Universität Osnabrück

Fachbereich Mathematik/Informatik

Seminararbeit

zum Masterseminar Optimierung im Wintersemester 2012

Thema: The Building Evacuation Problem with Shared Information

eingereicht von: Manuel Schwarz <manschwa@uos.de>
Matrikelnr.: 935488

eingereicht: Januar 2013

Betreuerin: Frau Prof. Dr. Sigrid Knust

Inhaltsverzeichnis

1 Einleitung

Die vorliegende Seminararbeit entstand im Rahmen des Masterseminars Optimierung an der Universität Osnabrück im Wintersemester 2012 unter der Leitung von Prof. Dr. Sigrid Knust als Ausarbeitung eines 60-minütigen Vortrages.
Das Seminar hatte zum Ziel Einblicke in verschiedene Anwendungsbereiche der Optimierung zu geben. Sowohl die vorliegende Ausarbeitung als auch der zugehörige Vortrag beschäftigen sich mit dem Problem der Gebäudeevakuierung mit geteilter Information, bzw. Shared Information und basieren auf den beiden Artikeln "*The Building Evacuation Problem with Shared Information*"[CMH08] sowie "*Lecture Note on Benders' Decomposition*"[CP08]. Hauptsächlich geht es darum, bestehende Optimierungsverfahren der Gebäudeevakuierung zu verbessern, um effizientere und sicherere Evakuierungswege aus Gebäuden zu finden und somit am Ende eventuell Leben zu retten. Das Ziel, welches erreicht werden soll, ist die Minimierung der Gesamtevakuierungszeit, also die Zeit, die benötigt wird, bis die letzte Person ein gefährdetes Gebäude verlassen hat. All das geschieht unter Annahme der *Shared Information*, das heißt, das stets der aktuelle Gebäudezustand bekannt ist und jede Person live mit den für sie relevanten Informationen versorgt wird.

In den folgenden Kapiteln wird zunächst eine genauere Beschreibung des Problems gegeben, worauf anschließend die Modellierung als gemischt-ganzzahliges lineares Programm (MIP) aufbaut. Aus Effizienzgründen wird zusätzlich das Lösungsverfahren *Benders Decomposition* vorgestellt, mit dessen Hilfe eine Evakuierung eines realen 4-stöckigen Gebäudes simuliert wird. Daraufhin wird kurz auf die Ergebnisse und Verbesserungsmöglichkeiten dieses Experimentes eingegangen, bevor eine knappe Zusammenfassung sowie ein Fazit die Ausarbeitung abschließen.

2 Problembeschreibung

Das *Building Evacuation Problem with Shared Information* oder kurz *BEPSI* bestimmt eine Zuordnung von Personen und Routen aus Gebäuden im Gefahrenfall und ist NP-schwer. Dabei ist der Begriff Gebäude allgemein gehalten und steht für Konstruktionen, die Menschen behausen können, also zum Beispiel Hochhäuser, Gebäudekomplexe (Universitätsgebäude) oder große Schiffe. Ein klassischer, oft herangezogener Gefahrenfall, ist ein ausgebrochenes Feuer. Aber auch andere Gefahren, wie beispielsweise Naturkatastrophen oder ein militärischer Angriff sind vorstellbar.
Das Ziel ist nun eine Zuordnung zu finden, die alle Personen in möglichst wenig Zeit aus dem Gebäude navigiert. Bisherige Ansätze sind unter anderem statische Fluchtpläne, die in jedem größeren und öffentlichen Gebäude aushängen. Diese Pläne werden einmal erstellt und können einen veränderten Gebäudezustand nicht erfassen. Aus diesem Grund führen sie im Fall eines Feuerausbruchs möglicherweise in eine Gefahrensituation hinein anstatt heraus. Zudem sind sie in einer Stresssituation eventuell nicht auf Anhieb zu erfassen oder unzugänglich, folglich ist die Person auf sich allein gestellt eine gute Evakuierungsroute zu finden. Des Weiteren wird bei bestehenden Ansätzen beliebiges Aufteilen von Personengruppen erlaubt, was nachgewiesener Weise nicht der Realität entspricht.
In diesem Ansatz wird ein realistischeres Gruppenverhalten simuliert, indem das Splitten von Gruppen nicht gestattet ist und somit dem Wunsch der Gruppe nach kollektivem Handeln nachgekommen wird. Im konkreten Fall bedeutete dies, dass Kollegen aus dem selben Büro oder Flur bei einer Evakuierung zusammen bleiben. Die am häufigsten auftretenden Probleme, die durch das *BEPSI* behoben werden, sind zu große Menschenansammlungen bzw. Stau sowie eine schlechte Wegwahl der Personen. Um dies zu vermeiden wird die *Shared Information* mit einbezogen. Unter dieser "geteilten Information" kann man sich vereinfachend "vollständige Information" vorstellen, wodurch dieses System wesentlich dynamischer operiert als beispielsweise die klassischen Fluchtpläne. Das bedeutet, dass der Gebäudezustand zu jedem Zeitpunkt bekannt ist und die Evakuierungswege dementsprechend situationsbedingt aktualisiert werden können. Dabei werden neue Informationen live an die betreffenden Personen weitergeleitet. Genau dies ist der Flaschenhals des gesamten Modells, da die Voraussetzung für ein solches Vorgehen ein dichtes Sensorsystem im gesamten Gebäude fordert, um stets überall den aktuellen Zustand abzufragen. Weiterhin müssen die Informationen bei den Personen ortsbedingt ankommen. Hier könnte man sich veränderliche Hinweisschilder oder ein Sprachsystem vorstellen oder jede andere Technologie, die die Voraussetzung der Informationsweiterleitung unter suboptimalen Bedingungen (wie Rauch oder Dunkelheit) erfüllt.

