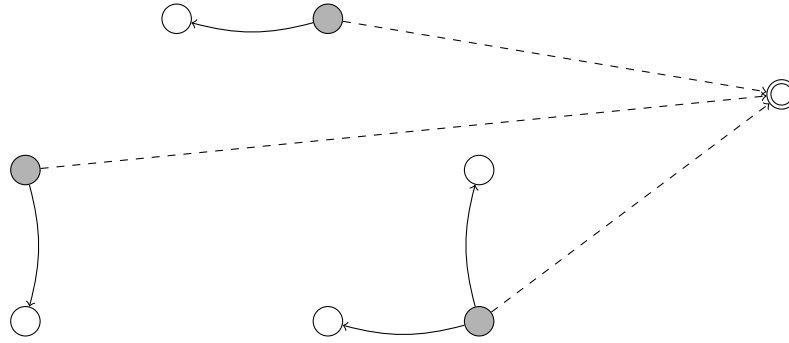


Abbildung 5.3: Die Konstruktion hebt die Ähnlichkeitsbeschränkung für den Zielknoten (durch den Doppelkreis dargestellt) auf.

Die Erweiterung des Graphen, wie in Abbildung 5.3 gezeigt, erleichtert dem Modell das Finden einer zulässigen Lösung. Gleichzeitig muss der Graph in Abhängigkeit von Knoten v^* durch einen Knoten, mehrere Kanten und das Modell durch dazugehörige Entscheidungsvariablen und Nebenbedingungen ergänzt werden. All dies wirkt sich negativ auf die Laufzeit für die Beantwortung einer Anfrage aus, da ein beträchtlicher Teil nicht mehr als Preprocess ausgelagert werden kann (siehe Übersicht in Abbildung 5.6).

5.3.3 Multikriterielle Zielfunktion

Als Zielfunktion kommen verschiedene Varianten und deren Kombination in Frage. Der wohl intuitivste Ansatz läge darin, die Anzahl der Autos, die umgebaut werden, zu minimieren.

$$\min \sum_{v \in V_{\text{BTS}}} 1 - y_v \quad (5.8)$$

Alle Knoten der Menge V_{BTS} deren y -Variable gleich 0 gesetzt werden, repräsentieren Autos, die zwar geplant wurden, aber nach dem Umbau nicht mehr gebaut werden.

Nicht betrachtet wird in diesem Fall der Grad der Veränderung, der an einem Auto vorgenommen wird. Dieser wird durch den Spezifikationsvektor an den Kanten beschrieben und lässt sich wie folgt in der Zielfunktion ausdrücken:

$$\min \sum_{e \in E} \|\text{spec}(e)\|_1 \cdot x_e \quad (5.9)$$

Diese Formulierung hat auf einem Übergangsgraphen G die Schwäche, dass aufgrund der heuristisch und somit nicht vollständigen Knotenmenge V oftmals nicht der direkte Weg zwischen zwei Knoten gewählt werden kann. Daher ist es möglich, dass das gleiche Attribut auf dem Weg zwischen zwei Knoten mehrmals ein- und

5 Das Auto Planungs Modifikations Problem

ausgebaut wird. Die gefundene Lösung entspricht zwar den Anforderungen der Nebenbedingungen, besitzt jedoch einen künstlich erhöhten Zielfunktionswert bei der Verwendung von Zielfunktion 5.9. Beispiel 9 zeigt einen solchen Fall.

Beispiel 9. Sei jeder Knoten durch vier Attribute spezifiziert und Kanten eingeführt, die der Schranke $\text{maxDiff} = 1$ genügen. Die obere Reihe der Knoten in Abbildung 5.4 entspricht den als validen gekennzeichneten Spezifikationen durch Betrachtung der Historie innerhalb der Datenbank. Ausgehend vom linken Knoten werden fünf Kanten benötigt um die rechte Spezifikation zu erreichen (durchgezogene Pfeile).

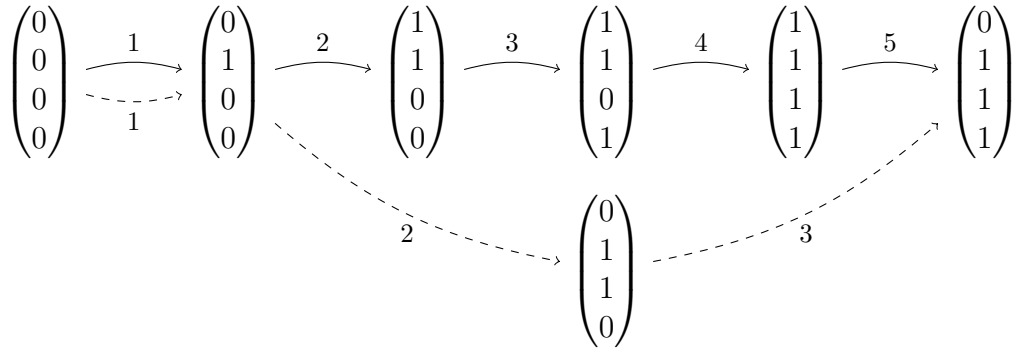


Abbildung 5.4: Mehrfacher Ein- und Ausbau von Attributen in einer zulässigen Lösung.

Wird für den unteren Knoten mit der Spezifikation $(0, 1, 1, 0)^T$ angenommen, dass er ebenfalls in der Knotenmenge liegt, so ist der Übergang durch drei Kanten möglich (gestrichelte Pfeile).

Der Ein- und Ausbau vom ersten Attribut findet lediglich aufgrund der eingeschränkten Knotenmenge statt.

Dieses Problem kann durch die Verwendung des eingeschränkten Übergangsgraphen G_t behoben werden, da dort keine Auswahl von Pfaden möglich ist.

Ein weiteres erstrebenswertes Ziel ist die Betrachtung der erzwungen umgebauten Autos. Während der gewünschte Knoten v^* in jedem Fall in einer zulässigen Lösung vorkommt, können sich die durch Ungleichung 5.4 erzwungenen Umbauten unterscheiden. Die daraus resultierenden Autos sind bereits mit einem Qualitätsmaß q_v versehen worden, nach welchem maximiert werden kann:

$$\min \sum_{v \in V} (1 - q_v) \cdot y_v \quad (5.10)$$

Die Gewichtung der unterschiedlichen Zielfunktionen ist eine komplexe Aufgabe. Generell ist das primäre Ziel das Finden einer zulässigen Lösung, da eine maximale

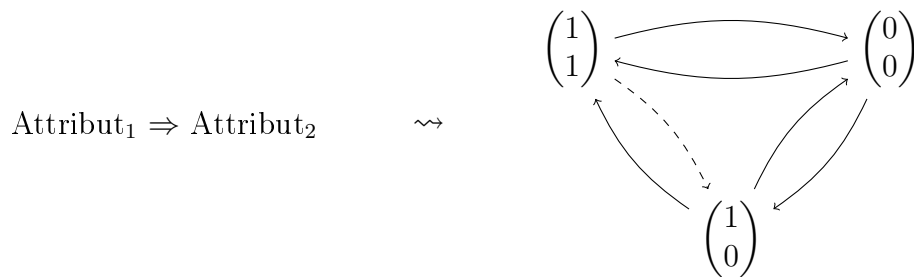
Abweichung zwischen Spezifikationen bereits durch die Ähnlichkeitsanforderung aus Abschnitt 5.3.1 gewährleistet ist. Zielfunktion 5.8 und 5.9 verhindern in jedem Fall auch einzeln die übermäßige Auswahl an Kanten, um kleinere Verbesserungen in der Qualität zu erreichen.

5.3.4 Korrelation zwischen Attributen

Trotz des Qualitätsmaßes in der Zielfunktion ist es möglich Autos zu erhalten, die als unverkaufbar gelten. Bedingt durch die Konstruktion der Knotenmenge existiert für jeden Kundenwunsch im Zeitraum T_{valid} ein Knoten. Vor allem bei BTO-Autos handelt es sich um teilweise extravagante Kombinationen an Attributen, die für die erzwungenen BTS-Autos verhindert werden sollen. Gesucht werden daher häufige Kombinationen von und Korrelationen zwischen den Attributen.

An dieser Stelle wird sich die in Abschnitt 3.2 eingeführte Assoziationsanalyse zu Nutze gemacht. Gefundene Regeln der Form $\text{Attribut}_i \Rightarrow \text{Attribut}_j$, die eine gewisse Vertraulichkeit besitzen, nehmen Einfluss auf den Übergangsgraphen. Aus Knoten, die sowohl Attribut_i als auch Attribut_j enthalten, werden alle Kanten gestrichen, die einen Ausbau von Attribut_j und die Erhaltung von Attribut_i beschreiben.

Beispiel 10. In diesem Beispiel wird der mögliche Einfluss der Assoziationsanalyse auf den Übergangsgraphen dargestellt. Die gestrichelte Kante wird aufgrund von vorhandenen Korrelationen zwischen Attribut_1 und Attribut_2 gestrichen oder ihre Benutzung durch Kosten bestraft.



Die gefundene Assoziationsregel auf der linken Seite sagt aus, dass falls Attribut_1 in einem Auto eingebaut ist, zu einer hohen Wahrscheinlichkeit auch Attribut_2 eingebaut wurde. Deshalb wird die gestrichelte Kante entweder entfernt oder ihre Benutzung bestraft. Die Rückrichtung ist hingegen erwünscht. Ein Ausbau beider Attribute verletzt die Regel nicht.

5.3.5 Lager

Während durch die Voraussetzung der Attributerhaltung die Notwendigkeit eines Lagers nicht vorhanden ist, entspricht dies nicht der Realität. Gewisse Attribute lassen sich sowohl lagern, als auch aus dem Lager entnehmen. Diese Tatsache macht das Finden einer zulässigen Lösung durch das Modell einfacher, da Ungleichung 5.4 relaxiert werden kann.

Angenommen vom i -ten Attribut lassen sich n_i Teile lagern, dann kann anstatt Gleichheit Folgendes gefordert werden:

$$-n_i \leq \sum_{e \in E} \text{spec}(e)_i \cdot x_e \leq 0 \quad \forall i \in \{1, \dots, k\} \quad (5.11)$$

Analog dazu ist der Fall, dass eine bestimmte Anzahl m_a an Teilen aus dem Lager entnommen werden kann. Dann dient Anzahl m_i als obere Schranke für Ungleichung 5.11.

Im Extremfall gibt es Teile die überhaupt nicht der Attributerhaltung unterliegen. Das Nicht-Erzeugen der Nebenbedingung für diese Attribute ist die eleganteste Methode, da die Information bereits zum Zeitpunkt der Graph- und Modellkonstruktion bekannt ist.

Unter der Annahme, dass diese Attribute keinen Einfluss auf die Ähnlichkeit zwischen Autos haben, ließe sich der Übergangsgraph deutlich vereinfachen. Bei der Ermittlung der eindeutigen Spezifikationen, aus denen später die Knotenmenge V entsteht, kann der Spezifikationsvektor auf die Attribute verkleinert werden, die der Attributerhaltung oder Ähnlichkeitsbeschränkung unterliegen.