3 Modellierung als gemischt-ganzzahliges lineares Programm

Da bei der Formulierung als MIP relativ viele Variablen definiert werden müssen, ist dieses Kapitel aus Gründen der Übersichtlichkeit in zwei Teile unterteilt. Zunächst wird in einem ersten Abschnitt der Großteil der Variablen eingeführt, worauf dann im zweiten Abschnitt das Modell beschrieben wird.

3.1 Variablen

Bei der Überführung in ein Modell bietet sich für das *BEPSI* die Netzwerkrepräsentation eines Gebäudes an. Dabei setzt sich das Netzwerk $\Im = (G, u, \tau)$ zunächst aus einem Graphen $G = (N, A, \{0, \ldots, T\})$ zusammen, der das Layout des Gebäudes repräsentiert. Die Knoten $N = \{1, \ldots, n\}$ stellen Orte bzw. Positionen, wie z. B. Büros, Meetingräume, Ausgänge oder Kreuzungen innerhalb des Gebäudes dar. Dementsprechend stehen die gerichteten Kanten $A = \{(i, j) | i, j \in N\}$ für die Verbindungswege zwischen diesen Orten (z. B. Treppenhaus, Aufzugschaft, Flur). T stellt den zu analysierenden Zeitraum dar, der in viele kleine Zeitintervalle $t \in \{1, \ldots, T\}$ diskretisiert wird. Die Kantenkosten $\tau_{ij}(t)$ entsprechen der Zeit, die benötigt wird, um den Weg zwischen zwei Positionen (i, j) zum Zeitpunkt t zurückzulegen (im Folgenden Reisekosten genannt). Da nicht beliebig viele Menschen die gleiche Kante zur gleichen Zeit wählen können, wird die Kapazität der Kante (i, j) zum Zeitpunkt t durch $u_{ij}(t)$ limitiert. Die Kapazität hängt dabei ganz von der Art und Größe des Weges ab. Damit die Zulässigkeit des Problems gewährleistet ist, müssen sogenannte Pufferzonen bzw. Wartekanten $(i, i),\ \forall i \in N$ eingeführt werden, die eine Reisezeit von 1 und eine Kapazität von ∞ besitzen. Um das Problem der Evakuierung zu formalisieren, wird eine Menge von Quellknoten $K = \{k_1, k_2, \ldots, k_M\}$ sowie die Anzahl (M) dieser und ein Senkeknoten l definiert. Die Quellknoten sind die Startpunkte der zu evakuierenden Personen und der Senkeknoten ist der Gebäudeausgang. O.B.d.A. wird angenommen, dass es nur einen Exit gibt. Dennoch könnte man ohne Schwierigkeiten mehrere Ausgänge modellieren, indem alle Senkeknoten in eine Supersenke münden, wobei die Reisezeit 0 und die Kapazität erneut ∞ betrüge.
Weiterhin wird mit $b_{k_m}(t)$ der Supply, das Angebot oder der "Nachschub" von Personen beim Quellknoten $k_m \in K$ zum Zeitpunkt t bezeichnet. Bei $t = T$ entspricht der Supply im Knoten l dem Gesamtsupply B, was bedeutet, dass sich alle Personen am Ausgang befinden. Man schreibt auch

$$B = \sum_{k_i \in K} \sum_{t=1}^{T} b_{k_i}(t) \ \text{, sodass } \ b_l(T) = -B \tag{3.1}$$

Das negative Vorzeichen resultiert daraus, dass der Senkeknoten die von den Quellknoten kommenden Personen "aufnehmen" muss. Zusätzlich wird angenommen, dass die Reisezeit, die Kapazität und der Supply zwar bekannt, aber zustandsabhängig und somit variabel sind.

3.2 Modell

Nun wird das *BEPSI* als gemischt-ganzzahliges lineares Programm formuliert. Die nicht-negative Entscheidungsvariable $x_{i,j}(t)$ repräsentiert den Fluss von Knoten i zum Zeitpunkt t entlang der Kante (i,j). Mit Fluss ist dabei eine Anzahl von Personen gemeint. Gleichzeitig legt die Binärvariable $\lambda_{i,j}(t)$ fest, welche Kanten auf dem Weg zum Ausgang gewählt werden. Unter Berücksichtigung der Reisezeit trifft der Fluss $x_{i,j}(t)$ zur Zeit $t + \tau_{i,j}(t)$ bei dem Knoten j ein. Schließlich gibt es noch Bezeichner für die eingehenden ($\Gamma^{in}(i) = \{j|(j,i) \in A\}$) sowie die ausgehenden Kanten ($\Gamma^{out}(i) = \{j|(i,j) \in A\}$) eines Knotens. Die Formulierung als MIP ergibt sich nun wie folgt.

$$P : \min \sum_{(i,j)\in A} \sum_{t\in\{0,\dots,T\}} \tau_{ij}(t)x_{ij}(t) \tag{3.2}$$

subject to:

$$\sum_{j\in\Gamma^{out}(i)} x_{ij}(t) - \sum_{j\in\Gamma^{in}(i)} \sum_{\{\bar{t}|\bar{t}+\tau_{ji}(\bar{t})=t\}} x_{ji}(\bar{t}) = b_i(t), \quad \forall i \in N, t \in \{0,\dots,T\} \tag{3.3}$$

$$\lambda_{ij}(t) \leq x_{ij}(t) \leq \lambda_{ij}(t)u_{ij}(t), \quad \forall (i,j) \in A, t \in \{0,\dots,T\} \tag{3.4}$$

$$\sum_{j\in\Gamma^{out}(i), j\neq i} \lambda_{ij}(t) \leq 1, \quad \forall i \in N\backslash l, t \in \{0,\dots,T\} \tag{3.5}$$

$$x_{ij}(t) \geq 0, \ \lambda_{ij}(t) \text{ binary}, \quad \forall (i,j) \in A, t \in \{0,\dots T\} \tag{3.6}$$

Wie bereits angesprochen minimiert die Zielfunktion 3.2 die Gesamtzeit, die benötigt wird, um alle Personen zu evakuieren, bzw. um den Gesamtfluss von den Quell- zu dem Senkeknoten zu bewegen. Die Nebenbedingung 3.3 stellt die Flusserhaltung sicher, das bedeutet, dass der ausgehende Fluss abzüglich dem eingehenden Fluss dem Supply an Knoten i zum Zeitpunkt t entspricht. Zusätzlich muss die Kapazität der Kanten berücksichtigt werden, wodurch der Fluss auf Kante (i,j) niemals größer werden darf als es die Kapazität erlaubt (Nebenbedingung 3.4). Die Bedingung 3.5 wird Shared Information Constraint genannt, der sicherstellt, dass jeweils nur eine Kante von einer Personengruppe ausgewählt wird. Zusammen mit der Kapazitätsbeschränkung garantieren diese beiden Randbedingungen, dass die Aufteilung (Splitting) von Gruppen (Fluss) verhindert wird. Der einzige erlaubte Split ist zwischen einer gleichbleibend ausgehenden Kante und einer Pufferkante des entsprechenden Knotens. Das heißt im konkreten Fall, dass sich Personengruppen nur kurzfristig aufteilen dürfen, wenn sichergestellt ist, dass sie den selben Weg gehen. Als letztes wird noch die nicht-negativ Bedingung 3.6 eingeführt.

4 Exaktes Lösungsverfahren Benders Decomposition

Der Benders Decomposition (BD) Algorithmus ist ein Verfahren, das es erlaubt Optimierungsprobleme mit sehr vielen Variablen schnell zu lösen. Mit dieser Voraussetzung ist es geradezu prädestiniert dafür zur Lösung des *BEPSI* eingesetzt zu werden, denn es wird schnell deutlich, dass für das Evakuierungsproblems sehr viele Entscheidungsvariablen für den Fluss benötigt werden, sobald das Netzwerk etwas größer wird. Nachdem das Verfahren allgemein erläutert wird, widmet sich diese Kapitel der Anwendung des BD Algorithmus' für das *BEPSI*.

4.1 Allgemeines Verfahren

Die Idee hinter dem BD Algorithmus ist, dass man das Ursprungsproblem zunächst in ein Sub- und ein Masterproblem unterteilt. Anschließend wird das lineare Programm (LP) des Subproblems dualisiert, also in ein Binärproblem umgeformt, da dies dann leicht zu lösen ist. Die Lösung des so entstandenden Subproblems liefert neue Nebenbedingungen (Constraints), die jeweils eine untere Schranke für den Zielfunktionswert darstellen und dem Masterproblem hinzugefügt werden, um dieses schließlich zu lösen.

Zunächst soll kurz die Umformung eines allgemeinen LP-Problems in ein Dual-Problem gezeigt werden. Gegeben sei dazu das folgende LP.

$$\begin{aligned} \min\ & z = cx \\ \text{s.t. } & Ax \geq b \\ & x \geq a \end{aligned} \tag{4.1}$$

Dabei steht A für die Randbedingungsmatrix und c für den Kostenvektor. Dualisiert man nun 4.1 so erhält man folgendes Binärproblem.

$$\begin{aligned} \max\ & w = ub \\ \text{s.t. } & uA \leq c \\ & u \geq a \end{aligned} \tag{4.2}$$

Die Dualvariablen werden in diesem Fall durch u repräsentiert. Findet man eine optimale Lösung für 4.2, so ist dies gleichzeitig eine optimale Lösung für 4.1.
Neben dieser linearen Dualität existiert zudem die allgemeine Dualität, auf die im Folgenden

etwas näher eingegangen wird. Formuliert man 4.1 als allgemeines Dualproblem, so gilt:

$$\begin{aligned} &\max\ \beta \\ &\text{s.t. } x \in S \xrightarrow{D} 4x \geq \beta \end{aligned} \tag{4.3}$$

D bezeichnet dabei den Definitionsbereich mit $D = \{x | x \geq a\}$, bzw. steht es für die nicht-negativ Bedingung. Man liest "$Ax \geq b$ impliziert $cx \geq \beta$ unter der Beachtung von D". Auch hier gilt wieder, dass 4.1 und 4.3 die selbe optimale Lösung besitzen.