$\begin{aligned} &\text{minimize} && w_1 \cdot \sum_{e \in E} \ \text{spec}(e)\ _1 \cdot x_e + w_2 \cdot \sum_{v \in V} (1 - q_v) \cdot y_v \\ &\text{subject to} && \sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e + y_v = \text{bts}(v) \\ &&& n_i \leq \sum_{e \in E} \text{spec}(e)_i \cdot x_e \leq m_i \\ &&& y_{v^*} = 1 \\ &&& x_e \in \{0, 1\} \\ &&& y_v \in \mathbb{Z}_{\geq 0} \end{aligned}$	$\begin{aligned} &&& \forall v \in V \\ &&& \forall i \in \{1, \dots, k\} \\ &&& \\ &&& \forall e \in E \\ &&& \forall v \in V \end{aligned}$
--	---

Abbildung 5.5: Seien $w_1, w_2 \in [0, 1]$ mit $w_1 + w_2 = 1$ die Gewichte der Zielfunktion. Zulässige Lösungen dieser ILP-Formulierung liefern Antworten auf das APMP und genügend weiteren realitätsrelevanten Anforderungen.

5.4 Übersicht

Das vollständige Modell mit einer Gewichtung der Zielfunktionen 5.9 und 5.10 ist in Abbildung 5.5 zu sehen und wird kurz mit *ILP1* bezeichnet.

Die Größe des Modells hängt maßgeblich von der Größe des Übergangsgraphen ab. Die Knotenanzahl und somit die Anzahl der y -Variablen wird vom Parameter T_{valid} gesteuert. Je weiter die Vergangenheit betrachtet wird, desto mehr Entscheidungsvariablen müssen generiert werden. Die Anzahl der Kanten und somit die Anzahl der x -Variablen wird durch die Ähnlichkeitsschranke maxDiff festgelegt. Durch die Assoziationsanalyse werden abhängig von der Vertraulichkeitsschranke der Regeln Kanten gestrichen. Die Anzahl der Attributerhaltungsnebenbedingungen ist durch die maximale Anzahl an Attributen k begrenzt und kann durch die Einführung eines Lagers für bestimmte Attribute lediglich verringert nicht aber erhöht werden.

Nachdem die einzelnen Konstruktions- und Lösungsschritte detailliert beschrieben worden sind, wird nun der Prozess als Ganzes betrachtet. Auf einem höheren Level gibt Abbildung 5.6 einen Überblick über den gesamten Prozess.

Anfangen in einem nicht laufzeitkritischen Zeitraum, beispielsweise während des Übergangs von einem zum anderen Produktionstag, wird die Menge der validen Spezifikationen ermittelt und somit die Knoten des Übergangsgraphen festgelegt. Kanten werden zwischen Spezifikationen erzeugt, falls diese einer vorher festge-

5 Das Auto Planungs Modifikations Problem

legten Ähnlichkeitsschranke genügen. Falls der Ein- bzw. Ausbau von Attributen, kodiert auf einer Kante, während der Assoziationsanalyse gefundenen Korrelationen widerspricht, werden diese Kanten im Anschluss gelöscht.

Ebenfalls vorher festgelegt ist der Suchzeitraum T_{such} . Für jeden Tag in diesem Suchzeitraum wird in Abhängigkeit der an diesem Tag geplanten Autos ein spezieller Übergangsgraph konstruiert, welcher als Grundlage für ein ILP dient.

Das ILP beinhaltet eine multikriterielle Zielfunktion. Das allgemeine Ziel ist die minimale Veränderung geplanter Autos und die Maximierung ihrer Verkaufbarkeit. Die Nebenbedingungen gewährleisten sowohl den gewünschten Zusammenhang zwischen Kanten und Knoten, als auch die Attributerhaltung. All dies kann als Preprocessing vorbereitet werden.

Durch die Nachfrage eines bestimmten Autos kann der zugehörige Bedarf dem Modell hinzugefügt werden. Liegt der Knoten, der das gewünschte Auto repräsentiert, nicht im Graphen, müssen ein neuer Knoten und neue Kanten hinzugefügt werden. Infolgedessen benötigt das ILP weitere Entscheidungsvariablen, die erst zur Laufzeit hinzugefügt werden können.

Findet das Modell eine zulässige Lösung, sind die ausgewählten Kanten als Umbauten zu verstehen. Das Auto, welches durch den Ausgangsknoten dargestellt wird, war geplant und wird in dieser Lösung nicht mehr gebaut. Stattdessen wird das Auto mit der Spezifikation des Zielknotens gebaut.

Produktionszeitpunkt

T_{valid}

T_{such}

Graphkonstruktion 5.1

G :

Assoziationsanalyse 5.3.4

Ähnlichkeitsanforderung 5.3.1

$t_1 \in T_{\text{such}}$

$t_2 \in T_{\text{such}}$

$\dots t_n \in T_{\text{such}}$

BTS 2.2

G_{t_1} :

G_{t_2} :

Ähnlichkeitsbeschränkung Zielknoten 5.3.2

Preprocessing

Anfrage

ILP1_{t₁}

Zielfunktion 5.8, 5.9, 5.10
s.t.

- Zusammenhang 5.3
- Attributerhaltung 5.4

- Bedarf 5.5

ILP2_{t₂}

Zielfunktion 5.8, 5.9, 5.10
s.t.

- Zusammenhang 5.3
- Attributerhaltung 5.4
- Bedarf 5.5
- Mehrfachnutzung 5.7

Lager 5.3.5

zulässige
Lösungen:

Wähle Lösung in Abhängigkeit von:

- Zielfunktionswert
- Zeitpunkt

Abbildung 5.6: Technische Übersicht des gesamten Lösungsprozesses.

5.5 Alternative Formulierung

Die Einschränkung lediglich die Nachbarschaft der BTS-Knoten zu betrachten (siehe Abschnitt 5.3.1) macht eine Matching-ähnliche Formulierung auf einem bipartiten Graphen möglich. Der Ansatz besteht darin Kanten einzuführen, die auf Kopien von Knoten zeigen und deshalb keinen Umbau repräsentieren.

Definition 17. Sei der bipartite Graph als $G_M = (V_1 \dot{\cup} V_2, E_M, \text{spec})$ definiert. V_1 enthält Knoten, deren Spezifikationen den Knoten aus V_{BTS} entsprechen. V_2 enthält einen Knoten für jede valide Spezifikation und entspricht somit der Knotenmenge V . Die Funktion spec beschreibt weiterhin den Umbau kodiert auf einer Kante und die für die Kantenmenge E_M gilt:

$$E_M = \{(v_1, v_2) \in V_1 \times V_2 \mid \|\text{spec}(v_1) - \text{spec}(v_2)\|_1 \leq \text{maxDiff}\}.$$

Aufgrund der Konstruktion existiert für jeden Knoten in V_1 ein Knoten in V_2 , der die gleiche Spezifikation besitzt und deshalb nachfolgend als *Kopie* bezeichnet wird.

Zusätzlich zur ursprünglichen Kantenmenge kommen demnach Kanten hinzu, die einen Knoten aus V_1 mit dessen Kopie in V_2 verbinden. Die beiden Knoten besitzen die gleiche Spezifikation, wodurch es sich bei dem Spezifikationsvektor der Kante um den Nullvektor handelt.

Das Qualitätsmaß, welches zuvor an den Knoten abgetragen war, lässt sich nach diesen Umformungen an den Kanten abtragen. Berechnet wird das Qualitätsmaß an der Kante durch die Differenz des Qualitätsmaßes der Knoten:

$$q_e = q_{v_2} - q_{v_1} \quad \forall (v_1, v_2) = e \in E_M$$

Durch die Einführung von Kanten, die keinerlei Veränderungen darstellen, lässt sich für alle Knoten der Knotenmenge V_1 fordern, dass diese genau eine ausgehende Kante besitzen:

$$\sum_{e \in \delta^+(v)} x_e = 1 \quad \forall v \in V_1 \tag{5.12}$$

Der Bedarf an einem Knoten $v^* \in V_2$ kann durch das Festlegen der Anzahl der eingehenden Kanten ausgedrückt werden:

$$\sum_{e \in \delta^-(v^*)} x_e = 1 \tag{5.13}$$

5 Das Auto Planungs Modifikations Problem

Ebenfalls durch die Nullvektoren als Spezifikationen der Kanten ermöglicht ist die Forderung nach der Attributerhaltung auf allen ausgewählten Kanten. Die Forderung wird exakt durch Ungleichung 5.4 formuliert, welche der Vollständigkeit halber an dieser Stelle nochmals aufgeführt ist:

$$\sum_{e \in E} \text{spec}(e)_i \cdot x_e = 0 \quad \forall i \in \{1, \dots, k\}$$

Die Zielfunktionen 5.9 und 5.10 lassen sich problemlos adaptieren. Zielfunktion 5.8, die die Anzahl der umgebauten Autos minimiert, lässt sich aufgrund der fehlenden Entscheidungsvariable y_v nicht direkt übernehmen. Allerdings ist bereits bei der Konstruktion bekannt, dass alle Kanten, deren Spezifikationsvektor nicht der Nullvektor ist, einen Umbau zur Folge haben:

$$\min \sum_{\substack{e \in E_M: \\ \text{spec}(e) \neq 0}} x_e \quad (5.14)$$

Die vollständige ILP-Formulierung ist in Abbildung 5.7 zu sehen und wird kurz *ILP2* genannt.

Es sei angemerkt, dass es sich bei zulässigen Lösungen im Allgemeinen nicht um Matchings handelt. Dazu müsste zusätzlich gefordert werden, dass für alle Knoten der Menge V die Anzahl der eingehenden Kanten kleiner gleich 1 ist. Haben 2 Kanten denselben Zielknoten, werden 2 unterschiedlich geplante Autos in das gleiche Auto umgebaut. Dies soll nach den Anforderungen des Problems möglich sein.

Im Folgenden wird der Zusammenhang zwischen der Entscheidungsvariante von ILP1 und ILP2 hergestellt.

Satz 5.1. *Für eine Instanz des APMP besitzt das ILP1 auf dem Übergangsgraphen G genau dann eine zulässige Lösung, wenn ILP2 auf Graphen G_M eine zulässige Lösung besitzt.*

Beweis. Falls für den Bedarfsknoten $v^* \in V$ gilt $\text{bts}(v^*) = 1$, gibt es eine zulässige Lösung in beiden Formulierungen. Betrachtet wird der Fall $\text{bts}(v^*) = 0$.