Beispiel

Anhand eines Beispiels soll einmal die Umformung von einem LP zu einem allgemeinen Dualproblem verdeutlicht werden. Es sei das folgende einfache LP gegeben:

$$\begin{aligned} \min\ z &= 4x \\ \text{s.t. } x &\geq 5 \\ x &\leq 10 \\ x &\geq 0 \end{aligned} \tag{4.4}$$

Daraus ergibt sich für $D = \{x | x \geq 0\}$ und $S = \{x | x \geq 5 \wedge x \leq 10\}$ womit nun das allgemeine Dualproblem aufgestellt werden kann.

$$\begin{aligned} &\max\ \beta \\ &\text{s.t. } x \in S \xrightarrow{D} 4x \geq \beta \end{aligned} \tag{4.5}$$

Auch, wenn das Problem leicht im Kopf zu lösen ist, soll deutlich gemacht werden, was 4.5 aussagt. Es ist β so zu maximieren, dass die Bedingung $4x \geq \beta$ für jedes $5 \leq x \leq 10$ erfüllt ist. Offensichtlich ergibt sich die Lösung aus $x = 5$ mit $\beta = 20$. Damit wird zusätzlich eine untere Schranke für den Zielfunktionswert aus 4.4 gegeben, die zeitgleich die optimale Lösung ist.

Generelle Methodik

Der erste Schritt des Benders Decomposition Algorithmus' ist die Fixierung bestimmter Variablen des Ursprungsproblems 4.1, was zur Folge hat, dass das resultierende Subproblem einfach zu lösen ist. Das Kernstück von Benders' Decomposition ist die Auswahl dieser zu fixierenden Variablen. Zum einen hängt diese stark vom Problem selbst und zum anderen von dem Wissen über ebendies ab.
Dazu werden die Variablen x des Ursprungsproblems in zwei Gruppen (x und y) aufgeteilt. Man erhält dadurch folgendes LP (umformuliert):

$$\begin{aligned} \min\ z &= cx + f(y) \\ \text{s.t. } Ax + g(y) &\geq b \\ x, y &\in D \end{aligned} \tag{4.6}$$

$g(y)$ stellt einen Vektor von Funktionen für jedes y dar. Nun lassen sich die Variablen y mit Testwerten $\overline{y}$ fixieren, wodurch das folgende Subproblem erzeugt wird.

$$\begin{aligned} \min\ & z = cx + f(\overline{y}) \\ \text{s.t. } & Ax \geq b - g(\overline{y}) \\ & x \in D \end{aligned} \tag{4.7}$$

Um nun eine untere Schranke für den Zielfunktionswert zu erhalten, bzw. eine weitere Nebenbedingung für das Masterproblem, wird das Subproblem 4.7 zu einem allgemeinen Binärproblem umgeformt.

$$\begin{aligned} &\max\ \beta \\ &\text{s.t. } Ax \geq b - g(\overline{y}) \xrightarrow{x,\overline{y} \in D} cx + f(\overline{y}) \geq \beta \end{aligned} \tag{4.8}$$

Beschränkt man sich ausschließlich auf das lineare Subproblem, so kann dies wie folgt umformuliert werden.

$$\begin{aligned} \max\ & w = u(b - g(\overline{y})) + f(\overline{y}) \\ \text{s.t. } & uA \leq c \\ & u \in D \end{aligned} \tag{4.9}$$

Die Lösung des allgemeinen Dualproblems liefert nun eine untere Schranke β^* für den Zielfunktionswert, unter der Annahme $y = \overline{y}$ (fixiertes y). Das Ziel ist es nun eine Funktion $\beta_{\overline{y}}(y)$ zu bestimmen, die eine gültige untere Schranke für jedes y liefert. Im oberen Fall bedeutete dies, dass $\beta_{\overline{y}}(\overline{y}) = \beta^*$. Wie bereits angesprochen, bedarf das Finden dieser Funktion oft sehr spezifischer Problemkenntnis. Mit Hilfe des Dualproblems 4.9 lässt sich die generalisierte Schranke $u(b - g(\overline{y})) + f(\overline{y})$ ableiten. Durch die Lösung des allgemeinen Binärproblems 4.8 kann die Funktion $z \geq \beta_{\overline{y}}(y)$ abgeleitet werden, die wir im Folgenden *Benders Cut* nennen. Folglich sieht ein solcher *Benders Cut* oder auch Zusatzconstraint für eine Schranke so aus:

$$z \geq u(b - g(y)) + f(\overline{y}) = \beta_{\overline{y}}(y) \tag{4.10}$$

Testet man nun verschiedene Werte $\overline{y}$, so erhält man mehrere Cuts. Alle Cuts gruppiert ergeben schließlich das sogenannte *Benders' Masterproblem*:

$$\begin{aligned} \min\ & z \\ \text{s.t. } & z \geq \beta_{y^k}(y), \quad k = 1, \ldots, K \\ & y \in D_y \end{aligned} \tag{4.11}$$

Hierbei repräsentiert y^k die Testwerte, die in der k-ten Iteration verwendet wurden. Das Lösen des Masterproblems ergibt die nächsten Werte für y zum Testen in der folgenden Iteration.

Algorithmus

Es soll jetzt das oben beschriebene Vorgehen als Algorithmus formuliert werden, der hier in Form von Pseudocode dargestellt wird (siehe Algorithmus 1). Zu Beginn werden Testwerte $\overline{y}$

Algorithm 1 BENDERS DECOMPOSITION

```
1: Choose ȳ in original problem
2: z̄ ← −∞
3: k ← 0
4: while (subproblem dual has feasible solution β ≥ z̄) do
5:    derive lower bound function β_ȳ(y) with β_ȳ(ȳ) = β
6:    k ← k + 1
7:    y^k ← ȳ
8:    add z ≥ β_ȳ(y) to masterproblem
9:    if (masterproblem is infeasible) then
10:      Stop. The original problem is infeasible.
11:   else
12:      let (z̄, ȳ) be the optimal value and solution to the masterproblem
13:   end if
14: end while
15: return (z̄, ȳ)
```

für die Variablen in y festgelegt. Die Wahl von $\overline{y}$ hat dabei eventuell Einfluss auf die Anzahl von Iterationen des Algorithmus' (zufällige Wahl oder guter Schätzwert). Zudem wird der Iterationszähler auf $k = 0$ und die untere Schranke für den Zielfunktionswert auf $\overline{z} = -\infty$ gesetzt. Nun wird versucht das Subproblem unter Berücksichtigung von $\overline{y}$ zu lösen. Falls es keine gültige Lösung gibt, so endet der Algorithmus, da somit auch das Originalproblem keine gültige Lösung besitzt. Sollte $\beta = \overline{z}$ sein, so kann ebenfalls gestoppt werden, da eine optimale Lösung gefunden wurde. Andererseits ($\beta > \overline{z}$) wird eine für alle y gültige Funktion $\beta_{\overline{y}}(y)$ zur Berechnung einer unteren Schranke mit Hilfe von $\beta = \beta_{\overline{y}}(\overline{y})$ ermittelt.
Bevor der *Benders Cut* $z \geq \beta_{\overline{y}}(y)$ zum Masterproblem hinzugefügt wird, erhöht man den Iterationscounter k um eins und setzt $y^k = \overline{y}$. Anschließend wird versucht, das Masterproblem zu lösen. Sollte keine Lösung gefunden werden, so kann an dieser Stelle abgebrochen werden, da folglich das Ursprungsproblem keine gültige Lösung besitzt. Kann hingegen eine Lösung gefunden werden, so wird $\overline{y}$ als neue optimale Lösung des Masterproblems angenommen und als neuer Testvektor der nächsten Iteration verwendet, in der das Subproblem erneut mit dem optimalen Wert $\overline{z}$ und dem neuen $\overline{y}$ gelöst wird.

4.2 Verfahren anhand des BEPSI

Nachdem der *Benders Decomposition* Algorithmus nun allgemein erläutert wurde, folgt nun die konkrete Anwendung auf das *BEPSI*. Als Subproblem wird die Bestimmung des Flusses anhand der Entscheidungsvariablen $x_{ij}(t)$ gewählt. Das Masterproblem stellt dementsprechend das Problem der Kantenauswahl anhand der Binärvariablen $\lambda_{ij}(t)$ dar. Fixiert werden die Werte von λ mit $\tilde{\lambda} \in \Lambda$, wobei $\tilde{\lambda}$ die fixierten Werte und Λ die Menge aller zulässigen λ beschreibt.
Vor der Neuformulierung des Subproblems wird die Kantenmenge $A = \{(i,j)|i,j \in N\}$ zunächst

in drei disjunkte Teilmengen unterteilt. Dies geschieht einzig aus Performancegründen. Die Laufzeit kann somit bis um den Faktor 10 verringert werden.

$$I_1(A) = \{(i,j)|i,j \in N \text{ and } \Gamma^{out}(i) \geq 2\}$$
$$I_2(A) = \{(i,j)|i,j \in N \text{ and } \Gamma^{out}(i) = 1\}$$
$$I_3(A) = \{(i,i)|i \in N\}$$

Nun kann das Subproblem wie folgt als gemischt-ganzzahliges lineares Programm notiert werden:

$$\mathrm{RS_p}(\tilde{\lambda}): \ \min \sum_{(i,j)\in A} \sum_{t\in\{0,\ldots,T\}} \tau_{ij}(t)x_{ij}(t) \tag{4.12}$$

subject to:

$$\sum_{j\in\Gamma^{out}(i)} x_{ij}(t) - \sum_{j\in\Gamma^{in}(i)} \sum_{\{\bar{t}|\bar{t}+\tau_{ji}(\bar{t})=t\}} x_{ji}(\bar{t}) = b_i(t), \quad \forall i \in N, \ t \in \{0,\ldots,T\} \tag{4.13}$$

$$x_{ij}(t) \leq \tilde{\lambda}_{ij}(t)u_{ij}(t), \quad \forall (i,j) \in I_1(A), \ t \in \{0,\ldots T\} \tag{4.14}$$

$$x_{ij}(t) \leq u_{ij}(t), \quad \forall (i,j) \in I_2(A), \ t \in \{0,\ldots T\} \tag{4.15}$$

$$x_{ij}(t) \geq 0, \quad \forall (i,j) \in A, \ t \in \{0,\ldots T\} \tag{4.16}$$