Sei eine zulässige Lösung von ILP1 gegeben, dann existiert eine Kante $(v, v^*) = e \in E$ mit $v \in V$, $\text{bts}(v) = 1$ und $x_e = 1$. Die Kante e ist in der Menge der ausgewählten Kanten

$$E^* = \{(v, v') = e \in E \mid \text{bts}(v) = 1 \text{ und } x_e = 1\}$$

5 Das Auto Planungs Modifikations Problem

minimize	$w_1 \cdot \sum_{e \in E_M} \ \text{spec}(e)\ _1 \cdot x_e + w_2 \cdot \sum_{e \in E_M} (1 - q_e) \cdot x_e$	
subject to	$\sum_{e \in \delta^+(v)} x_e = 1$	$\forall v \in V_1$
	$\sum_{e \in E_M} \text{spec}(e)_i \cdot x_e = 0$	$\forall i \in \{1, \dots, k\}$
	$\sum_{e \in \delta^-(v^*)} x_e = 1$	
	$x_e \in \{0, 1\}$	$\forall e \in E_M$

Abbildung 5.7: Alternative Formulierung des APMP.

enthalten. Für die Menge der ausgewählten Kanten gilt die Attributerhaltung. Nach Konstruktion existieren in G_M äquivalente Kanten $(v_M, v'_M) \in E_M$ mit $\text{spec}(v) = \text{spec}(v_M)$ und $\text{spec}(v') = \text{spec}(v'_M)$, die ausgewählt werden können. Dadurch ist der Bedarf gedeckt und die Attributerhaltung gewährleistet.

Für alle Knoten $v_M \in V_1$ mit $x_{e_M} = 0 \ \forall e_M \in \delta^+(v_M)$ existiert eine Kante auf die Kopie von v_M , die als Spezifikationsvektor den Nullvektor besitzt. Ihre Auswahl erfüllt die Nebenbedingung einer ausgehenden Kante für alle Knoten der Menge V_1 und verletzt nicht die Attributerhaltung. Die Gegenrichtung verhält sich analog. $\square$

Wie bei der Konstruktion beabsichtigt liefern beide Formulierungen die gleiche Lösung bei derselben Eingabeinstanz. Bei genauerer Betrachtung lässt sich der Zielfunktionswert einer optimalen Lösung von ILP1 in den optimalen Zielfunktionswert von ILP2 umrechnen, falls die gleiche Gewichtung der Zielfunktion verwendet wurde.

Die Größe von ILP2 ist kleiner als die von ILP1. Es werden zwar $|V_1|$ zusätzliche Entscheidungsvariablen benötigt, um die zusätzlichen Kanten in E_M auswählen zu können, jedoch entfallen die Entscheidungsvariablen y_v für jeden Knoten. Desweiteren muss die erste Nebenbedingung ausschließlich für die Menge V_1 , also für die BTS-Autos, anstatt für V definiert werden, wobei innerhalb der Daten gilt, dass die Anzahl der Knoten in V_1 deutlich kleiner ist als die in V .

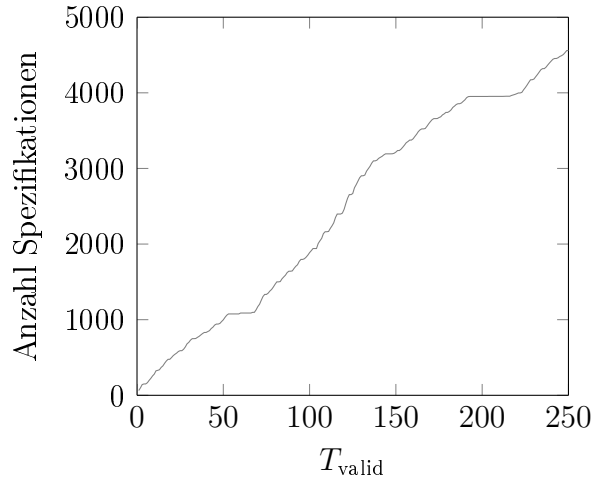


Abbildung 5.8: Wachstum der Anzahl der Knoten im Verhältnis zum Betrachtungszeitraum T_{valid} in der Vergangenheit.

In Abschnitt 5.6 werden Resultate vorgestellt, die unter anderem die Laufzeit der beiden Formulierungen vergleichen.

5.6 Resultate

Zur Interpretation etwaiger Resultate werden zunächst einige Annahmen bzw. Festlegungen gemacht. Der gesamte Lösungsprozess ist durch mehrere Parameter festgelegt, die im Folgenden definiert werden.

Der Zeitraum T_{valid} umfasst 90 Tage zurückgerechnet ab dem Betrachtungszeitpunkt. Dadurch werden durchschnittlich 1300 bis 1700 eindeutig verschiedene Spezifikationen betrachtet. Die Zeitspanne ist gewählt worden, um möglichst viele Spezifikationen zu erhalten und gleichzeitig auslaufende oder sich ändernde Spezifikationen auszuschließen. Abbildung 5.8 zeigt, dass aufgrund von Änderungen in der Produktpalette durchgehend neue Spezifikationen hinzukommen. Es erfolgt keine Sättigung.

Der Suchzeitraum T_{such} befindet sich in entgegengesetzter Richtung und misst 40 Tage. In diesem Zeitraum werden BTS-Auto gesucht und für einen Umbau in Betracht gezogen. Die Orderbank ist durch Ford für die nächsten 4-8 Wochen durch Vorhersagealgorithmen gefüllt, was eine Vergrößerung des Suchzeitraums ermöglicht. Orientiert wurde sich an den maximalen Lieferzeiten für Autoteile. Kann ein Attribut ab einem bestimmten Zeitpunkt geliefert werden, unterliegt es

5 Das Auto Planungs Modifikations Problem

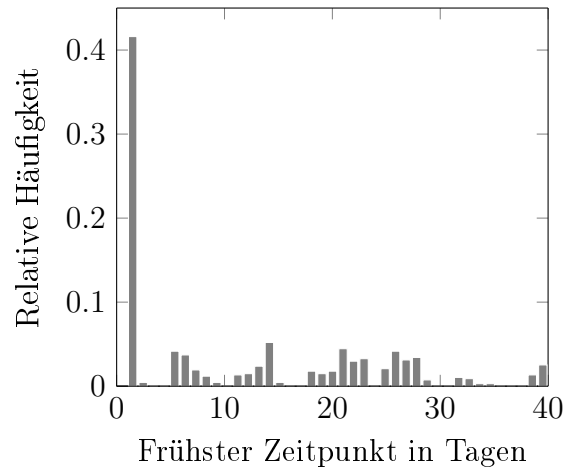


Abbildung 5.9: Relative Häufigkeit, ein gewünschtes Auto frühestens am Tag, abgetragen auf der x-Achse, zu finden.

nicht mehr der Attributerhaltung. Zusätzlich muss als Offset ein Vorlauf von 7 Tagen eingerechnet werden. In diesem Zeitraum ist es Ford nicht möglich etwas an einer bestehenden Bestellung zu ändern. Abbildung 5.9 zeigt die relative Häufigkeit an, das gewünschte Auto frühestens an dem auf der x-Achse abgetragenen Tag umbauen zu können. Der Verlauf der Kurve spricht gegen eine Vergrößerung des Suchzeitraums.

Bei der Definition der Ähnlichkeit zwischen Spezifikationen wird zwischen Attributen, die der Entity angehören, und dem Rest unterschieden. Entityattribute symbolisieren größere Eingriffe in das Auto und werden mit einem Gewicht von 1 bemessen. Die Änderung von Features hingegen mit einem Gewicht von 0.5. Das daraus resultierende Gewicht einer Änderung wird durch die Schranke maxDiff begrenzt und steuert auf diese Weise, wie in Abbildung 5.10 zu sehen, die Anzahl der Kanten, die der Graph enthält.

Das Gewicht der maximalen Änderung zwischen zwei Autos beträgt zum betrachteten Zeitpunkt 16, aufgeteilt auf 5 Änderungen in der Entity und 22 Feature-Änderungen. Bei solch extremen Umbaumaßnahmen kann nicht mehr von Ähnlichkeit gesprochen werden, weshalb der realistisch mögliche Raum für die Wahl der Schranke maxDiff in Abbildung 5.11 aufgezeigt ist. Für die Generierung der Ergebnisse wurde maxDiff auf 2 gesetzt.

Das Vorhandensein eines Lagers vereinfacht dem Modell das Finden einer zulässigen

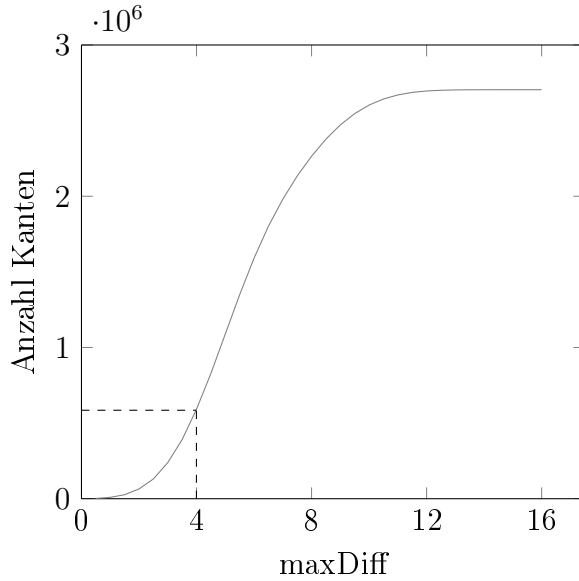


Abbildung 5.10: Verhältnis von maxDiff zur Anzahl der Kanten.

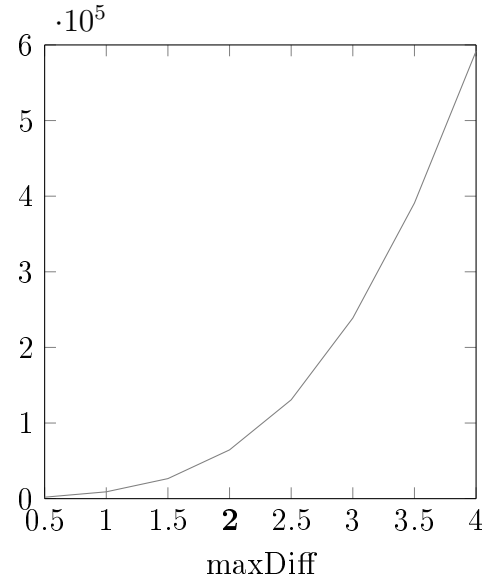


Abbildung 5.11: Relevanter Ausschnitt der Kurve.

Lösung enorm. Trotz dass in der Realität die Attributerhaltung lediglich für ca. 20 Attribute gilt, wird bei den Ergebnissen angenommen, dass kein Attribut gelagert werden kann.

5.6.1 Laufzeit

Die Konstruktion der Übergangsgraphen kann wie in Abbildung 5.6 als Preprocessing stattfinden. Einfluss auf die Konstruktionszeit haben zum einen der Suchzeitraum T_{such} und zum anderen die Anzahl der BTS-Autos am jeweiligen Tag. Insgesamt werden etwa 3min - 5min benötigt, um Übergangsgraphen für jeden Tag und die darin gespeicherten Informationen zu erstellen.

Bei der Betrachtung der Laufzeit, also der benötigten Zeit für eine Anfrage, werden ausschließlich Instanzen betrachtet, in denen ein Umbau möglich ist. Verglichen werden in Abbildung 5.12 die Optimierungsvariante des ILP1 mit und ohne Ähnlichkeitsbeschränkung (ILP1*) und die Optimierungsvariante des ILP2.