Die Zielfunktion 4.12, Flusserhaltungsbedingung 4.13 sowie die nicht-negativ Bedingung 4.16 sind deckungsgleich mit dem Ursprungsproblem aus Abschnitt 3.2. Die Kapazitätsbeschränkung hingegen ist auf die Knotengruppen aufgeteilt worden (Constraints 4.14 und 4.15). Der Shared Information Constraint fällt weg, da λ fixiert und somit konstant ist. Als nächstes wird das Subproblem dualisiert, um es einfacher lösen zu können und die Constraints für untere Schranken des Zielfunktionswertes zu erhalten, die anschließend dem Masterproblem hinzugefügt werden.

$$\mathrm{DRS}_p(\tilde{\lambda}): \max \sum_{t\in\{0,\ldots,T\}} \left(\sum_{i\in N} \pi_i(t)b_i(t) + \sum_{(i,j)\in I_2(A)} u_{ij}(t)m_{ij}(t) + \sum_{(i,j)\in I_1(A)} \tilde{\lambda}_{ij}(t)u_{ij}(t)m_{ij}(t) \right) \tag{4.17}$$

subject to:

$$\pi_i(t) - \pi_j(t+\tau_{ij}(t)) + m_{ij}(t) \leq \tau_{ij}(t), \quad \forall (i,j) \in A\backslash I_3(A), \ t \in \{0,\ldots,T\} \tag{4.18}$$

$$m_{ij}(t) \leq 0, \quad \forall (i,j) \in A\backslash I_3(A), \ t \in \{0,\ldots,T\} \tag{4.19}$$

Hier steht $\pi_i(t)$ für die Dualvariablen, die aus der Flusserhaltung 4.13 resultieren und $m_{ij}(t)$ für die Dualvariablen aus den Constraints 4.14 und 4.15.
Es sei nun D der durch die Constraints 4.18 und 4.19 definierte Lösungsraum-Polyeder, dann seien P_D und R_D die Mengen der Extrempunkte bzw. der Extremlinien (Extremrays) von D. Mit diesen Definitionen wird nun das Masterproblem formuliert, dass die Flusserhaltungsbedingung, die Kapazitätsbeschränkung sowie den Shared Information Constraint durch *Benders Cuts* ersetzt. Diese Cuts (oder Constraints) setzen sich aus einem Optimalitäts-Cut 4.21 und einem Gültigkeits-Cut 4.22 zusammen, die, wie der Name schon sagt, dafür sorgen, dass nicht-optimale Lösungen ausgeschlossen werden bzw. Lösungen des Subproblems zulässig bleiben. Mit

der freien Variable Z lässt sich das Masterproblem wie folgt formulieren:

$$\overline{\mathrm{P}} : \min \ Z \tag{4.20}$$

subject to:

$$Z - \sum_{t\in\{0,\ldots,T\}} \sum_{(i,j)\in I_1(A)} u_{ij}(t)m_{ij}(t)\lambda_{ij}(t) \geq \sum_{t\in\{0,\ldots,T\}} \left(\sum_{i\in N} \pi_i(t)b_i(t) + \sum_{(i,j)\in I_2(A)} u_{ij}(t)m_{ij}(t) \right), \quad (\pi, m) \in P_D \tag{4.21}$$

$$\sum_{t\in\{0,\ldots,T\}} \sum_{(i,j)\in I_1(A)} u_{ij}(t)m_{ij}(t)\lambda_{ij}(t) \leq - \sum_{t\in\{0,\ldots,T\}} \left(\sum_{i\in N} \pi_i(t)b_i(t) + \sum_{(i,j)\in I_2(A)} u_{ij}(t)m_{ij}(t) \right), \quad (\pi, m) \in R_D \tag{4.22}$$

$$\sum_{j \ in\Gamma^{out}(i), j\neq i} \lambda_{ij}(t) \leq 1, \quad \forall i \in N\backslash l, \ t \in \{0,\ldots,T\} \tag{4.23}$$

$$\lambda_{ij}(t) \text{ binary}, \quad \forall (i,j) \in A, \ t \in \{0,\ldots,T\} \tag{4.24}$$

Der im vorherigen Abschnitt vorgestellte Algorithmus muss nun noch ein wenig an das *BEPSI* angepasst werden. Der Startvektor $\tilde{\lambda}$ für das zu lösende Subproblem ist initial mit 1en gefüllt, was bereits in der ersten Iteration zu Konflikten mit dem nicht erlaubten Flusssplitting führt und abgeändert wird. Die hieraus entstehenden Constraints werden dem Masterproblem hinzugefügt und dieses wird gelöst. Das resultierende Ergebnis ist eine verbesserte Auswahl von Kanten, die wiederum als Input des Subproblems dienen. Beim Lösen des Subproblems wird überprüft, ob der neue Zielfunktionswert kleiner ist, als der alte und alle Constraints erfüllt sind. Ist dies der Fall, so ist man fertig. Ansonsten muss der Lösungsraum durch Hinzufügen weiterer Constraints (*Benders Cuts*) eingeschränkt werden. Der Algorithmus wiederholt dies iterativ bis Konvergenz eintritt und findet für ein zulässiges Problem immer die optimale Lösung. Die Laufzeit ist im worst-case exponentiell, da unter Umständen alle Extrempunkte und -kanten des Lösungsraums (Polyeder) abgelaufen werden müssen.

5 Beispiel

Um das Verfahren zu testen wurde ein möglichst realitätsnahes Beispiel gewählt, wobei das 4-stöckige A. V. Williams Gebäude auf dem Campus der University of Maryland als Grundlage dient. Ein Grundriss des zweiten Stockwerks ist in Abbildung 5.1 zu sehen. Da das Gebäude

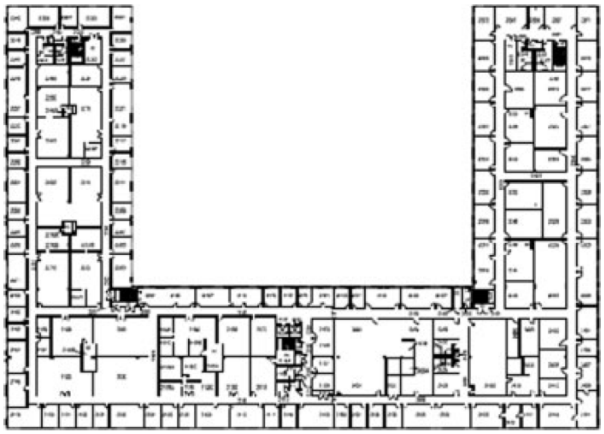

Abbildung 5.1: A. V. Williams Gebäude: Grundriss des zweiten Stocks.

aus 4 identischen Etagen besteht, genügte es die Daten eines Geschosses zu erheben. Dabei wurden unter anderem die Breiten und Längen von Korridoren, Türrahmen und Treppenhäusern für die Approximation der Kapazitäten und Reisezeiten ausgemessen sowie eine Netzwerkrepräsentation entwickelt. Fahrstühle wurden nicht berücksichtigt, da der Gebrauch im Gefahrenfall untersagt ist. Zudem wurde auf eine Modellierung von Fensterausstiegen im Erdgeschoss oder anderen Stockwerken verzichtet. Nach der Überführung des Geäudegrundrisses in ein Netzwerk, besteht dieses insgesamt aus 612 Knoten, 1480 Kanten und fünf Ausgängen. Um eine realistische Einschätzung für das Fassungsvermögen der Räume (Vorlesungssäle, Büros, Labore) und den daraus resultierenden Flussraten zu erhalten, wurde der NFPA 101 Life Safety Code [NFPA01] herangezogen. Die Flussrate wird für die später folgenden Testszenarien in drei Stufen unterteilt: average (1), maximum (2) und maximum plus (3). Dabei wurden zur Schätzung der jeweiligen Flussrate zum einen empirische Beobachtungen sowie Berechnungen [CFS82] und zum anderen charakteristische Fußgängerbewegungen [DCWW88] mit einbezogen. Der Supply orientiert sich am maximalen Fassungsvermögen der Räume.
Anhand der Tabelle 5.1 wurden Näherungswerte für die Reisezeit sowie die Kapazität der jeweiligen Kante ermittelt. Die Tabelle macht deutlich, dass ein Treppenhaus weniger Menschen fassen kann als beispielsweise ein Flur und dadurch die langsamste Alternative darstellt.

Tabelle 5.1: Bewegungsparameter für Gebäude [DCWW88].

Facility	Density $[\frac{person}{ft^2}]$	Speed $[\frac{ft}{min}]$	Flow $[\frac{person/min}{ft}]$
Doorway	0.22	120	26
Pathway	0.20	120	24
Stairwell	0.19	95	18

5.1 Szenarien

In diesem Abschnitt werden die sechs Testszenarien, die in Tabelle 5.2 aufgelistet sind und den Testläufen zugrunde liegen, näher erläutert. Die bei der Erstellung der Szenarien berücksichtigten Faktoren sind die initiale Personenanzahl an den Quellknoten, Informationen darüber, ob Treppen und Flure passierbar sind oder nicht (arbeiten die Kanten auf maximaler Kapazität bzw. Reisegeschwindigkeit) sowie die Art und den Ort des Ereignisses, das die Evakuierung ausgelöst hat. Für alle Szenarien gilt $T = 20\,min$ (betrachteter Zeitraum) und, dass alle Kanten initial frei sind. Die ersten beiden Szenarien spielen unter idealen Bedingungen, allein der Supply wird