Die durchschnittliche Anfragezeit für das ILP1 beträgt 1.57s. Minimal schneller, wahrscheinlich aus der geringeren Anzahl an Nebenbedingungen resultierend, beantwortet das ILP2 in durchschnittlich 1.34s eine Anfrage. Die Menge der Ausreißer in Richtung der hohen Laufzeit besteht aus Fällen, in denen der gesuchte Knoten v^* in jedem Graphen G_t liegt, jedoch erst zu einem späteren Zeitpunkt ein valider

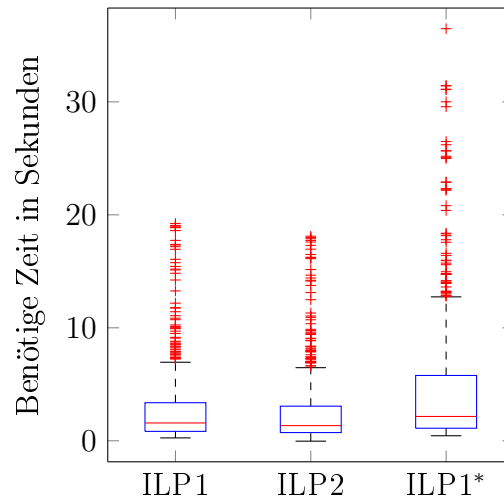


Abbildung 5.12: Laufzeiten der Optimierungsvarianten von ILP1, ILP2 und ILP1* ohne Ähnlichkeitsbeschränkung am Zielknoten.

Umbau existiert. Dass die durchschnittliche Laufzeit trotzdem sehr gering ist, ist in Abbildung 5.9 erklärt. Die meisten validen Umbauten können am ersten Tag gefunden werden, wonach der Algorithmus abbricht. Soll der vollständige Suchzeitraum durchsucht werden, um ggf. einen späteren, validen Umbau mit geringerem Zielfunktionswert zu wählen, muss mit deutlich höheren Laufzeiten gerechnet werden.

Entfernt man die Ähnlichkeitsbeschränkung für den Zielknoten, müssen innerhalb der Laufzeit Entscheidungsvariablen für Kanten und Knoten hinzugefügt werden. Dies erklärt die erhöhte Laufzeit. Zusätzliche Laufzeitverbesserungen können durch das Parallelisieren der Anfragen für die einzelnen Tage erreicht werden.

5.6.2 Einzelergebnisse

Im Folgenden wird ein Beispiel für einen validen Umbau, basierend auf Ford Fiesta Daten, gezeigt. Im ausgewählten Fall, zu sehen in Abbildung 5.13, benötigt das Wunschauto einen anderen Motor und ein zusätzliches Feature. Die unten grau dargestellten Autoteile sind in den geplanten Autos enthalten und werden ausgebaut. Alle Einträge, die einen eingehenden Pfeil besitzen, waren ursprünglich nicht in den geplanten Autos enthalten, sind es aber nach dem Umbau.

Im nächsten Beispiel (Abbildung 5.14) wird die Schwäche der unvollständigen Menge an validen Spezifikationen deutlich. Es findet ein Ringtausch der Motoren statt. Das Flex Pack 5 wird getauscht, weil die Menge der validen Spezifikationen nicht BTS-Auto 1 mit diesem Feature enthält.

5 Das Auto Planungs Modifikations Problem

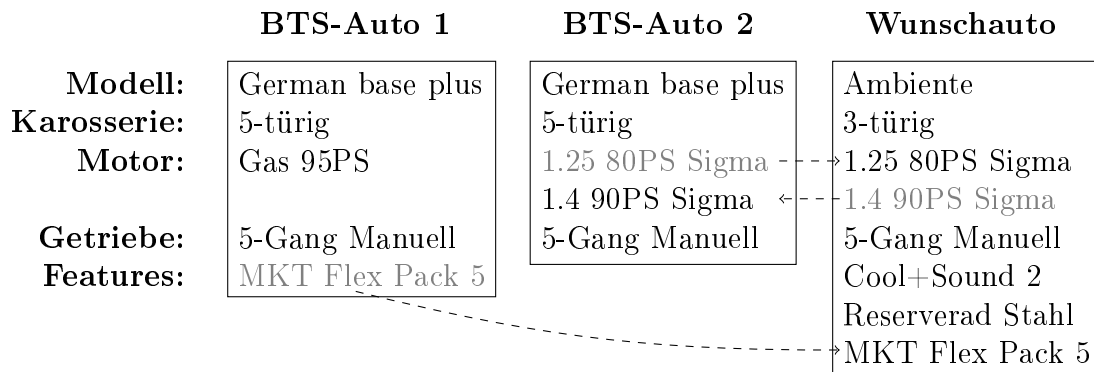


Abbildung 5.13: Zu sehen ist ein valider Umbau, der drei Autos beeinflusst.

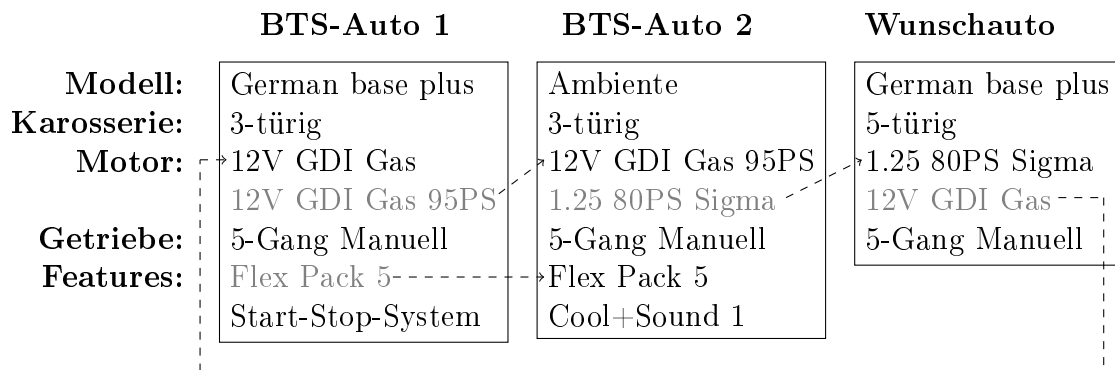


Abbildung 5.14: Zu sehen ist ein Ringtausch der Motoren und zusätzlich der Tausch eines optionalen Features.

5.6.3 Gesamtergebnis

Um eine begründete Aussage über die Qualität des Lösungsansatzes treffen zu können, wurden zu drei unterschiedlichen Zeitpunkten die vollständige Menge der validen Spezifikationen nacheinander angefragt.

Innerhalb des vorgegebenen Suchzeitraums T_{such} wird nach der angeforderten Spezifikation gesucht und folgende vier Klassen unterschieden.

1. Die gesuchte Spezifikation ist weder zu bauen geplant, noch kann es durch das Modell und einen validen Umbau umgebaut werden.
2. Die gesuchte Spezifikation ist zu einem Zeitpunkt in der Datenbank zu finden. Allerdings ist der Umbau durch das Modell zu einem früheren Zeitpunkt

5 Das Auto Planungs Modifikations Problem

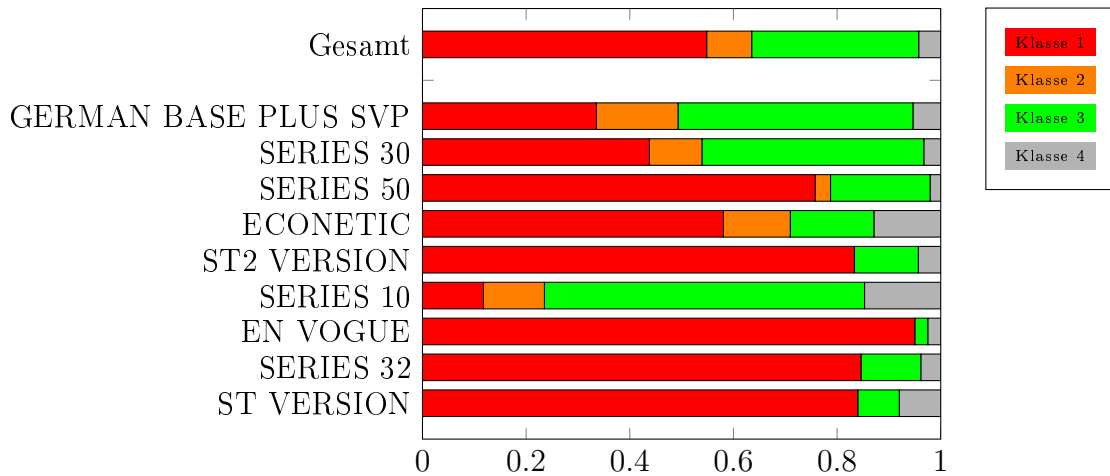


Abbildung 5.15: Klassenverteilung bei der Anfrage von allen validen Spezifikationen zu drei Zeitpunkten.

möglich. In diesem Fall muss entschieden werden, wie die frühere Lieferung des Autos an den Kunden in Relation zu einem Umbau steht. Hierbei handelt es sich um eine eventuelle Verbesserung zum aktuellen Verfahren.

3. Die gesuchte Spezifikation ist nicht geplant, kann aber durch einen validen Umbau erreicht werden. Diese Klasse ist als klare Verbesserung zum aktuellen Verfahren anzusehen.
4. Es existiert ein geplantes BTS-Auto in der Datenbank, dessen Spezifikation genau der gesuchten entspricht. Gleichzeitig existiert kein valider Umbau zu einem früheren Zeitpunkt.

In Abbildung 5.15 wird ein Überblick über die Klassenverteilung aller Anfragen gegeben. Unterschieden wird desweiteren nach den neun häufigsten Modellvarianten, die absteigend sortiert sind.

Ein prozentual großer Anteil der Anfragen liegt in Klasse 1 und gilt somit als ungelöst. Bei genauerer Betrachtung der Ergebnisse fällt auf, dass für einen Großteil der ungelösten Anfragen kein ILP aufgestellt wurde. Wie in Abschnitt 5.3.2 bereits angemerkt, ist das Fehlen des Bedarfsknotens v^* in dem Übergangsgraph G_t der Grund dafür. Wird die Ähnlichkeitsbeschränkung für den Zielknoten aufgehoben, können weitere 5% der Anfragen durch einen Umbau erfüllt werden.

Weitere Ursachenforschungen liefern drei Hauptgründe, die einen Großteil der ungelösten Anfragen erklären.

Als erstes sei die unvollständige Menge der validen Spezifikationen angeführt.

5 *Das Auto Planungs Modifikations Problem*

Bedingt durch die Unvollständigkeit müssen häufig zusätzliche Autoteile ein- bzw. ausgebaut werden. Diese Differenz führt aufgrund der Attributerhaltung zu weiteren Umbauten, was das Finden einer zulässigen Lösung erschwert. Aus theoretischer Sicht wäre eine vollständige Darstellung der validen Spezifikationen wünschenswert. Aus praktischer Sicht darf nicht vergessen werden, dass die gewählte Art und Weise der Konstruktion valider Spezifikation implizit garantiert, dass nur verkaufbare oder bereits vom Kunden zusammengestellte Autos gebaut werden.

Selten gebaute Modellvarianten oder im Allgemeinen selten verbaute Autoteile seien als zweiter Grund aufgeführt. Im Extremfall wird ein Autoteil ein einziges Mal in einem Auto verbaut. Die daraus resultierende Spezifikation geht genau wie gängigere Spezifikation mit einem Gewicht von 1 in die Übersichtsdarstellung ein. Ein Umbau ist in den seltensten Fällen möglich, da es keine Auswahl für das Modell gibt. So werden beispielsweise von der Modellvariante "SERIES 32" durchschnittlich zwei BTS-Autos pro Tag gebaut. Die Wahrscheinlichkeit einen Umbau mit einer Ähnlichkeitsbeschränkung mit einem Gewicht von 2 zu finden ist gering.

Aus der Vermarktungsstrategie von Ford resultiert, dass für teurere Modelle mehr Auswahlmöglichkeiten zum Beispiel an Motoren und vor allem an zusätzlichen Features angeboten werden. Daher ist die Variantenvielfalt bei hochwertigeren Modellen deutlich höher, was das Finden einer zulässigen Lösung erschwert. Mit der Zusatzinformation, dass es sich bei SERIES 50, EN VOGUE und den ST-Versionen um teurere Modellvarianten handelt, ist dieser Trend ebenfalls in Abbildung 5.15 zu sehen.

Generell sollte nicht erwartet werden, dass alle Anfragen unter realistischen Bedingungen erfüllt werden können. Es gilt jedoch; je extravaganter ein Wunschauto im Verhältnis zu ursprünglich produzierten Autos ist, desto schwerer wird ein Umbau und folglich desto länger wird seine Lieferzeit. Dem Kunden schwer zu vermitteln sind hohe Lieferzeiten bei gängigen Varianten, deren Umbau zum Großteil durch das Modell ermöglicht wird.

6 Komplexitätsanalyse

Das Auto Planungs Modifikations Problem unterliegt aufgrund der Anwendung der Lieferzeitplanung für ein gewünschtes Auto laufzeittechnischen Forderungen. Dieses Kapitel soll Antwort darauf geben, ob der Entwurf eines effizienten Algorithmus möglich ist. Falls nicht, würde dies die Verwendung von ILPs rechtfertigen.

6.1 Komplexitätsklassen P und NP

Ein Entscheidungsproblem, wie das Auto Planungs Modifikations Problem, lässt sich in Komplexitätsklassen einordnen, die Auskunft darüber geben, in welcher Laufzeit ein solches Problem lösbar ist. Falls für ein Entscheidungsproblem ein Algorithmus existiert, der das Problem in polynomieller Zeit löst, liegt dieses in der Komplexitätsklasse P (siehe Definition 18). Algorithmen mit polynomiell begrenzter Laufzeit werden auch als effiziente Algorithmen bezeichnet.

Definition 18. Die Komplexitätsklasse P enthält alle Entscheidungsprobleme, für die ein Algorithmus existiert, der zu allen möglichen Eingaben eine korrekte Antwort in polynomialer Zeit liefert.

Für das Sortieren einer gegebenen Menge an Zahlen oder die Entscheidung, ob eine gegebene Zahl eine Primzahl ist, existieren beispielsweise effiziente Algorithmen und diese Entscheidungsprobleme liegen somit in P.

Die Komplexitätsklasse NP (nichtdeterministische Polynomialzeit), einführt in Definition 19, kann durch nichtdeterministische Turingmaschinen (NTM), deren Funktionsweise von Alan Turing entwickelt wurde und beispielsweise in [L81] beschrieben sind, charakterisiert werden. Einer NTM ist es möglich den Übergang von einem zum anderen ihrer Zustände zu raten und dadurch Probleme, die auf deterministischen Turingmaschinen (DTM) bisher nur in exponentieller Laufzeit gelöst werden konnten, in Polynomialzeit zu lösen.

Definition 19. Die Komplexitätsklasse NP enthält alle Entscheidungsprobleme, die sich in Polynomialzeit durch eine nichtdeterministische Turingmaschine entscheiden lassen.

Dass ein Entscheidungsproblem in NP liegt, kann alternativ durch die Angabe eines Polynomialzeitalgorithmus gezeigt werden, der eine gegebene Lösung auf Korrektheit überprüft. Weiterhin muss die dazugehörige Eingabe eine polynomiell beschränkte Länge besitzen.

Definition 20. Seien L und L' zwei Entscheidungsprobleme mit $L, L' \subseteq \Sigma^*$. Das Problem L ist auf das Problem L' polynomiell reduzierbar ($L \leq_p L'$), wenn es eine in polynomieller Zeit berechenbare Funktion $f : \Sigma^* \rightarrow \Sigma^*$ gibt, so dass für alle Eingaben $w \in \Sigma^*$ gilt: $w \in L \Leftrightarrow f(w) \in L'$.

Mittels der in Definition 20 eingeführten Polynomialzeitreduktion lässt sich die NP-Schwere (Definition 21) definieren, die die Eigenschaft von Entscheidungsproblemen beschreibt, dass ein NP-schweres Problem mindestens so schwer ist, wie alle Probleme in NP.

Definition 21. Ein Entscheidungsproblem $L' \subseteq \Sigma^*$ heißt NP-Schwer, wenn gilt:

$$\forall L \in \text{NP} : L \leq_p L'.$$

Liegt ein Entscheidungsproblem in NP und kann die NP-Schwere nachgewiesen werden, so wird das Problem als NP-vollständig bezeichnet. Der Nachweis der NP-Schwere kann unter anderem und wie in den nächsten Abschnitten durch eine Polynomialzeitreduktion von einem Problem stattfinden, dessen NP-Vollständigkeit bereits bewiesen wurde.

6.2 Komplexität des Auto Planungs Modifikations Problems

Ziel dieses Kapitels ist es die vermutete NP-Vollständigkeit des Auto Planungs Modifikations Problems zu zeigen. Dafür wird eine Reduktion von dem Erfüllbarkeitsproblem der Aussagenlogik, mit Zwischenschritt über ein spezielles Binary Programming Problem, auf das Auto Planungs Modifikations Problem geführt.

6.2.1 Das Hyper Binary Integer Programming Problem

In dem Erfüllbarkeitsproblem der Aussagenlogik, kurz SAT, wird für eine gegebene aussagenlogische Formel Φ in konjunktiver Normalform (KNF) entschieden, ob es eine Belegung der Variablen gibt, sodass Φ erfüllt ist.

Erfüllbarkeitsproblem der Aussagenlogik (SAT)

Eingabe: Aussagenlogische Formel Φ in KNF.

Frage: Existiert eine Belegung, sodass Φ erfüllt ist.

Für SAT wurde 1971 durch den Satz von Cook [C71] gezeigt, dass das Problem NP-vollständig ist.

Das 0-1 Integer Programming Problem, kurz 01-INT, gehört zu Karp's 21 Problemen [K72], deren NP-Vollständigkeit er im Jahre 1972 mittels Reduktion von dem Erfüllbarkeitsproblem der Aussagenlogik zeigte.

Für das 0-1 Integer Programming Problem wird ein Binärvektor x für eine ganzzahlige Matrix C und einen ganzzahligen Vektor b gesucht, sodass das Gleichungssystem $Cx = b$ erfüllt ist.

0-1 Integer Programming (01-INT)

Eingabe: $C \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$

Frage: Existiert ein $x \in \{0, 1\}^n$, sodass $Cx = b$.

Wie bereits erwähnt ist die NP-Vollständigkeit bereits von Karp durch eine polynomielle Reduktion von SAT auf 01-INT gezeigt worden. Da die Beweisidee im späteren Verlauf dieser Arbeit aufgegriffen wird, folgt zunächst eine Beweisskizze.

Satz 6.1. *Das 01-INT Problem ist NP-schwer [K72].*

6 Komplexitätsanalyse

Beweisskizze. Sei eine SAT Instanz $\Phi = C_1 \wedge \dots \wedge C_m$ über der Variablen $x_1, \dots, x_n$ gegeben.

Jede Spalte der Matrix C symbolisiert eine Variable. Jede Zeile stellt den Zusammenhang von Variablen innerhalb einer Klausel dar. Innerhalb der Matrix $C \in \mathbb{Z}^{m \times n}$ ist jeder Eintrag c_{ij} mit $i \in \{1, \dots, m\}$ und $j \in \{1, \dots, n\}$ definiert als:

$$c_{ij} = \begin{cases} 1 & , \text{ falls } x_j \in C_i \\ -1 & , \text{ falls } \overline{x_j} \in C_i \\ 0 & , \text{ ansonsten.} \end{cases}$$

Die rechte Seite des Gleichungssystems wird in Abhängigkeit der negierten Variablen pro Klausel gebildet. Es gilt:

$$b_i = 1 - \text{Anzahl negativer Variablen in } C_i \quad \forall i \in \{1, \dots, m\}.$$

Durch das Einführen von zwei Schlupfvariablen für jede Zeile kann die Gleichheit gefordert werden. □

Um im späteren Verlauf eine sauberere Reduktion auf das APMP führen zu können, wird das *Hyper Binary Integer Programming Problem*¹, kurz H-INT, eingeführt und dessen NP-Vollständigkeit bewiesen. Beim H-INT Problem handelt es sich um einen Spezialfall des 01-INT Problems, in welchem ebenfalls ein Binärvektor x gesucht wird, allerdings zusätzliche Forderungen an C und b gestellt werden.

Hyper Binary Integer Programming (H-INT)

Eingabe: $D \in \{-1, 0, 1\}^{m \times n}$, $b \in \{-1, 0, 1\}$

Frage: Existiert ein $x \in \{0, 1\}^n$, sodass $Dx = b$.

Die Beschränkung des Vektors b ist im späteren Verlauf wichtig. Diese Beschränkung wird bei der Reduktion von SAT auf H-INT implizit erreicht, indem die Anzahl der Literale pro Klausel begrenzt werden und weder alle Literale wahr noch falsch sein dürfen. Die erste Einschränkung wird durch die Verwendung von 3-SAT, dem wohl bekanntesten Spezialfall von SAT, gewährleistet. 3-SAT wird an dieser Stelle formal eingeführt.

¹Der Name Hyper ist aufgrund der Ähnlichkeit zu der Kodierung von Hyperkanten in einem Hypergraph gewählt worden.

3-SAT

Eingabe: Aussagenlogische Formel Φ in KNF, in der jede Klausel maximal 3 Literale enthält.
Frage: Existiert eine Belegung, sodass Φ erfüllt ist.

Die NP-Vollständigkeit für den Spezialfall 3-SAT wurde ebenfalls von Karp 1972 gezeigt.

Satz 6.2. *3-SAT ist NP-vollständig [K72].*

Dass 3-SAT in NP liegt, ist durch die Tatsache gegeben, dass es sich um einen Spezialfall von SAT handelt. Die Idee der Reduktion, und somit der Beweis der NP-Schwere, wird in Beispiel 11 verdeutlicht.

Beispiel 11. Sei eine Klausel C einer SAT Instanz $(y_1 \wedge y_2 \wedge \dots \wedge y_l)$ mit $l > 3$ gegeben. Durch die Einführung einer neuen Variable z kann die Größe der Klausel schrittweise um 1 verringert werden.

$$C = (y_1 \wedge y_2 \wedge \dots \wedge y_l) \rightsquigarrow (y_1 \wedge y_2 \wedge z) \vee (y_3 \wedge \dots \wedge y_l \wedge \bar{z}) = C'$$

Klausel C kann genau dann erfüllt werden, wenn Formel C' erfüllt werden kann.

Die zweite Forderung, dass nicht alle Literale einer Klausel wahr bzw. falsch sein dürfen, ist in der eingeschränkten Version NAE-3-SAT (engl. not all equal) gegeben, für die ebenfalls die NP-Vollständigkeit bekannt ist.

Satz 6.3. *Das NAE-3-SAT Problem ist NP-vollständig [S78].*

Final wird die NP-Vollständigkeit des H-INT Problems gezeigt. Dabei gilt ebenfalls, dass das H-INT Problem in NP liegt, da es sich um ein Spezialfall des 01-INT Problems handelt. Durch eine polynomielle Reduktion von NAE-3-SAT auf H-INT wird die NP-Schwere gezeigt.

Satz 6.4. *Das H-INT Problem ist NP-schwer.*

Beweis. Sei eine NAE-3-SAT Instanz $\Phi = C_1 \wedge \dots \wedge C_m$ mit $|C_i| \leq 3$ für alle $i \in \{1, \dots, m\}$ über den Variablen $y_1, \dots, y_n$ gegeben.

Die konstruierte Matrix D für das H-INT Problem besitzt m Zeilen und $2m + n$

6 Komplexitätsanalyse

Spalten. Für einen Eintrag $d_{ij} \in \{-1, 0, 1\}$ mit $i \in \{1, \dots, m\}$ und $j \in \{1, \dots, 2m+n\}$ gilt:

$$d_{ij} = \begin{cases} 1 & , \text{ falls } y_j \in C_i \text{ für } j \leq n \\ -1 & , \text{ falls } \overline{y_j} \in C_i \text{ für } j \leq n \\ -1 & , \text{ falls } 2i = j + n \vee 2i + 1 = j + n \text{ für } j > n \\ 0 & , \text{ ansonsten} \end{cases}$$

Die negativen Einträge der Matrix D für $j > n$ entsprechen Schlupfvariablen, die frei mit 0 oder 1 belegt werden können, um die Gleichheit zu erfüllen. Die rechte Seite des Gleichungssystems b ist analog zur Reduktion von SAT nach 01-INT definiert:

$$b_i = 1 - \text{Anzahl negativer Variablen in } C_i \quad \forall i \in \{1, \dots, m\}$$

Die Anzahl negativer Variablen pro Klausel kann aufgrund der Problemstellung entweder 1 oder 2 betragen. Daher sind die Einträge des Vektors b auf 0 und -1 beschränkt. Die Konstruktion kann in polynomieller Zeit durchgeführt werden. Im Anschluss wird die Korrektheit der Konstruktion bewiesen.

($\Rightarrow$): Sei eine erfüllbare NAE-3-SAT Formel gegeben. Das heißt, es existiert eine Belegung der Variablen, sodass für jede Klausel mindestens ein Literal wahr ist. Der Eintrag x_j mit $j \in \{1, \dots, n\}$ des gesuchten Binärvektor x wird genau dann auf 1 gesetzt, wenn für Variable y_j innerhalb der erfüllenden Belegung $y_j = 1$ gilt. Für eine Klausel $C_i = (y_{j_1} \vee y_{j_2} \vee y_{j_3})$ mit $i \in \{1, \dots, m\}$ und $j_1, j_2, j_3 \in \{1, \dots, n\}$ existiert eine Nebenbedingung der folgenden Gestalt in der Konstruktion der H-INT Instanz:

$$x_{j_1} + x_{j_2} + x_{j_3} - x_{n+2i-1} - x_{n+2i} = 1.$$

Nach Voraussetzung ist mindestens ein Literal wahr und somit die größer-gleich-Beziehung erfüllt. Die Gleichheit wird durch die Schlupfvariablen x_{n+2i-1} und x_{n+2i} gewährleistet, die frei gewählt werden können. Für negierte Variablen $\overline{y_j}$ ist sowohl der Koeffizient der x_j Variable negativ, als auch die rechte Seite um den Wert 1 verringert. Dadurch ist in allen Fällen die Erfüllung der Gleichung möglich, falls, wie vorausgesetzt, die zugehörige Klausel erfüllt ist. Insgesamt erfüllt x das Gleichungssystem $Dx = b$.

($\Leftarrow$): Sei für die Matrix D und den Vektor b ein Binärvektor x gegeben, sodass das Gleichungssystem $Dx = b$ erfüllt ist.

Daraus folgt, dass für jede Zeile mindestens eine x -Variable mit positivem Vorzeichen

gleich 1 oder eine x -Variable mit negativem Vorzeichen gleich 0 sein muss. Andernfalls kann die Nebenbedingung nicht erfüllt werden. Werden die zugehörigen y_j Variablen gleich 1 bzw. 0 gesetzt, falls $x_j = 1$ bzw. $x_j = 0$ gilt, enthält jede Klausel ein wahres Literal. Die NAE-3-SAT Formel ist somit erfüllt. $\square$

Aus der NP-Schwere und der Tatsache, dass das H-INT Problem in NP liegt, folgt die NP-Vollständigkeit.

Satz 6.5. *Das H-INT Problem ist NP-vollständig.*

6.2.2 Reduktion auf APMP

Zum Beweis der NP-Vollständigkeit des APMPs muss sowohl gezeigt werden, dass das APMP in NP liegt, als auch dass es eine polynomielle Reduktion von einem NP-vollständigen Problem auf das APMP gibt. Zur Erinnerung ist die formale Definition aus Kapitel 5 hier nochmals aufgeführt.

AUTO PLANUNGS MODIFIKATIONS PROBLEM

Eingabe: Übergangsgraph $G = (V, E, \text{spec})$, Menge $V_{\text{BTS}} \subseteq V$ und Knoten $v^* \in V$.

Frage: Existiert ein valider Umbau auf G mit V_{BTS} , sodass die Attributerhaltung gewährleistet ist und v^* gebaut wird.

Satz 6.6. *Das APMP liegt in NP.*

Beweis. Eine Lösung des APMPs kann durch einen Bitvektor der Länge entsprechend der Anzahl der Kanten dargestellt werden. Einem Algorithmus ist es in Polynomialzeit möglich die Attributerhaltung der ausgewählten Kanten und das Erreichen des Bedarfsknotens v^* zu überprüfen. $\square$

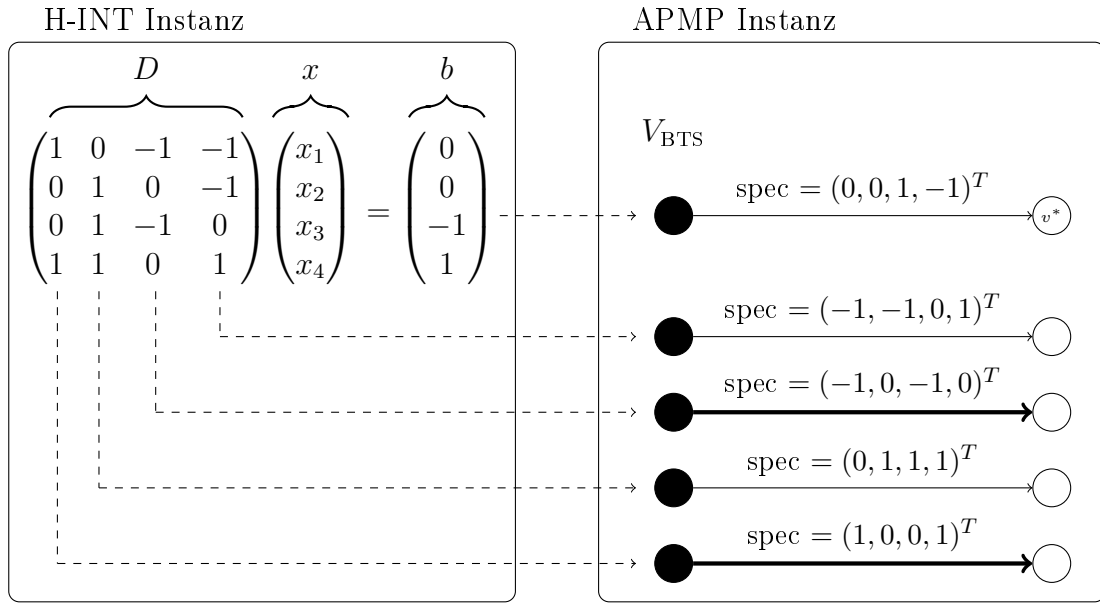
Im nächsten Schritt wird die NP-Schwere des APMPs durch eine Reduktion von dem in Abschnitt 6.2.1 eingeführten H-INT Problem bewiesen.

Satz 6.7. *Das APMP ist NP-schwer.*

6 Komplexitätsanalyse

Zuerst wird die Konstruktionsidee einer APMP Instanz durch eine H-INT Instanz in Beispiel 12 verdeutlicht.

Beispiel 12. Sei eine Matrix $D \in \{-1, 0, 1\}^{4 \times 4}$ und ein Vektor $b \in \{-1, 0, 1\}^4$ gegeben. Eine Spalte der Matrix D wird als Spezifikationsvektor auf einer Kante abgetragen. Eine Auswahl der Spalte j entspricht dem Umbau des Startknotens in den Zielknoten der entsprechenden Kante. In der H-INT Instanz garantiert die rechte Seite b , dass der Nullvektor im Allgemeinen keine zulässige Lösung für $Dx = b$ ist. Analog dient der Knoten v^* , der in einer Ja-Instanz gebaut werden muss, dazu, dass Umbauten stattfinden müssen. Auf der eingehenden Kante in v^* ist aufgrund dessen als Spezifikationsvektor $-b$ abgetragen.



Durch die Auswahl der dick gedruckten Kanten in der APMP Instanz kann durch Rückübersetzung auf die Variablen $x_1 = x_3 = 1$ die H-INT Instanz gelöst werden.

Nach diesem eingehenden Beispiel wird die Konstruktion im folgenden Beweis formal beschrieben und ihre Korrektheit bewiesen.

Beweis. Gegeben sei eine Instanz des H-INT Problems mit einer Matrix $D \in \{-1, 0, 1\}^{m \times n}$ und einem Vektor $b \in \{-1, 0, 1\}^m$.

Jede Spalte der Matrix D wird im APMP durch einen möglichen Umbau dargestellt. Der Spezifikationsvektor der Kante $e_j \in E$, die für den Umbau ausgewählt wird, entspricht den Einträgen der jeweiligen Spalte der Matrix D .

Für die j -Spalte D_j mit $D_j = (d_{1j}, \dots, d_{mj})^T$ und $j \in \{1, \dots, n\}$ werden zwei

6 Komplexitätsanalyse

Knoten $v_{j1} \in V_{\text{BTS}}$ und $v_{j2} \in V$ in den Übergangsgraphen eingefügt. Die Spezifikation der beiden Knoten wird so gewählt, dass der sich zu berechnende Spezifikationsvektor der Kante $e_j = (v_{j1}, v_{j2})$ den Einträgen in D_j entspricht. Für die Spezifikation von v_{j1} gilt:

$$\text{spec}(v_{j1})_i = \begin{cases} 0 & , \text{ falls } d_{ij} = 1 \\ 1 & , \text{ falls } d_{ij} = -1 \text{ oder } d_{ij} = 0 \end{cases} \quad \forall i \in \{1, \dots, m\}$$

und weiterhin für die Spezifikation von v_{j2} :

$$\text{spec}(v_{j2})_i = \begin{cases} 0 & , \text{ falls } d_{ij} = -1 \\ 1 & , \text{ falls } d_{ij} = 1 \text{ oder } d_{ij} = 0. \end{cases} \quad \forall i \in \{1, \dots, m\}$$

Vektor $b \in \{-1, 0, 1\}$ wird auf die gleiche Weise wie eine Spalte der Matrix in eine Kante (v_{b1}, v_{b2}) übersetzt, sodass $\text{spec}((v_{b1}, v_{b2})) = -b$ gilt. Zusätzlich handelt es sich bei dem Zielknoten v_{b2} um den Bedarfsknoten v^* , welcher in einer Ja-Instanz gebaut werden muss.

($\Rightarrow$): Gegeben ist eine Matrix D , ein Vektor b und x , sodass gilt: $Dx = b$. In der konstruierten APMP Instanz muss die eindeutige Kante zum Erreichen von Knoten v^* ausgewählt werden. Der auszugleichende Spezifikationsvektor entspricht $-b$. Da für jede Zeile $D_{i\cdot}$ und b_i die Gleichheit $D_{i\cdot}x = b_i$ gilt, kann durch Auswahl aller Kanten e_j mit $x_j = 1$ die Attributerhaltung gewährleistet werden:

$$\sum_{\substack{j \in \{1, \dots, n\}: \\ x_j = 1}} \text{spec}(e_j)_i - b_i = 0 \quad \forall i \in \{1, \dots, m\}.$$

($\Leftarrow$): Gibt es in der Gegenrichtung einen validen Umbau, sodass der Bedarf erfüllt und die Attributerhaltung eingehalten wird, wird jedes Attribut genau so oft ein- wie ausgebaut. Wird $x_j = 1$ gesetzt, falls die zugehörige Kante $e_j \in E$ ausgewählt wurde und als Zielknoten nicht den Knoten v^* enthält, werden aufgrund der geltenden Attributerhaltung Spalten in D ausgewählt, sodass das Gleichungssystem erfüllt ist. $\square$

Da das APMP in NP liegt (Satz 6.6) und nach Satz 6.7 NP-schwer ist folgt zusammengefasst in Satz 6.8 die NP-Vollständigkeit.

Satz 6.8. *Das APMP ist NP-vollständig.*

6 Komplexitätsanalyse

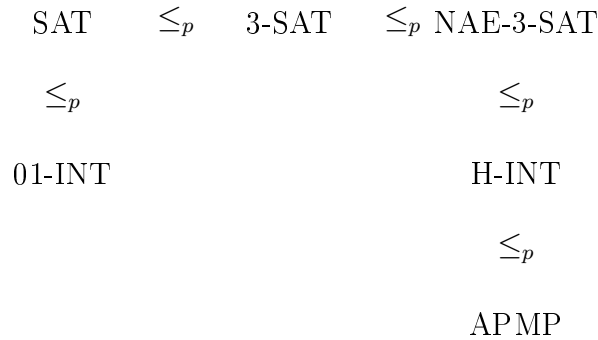


Abbildung 6.1: Übersicht über die Reduktionen, die in Abschnitt 6.2 durchgeführt worden sind.

Unter der Annahme, dass P ungleich NP gilt, existiert kein effizienter Algorithmus, der das APMP in Polynomialzeit lösen kann. Die Optimierungsvariante ist deshalb ebenfalls nicht in Polynomialzeit lösbar. Abschließend gibt Abbildung 6.1 einen Überblick über die durchgeführten Reduktionen.

Abschnitt 6.3 gibt Antworten auf die Frage, ob es durch weitere Einschränkungen möglich ist, die Komplexität des Problems zu reduzieren.

6.3 Einschränkungen an das APMP

Mit großer Wahrscheinlichkeit ist die Möglichkeit des Ein- und Ausbaus mehrerer Attribute pro Kante der Grund für die hohe Komplexität des Problems. Beschränkt man die Attribute beispielsweise auf die Motoren, im Folgenden als Attributfamilie bezeichnet, kann sogar die Optimierungsvariante des Problems in Polynomialzeit gelöst werden.

Für alle Attribute der Entity gilt, dass genau ein Attribut ausgewählt sein muss, um ein valides Auto zu erhalten. Das heißt im Umkehrschluss, dass der Ausbau eines dieser Autoteile den Einbau genau eines anderen Autoteiles der Attributfamilie zur Folge hat. Ein Spezifikationsvektor, begrenzt auf eine Attributfamilie, besitzt demnach genau einen Eintrag gleich 1 und einen Eintrag gleich -1 .

Die Idee, verdeutlicht in Beispiel 13, ist das APMP auf das kürzeste Wege Problem zu reduzieren, für das ein Polynomialzeitalgorithmus bekannt ist. Dabei wird ein Graph G_D konstruiert, der einen Knoten für jedes Attribut der Attributfamilie

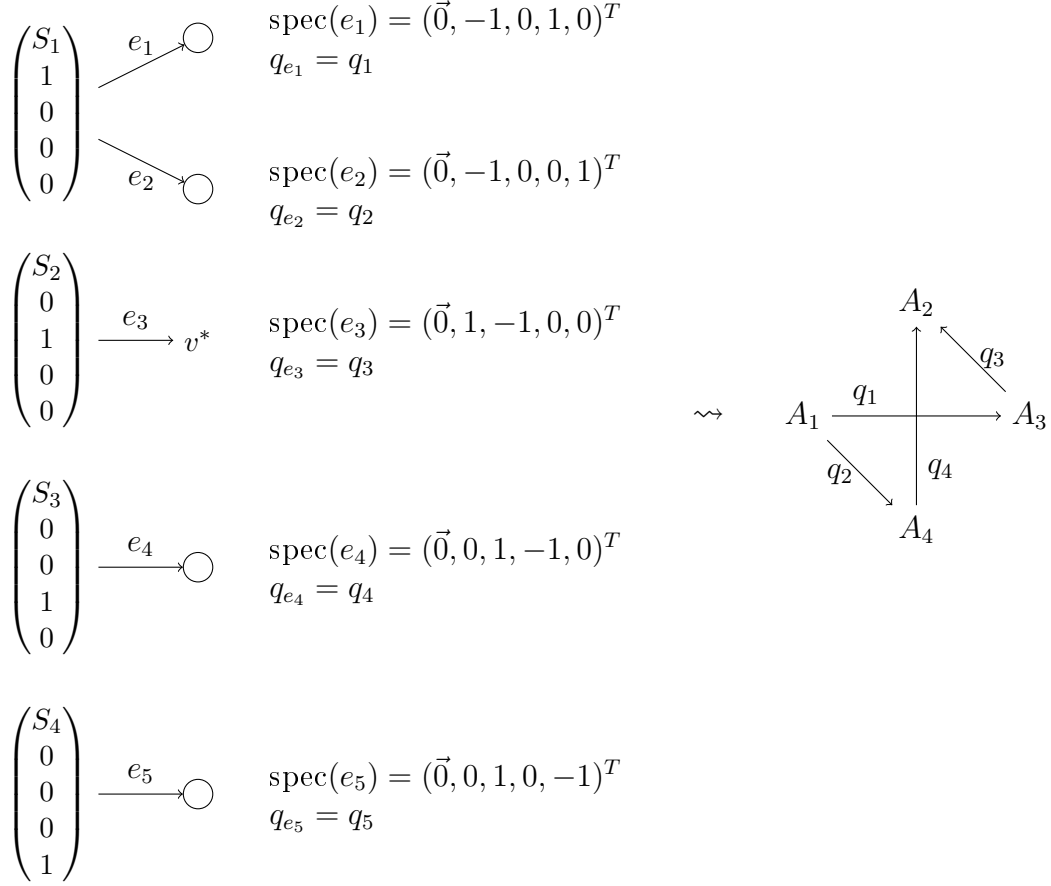
6 Komplexitätsanalyse

enthält. Aufgrund der speziellen Gestalt der Spezifikationsvektoren auf den Kanten des Übergangsgraphen, kann für jeden dieser Spezifikationsvektoren eine Kante in G_D eingefügt werden. Ausgehend von dem Knoten, dessen Attribut ausgebaut wurde, und eingehend in den Knoten, dessen Attribut eingebaut wurde. Als Distanz wird das Qualitätsmaß verwendet. Es ist möglich, dass mehrere Spezifikationsvektoren des Übergangsgraphen die gleiche Kante in G_D kodieren. In diesem Fall wird als Distanz der Kante in G_D das maximale Qualitätsmaß der Kanten im Übergangsgraphen gewählt.

Für alle Kanten (v, v^*) mit $v \in V_{\text{BTS}}$, $\text{spec}((v, v^*))_i = 1$ und $\text{spec}((v, v^*))_j = -1$ im Übergangsgraphen, also die Kanten, die den Umbau von einem geplanten Auto in das gewünschte darstellen, werden auf dem konstruierten Graphen G_D die kürzesten Wege zwischen Knoten i und Knoten j gesucht.

Beispiel 13. Durch das Ermitteln des kürzesten Weges wird die Entscheidung getroffen, welcher der beiden möglichen Umbauten der optimale bezüglich des Qualitätsmaßes ist.

6 Komplexitätsanalyse



Das Problem des kürzesten Weges zwischen einem Start- und Zielknoten in einem Graphen kann durch effiziente Algorithmen, beispielsweise den Dijkstra-Algorithmus [D59] in Polynomialzeit gelöst werden. Ein kürzester Weg muss maximal $|V_{\text{BTS}}|$ mal gefunden werden, wodurch das auf eine Attributfamilie eingeschränkte APMP in Polynomialzeit gelöst werden kann.

Satz 6.9. *Das auf eine Attributfamilie eingeschränkte Auto Planungs Modifikations Problem liegt in der Komplexitätsklasse P .*

7 Zusammenfassung & Ausblick

Das letzte Kapitel gibt eine Zusammenfassung über die geleistete Arbeit und beschreibt den erreichten Umsetzungsgrad hinsichtlich der anfangs erläuterten Problemstellung und der Ziele. Die vorangegangenen Kapitel lassen sich thematisch in drei Teile aufteilen.

Im ersten Teil wurden die von Ford bereitgestellten Verkaufsdaten mittels Data Mining Methoden analysiert. Nach der Übersichtsgewinnung durch statistische Methoden konnte die Größenordnung des später entwickelten Modells abgeschätzt werden. Erste anwendbare Ergebnisse konnten durch die Assoziationsanalyse erreicht werden. Auf diese Weise können in einem unbekannten Datensatz Korrelationen gefunden werden, die Auskunft über das Kaufverhalten der Kunden geben. Das darauf folgende Ziel war die Verkaufbarkeit der Autos abzuschätzen. In diesem Zusammenhang wurde die Verkaufsgeschwindigkeit von BTS-Autos analysiert und es konnten Unterschiede zwischen den auf Entity-Level gruppierten Daten festgestellt werden. Letztendlich sollte, bedingt durch die Problemstellung, die Ähnlichkeit zwischen Autos mathematisch gefasst werden. Es wurden einige Tests mit Entscheidungsbäumen durchgeführt, die die vorhandenen Daten in Ähnlichkeitsgruppen einteilen sollten. Es konnten gute Resultate hinsichtlich der Trennschärfe erzielt werden, allerdings war es nicht möglich die Ergebnisse abschließend zu verifizieren und zu erklären. Deshalb wurde das Ähnlichkeitsmaß manuell durch die Expertise der Ford Mitarbeiter definiert.

Den zweiten Teil und gleichzeitigen Kern dieser Arbeit bilden der aus den historischen Daten generierte Übergangsgraph und die darauf operierenden linearen Optimierungsmodelle.

Mittels des Übergangsgraphen und dessen Knoten und Kanten sind valide Autos und mögliche Umplanungen zwischen diesen kodiert worden. Gleichzeitig diente der Übergangsgraph als Basis und Speicher für gewonnene Erkenntnisse aus dem ersten Teil der Arbeit. Beispielsweise flossen die durch die Assoziationsanalyse gewonnenen Informationen in die Graphgestalt ein. Obwohl für ihren ursprünglichen Zweck nicht verwendbar, lieferten Entscheidungsbäume einen Eindruck von

der unterschiedlichen Gewichtung, mit der Attribute in die Definition des Ähnlichkeitsmaßes eingegangen sind. Diese Definition hatte direkten Einfluss auf die Kantenumenge des Übergangsgraphen.

In ihrer Entscheidungsvariante entsprechen zulässige Lösungen von ILP1 und ILP2 einem validen Umbau, der der Attributerhaltung genügt. Weiterhin ist die Modellierung so flexibel gehalten worden, dass das Grundmodell durch eine Vielzahl von Erweiterungen hinsichtlich der Realität verbessert werden konnte. Dazu gehören das Hinzufügen einer multikriteriellen Zielfunktion und die per Attribut flexible Relaxierung der Attributerhaltung, da für viele Attribute eine Lagerung möglich ist.

Die erzielten Ergebnisse lieferten nach Rücksprache mit Experten von Ford realitätsnahe und anwendbare Vorschläge für Umplanungen. Durch eine feine Justierung der Modellparameter, wie dem Suchzeitraum, der Ähnlichkeitsschranke oder dem Qualitätsmaß, können markt- und modellspezifische Lösungen generiert werden.

Im dritten Teil wurde die Komplexität der Entscheidungsvariante des Auto Planungs Modifikations Problems genauer untersucht. Ziel war es entweder durch die Einordnung des Problems in die Komplexitätsklasse P die Konstruktion eines effizienten Algorithmus zu motivieren, oder durch den Nachweis der NP-Vollständigkeit die Verwendung von ganzzahliger linearer Programmierung zu rechtfertigen. Insgesamt wurde für den Beweis der NP-Vollständigkeit ein neues Entscheidungsproblem, genannt Hyper Binary Integer Programming, eingeführt und dessen Komplexität durch Reduktion von einem Spezialfall von dem Erfüllbarkeitsproblem der Aussagenlogik gezeigt. Aufgrund der konstruierten Gestalt des H-INT Problems, konnte eine ersichtliche Reduktion auf das Auto Planungs Modifikations Problem geführt und auf diese Weise dessen NP-Vollständigkeit bewiesen werden.

Desweiteren ist der Grund für die hohe Komplexität des Problems bestimmt worden. Ursache bilden hierbei die geforderte Attributerhaltung in Kombination mit der Möglichkeit mehrere Attribute eines Autos im Zuge einer Umplanung zu verändern. Der Beweis für die anfängliche Vermutung ist durch eine Reduktion, des auf eine Attributfamilie eingeschränkten APMPs, auf das kürzste Wege Problem erbracht worden.

Ausblick

Insgesamt steht durch diese Arbeit ein Gerüst zur Verfügung, um das Auto Planungs Modifikations Problem zu lösen. An diesem Gerüst gibt es einige Stellen, die durch weitere und tiefer gehende Forschung und die Verfügbarkeit von zusätzliche Daten verbessert werden können.

Eine wahrscheinlich hilfreiche Zusatzinformation wäre die Verfügbarkeit einer genauen Produktspezifikation für ein Automodell. Dadurch könnte die Knotenmenge exakt definiert werden und ein Optimierungsmodell, wie es in dieser Arbeit vorgestellt wurde, wäre aufgrund der erhöhten Auswahlmöglichkeit an Umplanungen öfter in der Lage eine zulässige Lösung finden. Die erhöhte Knotenanzahl konnte in dieser Arbeit nur abgeschätzt werden, allerdings wird vermutet, dass die Größenordnung, aufgrund der vielen Beschränkungen innerhalb der Teilekombinationen, beibehalten werden kann und sich somit keine negativen Auswirkungen für die Laufzeit ergeben.

Würde die Knotenmenge auf diese Weise konstruiert werden, muss zusätzliche Arbeit im Bereich des Qualitätsmaßes erfolgen. Während aufgrund der Konstruktion für jedes zu einem Knoten gehörige Auto bekannt war, ob und wie schnell es verkauft wurde, sind diese Daten bei der Verwendung der Produktspezifikation nicht vorhanden. Es müsste genauer analysiert werden, wie einzelne Autoteile und vor allem deren Kombinationen die Verkaufbarkeit beeinflussen.

Bekannt sind zwei Annahmen, durch die sich von der Realität wegbewegt wird und die gleichzeitig einen hohen Einfluss auf die bisherigen Ergebnisse haben.

Dafür sei als erstes die geforderte Attributerhaltung für alle Attribute genannt. Wie schon an mehreren Stellen der Arbeit angemerkt, ist es Ford möglich einige Komponenten zu lagern. Nach Auskunft von Ford Mitarbeitern liegt die Anzahl der zwingend lagerlosen Komponenten ungefähr bei 20. Kann die zugehörige Nebenbedingung für alle anderen Attribute relaxiert werden, wobei es sich ausschließlich um optionale Features handeln wird, ist mit einer deutlichen Steigerung der umplanbaren Anfragen zu rechnen.

Eine deutliche Reduzierung der erfüllbaren Anfragen würde die Korrektur der zweiten Annahme nach sich ziehen. Wie anfangs erläutert, ist jeder Order eine Händlernummer zugeordnet. Innerhalb der Arbeit wurde nach einer Umplanung, basierend auf allen BTS-Autos eines Tages, gesucht. Wird diese Menge durch die jeweiligen Händler partitioniert, wird es aufgrund der begrenzten Anzahl an BTS-Autos fast keine möglichen Umplanungen mehr geben.

7 Zusammenfassung & Ausblick

Um im laufenden Betrieb eingesetzt werden zu können, müsste zum Teil eine bessere Laufzeit erreicht werden. Eine effizientere Implementierung, geschrieben in einer nicht prototypbasierten Programmiersprache, und das Entfernen von redundanten Daten, die zur Fehlersuche benutzt worden sind, sollte zu einigen Laufzeitverbesserungen führen.

Die wohl größte Verbesserung könnte dadurch erreicht werden, dass das Suchen nach einer zulässigen Lösung durch das Modell für die jeweiligen Tage im Suchzeitraum parallelisiert würde. In den dargestellten Laufzeiten der unterschiedlichen Formulierungen ist die Suche des Modells für den überwiegenden Anteil der Laufzeit verantwortlich. Die Laufzeit muss im schlechtesten Fall, falls der gesuchte Knoten für jeden Tag im Graphen enthalten ist, mit der Anzahl der Suchtage multipliziert werden. Die Voraussetzungen für einen Suchtag stehen nicht in Abhängigkeit zu anderen Suchtagen, wodurch sich eine Parallelisierung problemlos durchführen lassen sollte. Beibehalten werden könnte der Ansatz mögliche Teilprozesse als Pre-processing auszulagern.

Literaturverzeichnis

- [V80] J. Virágh. Deterministic ascending tree automata I. *Acta Cybernetica*, pp. 33–44, 1980.
- [M67] J. B. MacQueen. Some Methods for classification and Analysis of Multivariate Observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*. 1, University of California Press, pp. 281–297, 1967.
- [L82] S. P. Lloyd. Least squares quantization in PCM. In: *IEEE Transactions on Information Theory*, 28, pp. 129–137, 1982.
- [A07] D. Arthur, S. Vassilvitskii. The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, pp. 1027-1035, 2007.
- [H97] Z. Huang. A Fast Clustering Algorithm to Cluster very Large Categorical Data Sets in Data Mining, In *DMKD*, 1997.
- [E96] M. Ester, H. P. Kriegel , J. Sander, X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, In *KDD*, pp. 226—231, 1996.
- [H00] J. Han, H. Pei, Y. Yin. Mining Frequent Patterns without Candidate Generation. In *Proc. Conf. on the Management of Data*. ACM Press, New York, NY, USA 2000.
- [A93] R. Agrawal, T. Imielinski, A. Swami. Mining associations between sets of items in large databases. In *Proceedings of the ACM SIGMOD Int’l Conference on Management of Data*, pp. 207-216, Washington D.C., 1993.
- [B97] S. Brin, R. Motwani, J. D. Ullman, S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proceedings of the ACM SIGMOD Int’l Conf. on Management of Data*, pp. 265-276, 1997.
- [Q86] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1), pp. 81-106, 1986.

Literaturverzeichnis

- [Q93] J. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.
- [E48] E. Shannon. A Mathematical Theory of Communication. Bell System Technical Journal 27 (3), pp. 379–423, 1948.
- [K72] Richard M. Karp. Reducibility Among Combinatorial Problems. In R. E. Miller, J. W. Thatcher: Complexity of Computer Computations. Plenum Press, pp. 85–103, New York, 1972.
- [C71] Stephen A. Cook. The complexity of theorem-proving procedures. In: Proceedings of the 3rd Annual ACM Symposium on the Theory of Computing. ACM, pp. 151–158, New York 1971.
- [D59] Edsger W. Dijkstra. A note on two problems in connexion with graphs. In: Numerische Mathematik. 1, pp. 269–271, 1959.
- [A93] Rakesh Agrawal, Tomasz Imieliński, Arun Swami. Mining Association Rules between Sets of Items in Large Databases. In: Peter Buneman; Sushil Jajodia (Hrsg.): Proceedings of the 1993 ACM SIGMOD international conference on Management of data. ACM, New York NY, 1993
- [L81] Harry R. Lewis, Christos Papadimitriou. Elements of the Theory of Computation (1st ed.). Prentice-Hall. Section 4.6: Nondeterministic Turing machines, pp. 204–211, 1981.
- [S78] Thomas J. Schaefer. The complexity of satisfiability problems. Proceedings of the 10th Annual ACM Symposium on Theory of Computing. San Diego, California, pp. 216–226, 1978.