Tabelle 5.2: Testszenarien

Scenario	Capacities	Travel times	Supply level	Severity of conditions
1	1	1	1	ideal conditions
2	1	1	3	ideal conditions
3	0.98	1.02	1	slightly impacted
4	0.98	1.02	2	slightly impacted
5	0.96	1.04	3	impacted
6	0.95	1.06	3	severely impacted, some links disabled

im zweiten Fall angehoben. In der Realität würde dies beispielsweise eine Feuerübung simulieren. Zudem wird der Gebäudezustand als gleichbleibend angenommen im Gegensatz zu den restlichen Szenarien, die eine kontinuierliche Verschlechterung der Konditionen mit einbeziehen. Dies geschieht, indem die aus der Tabelle 5.2 ersichtlichen Multiplikatoren für die Kapazität und die Reisezeit in jedem Zeitschritt mit dem aktuellen Wert multipliziert werden. Somit wird die Kapazität der Kanten stetig geringer und die Reisezeit größer. Die Testfälle sind so entwickelt worden, dass sich der Gebäudezustand mit steigender Szenariennummer verschlechtert und sich einem realistischen Szenario angenähert wird. Dementsprechend sind die Bedingungen der Szenarien drei bis fünf ungünstiger als in den ersten beiden Fällen, da diese sich nun über die Zeit verschlechtern, wobei noch keine realistische Gefahrensituation simuliert wird. Erst im sechsten und letzten Fall werden realistische Szenarienwerte erreicht. Simuliert wurde ein Feuerausbruch im Westflügel des vierten Stocks unter der Annahme, dass ein Korridor versperrt und die nächste Treppe unpassierbar sei.
Die Ergebnisse des Benders Decomposition Algorithmus werden im folgenden Abschnitt dargestellt.

5.2 Ergebnisse

Der Benders Decomposition Algorithmus wurde in Microsoft Visual Studio C++ 6.0 implementiert und auf einem PC mit einem Pentium 4 CPU 3,20 GHz und 2,0 GB RAM getestet.
Die resultierenden Ergebnisse zeigt Tabelle 5.3. In der ersten Spalte ist das jeweilige Szenario aufgeführt, die zweite Spalte beschreibt die Anzahl der Iterationen bzw. die Anzahl der Benders cuts und die Spalten drei und vier listen die Laufzeiten des Algorithmus in Sekunden auf, wobei zum einen bei Erreichen von 95 %iger Optimalität abgebrochen wurde und zum anderen die optimale Lösung berechnet wurde. Aus der Tabelle ist ersichtlich, dass die Laufzeit mit schlechter

Tabelle 5.3: Ergebnisse für das reale Netzwerk

Scenario	Number of cuts	to 95 % optimality [$CPU seconds$]	to optimality [$CPU seconds$]
1	4	-	3.0
2	4	1.6	3.3
3	12	1.9	30.8
4	36	6.0	31.2
5	32	19.6	58.5
6	44	17.7	94.8

werdendem Gebäudezustand wie erwartet zunimmt. Dabei ist interessant zu beobachten, dass die Rechenzeit bei steigendem Gebäudeverfall weniger als linear ansteigt. Zudem benötigt der Benders Decomposition Algorithmus signifikant weniger Zeit um eine 95 %ige Optimalität zu erreichen als bei der Berechnung der optimalen Lösung. Im Fall einer Evakuierung müsste das *BEPSI* zudem ständig neu gelöst werden, da sich der Gebäudezustand vermutlich kontinuiertlich ändert (insbesondere der Supply, die Kantenkapazitäten sowie -reisezeiten). Aus diesem Grund könnte es sinnvoll sein den Algorithmus bereits nach kurzer Zeit abzubrechen, was kein Problem darstellt, da dieser immer eine gültige Lösung liefert, um Zeit zu sparen und die zu evakuierenden Personen möglichst sofort mit den neusten Informationen zu versorgen. Ist beispielsweise ein Treppenhaus eingestürzt, so will man dies den Personen auf dem Weg dorthin nicht erst nach 95 Sekunden mitteilen und sie damit eventuell in eine Gefahrensituation laufen lassen, sondern besser bereits nach 18 Sekunden.
Es sei zudem noch angemerkt, dass der BD Algorithmus mit einem hier nicht aufgeführten *Branch and Cut* Algorithmus verglichen wurde, wobei BD in jedem Fall deutlich schneller war.

6 Fazit

Abschließend folgt nun eine kurze Zusammenfassung in Verbindung eines kleinen Fazits. Zunächst wurde das *Building Evacuation Problem with Shared Information* beschrieben und anschließend als gemischt-ganzzahliges lineares Programm formuliert. Um dieses MIP zu lösen wurde das exakte Lösungsverfahren *Benders Decomposition* vorgestellt. Am Beispiel eines vierstöckigen, real existierenden Gebäudes wurde dieses Verfahren demonstriert und seine Anwendbarkeit gezeigt. Das *BEPSI* bedient sich zwar bereits bekannter Verfahren, zeichnet sich aber insbesondere durch die zuvor nie betrachtete Berücksichtigung der Gruppendynamik aus. Personengruppen dürfen folglich nicht einfach an einem Knoten auf verschiedene Kanten (Wege) aufgeteilt werden. Zudem wird eine kontinuierliche Verschlechterung der Bedingungen in die Kalkulation mit einbezogen (beispielsweise das dynamische Ausbreiten eines Feuers oder zerstörte Gebäudestruktur). Dadurch können die zu evakuierenden Personen mit Hilfe ständiger Liveupdates aus Gefahrensituationen oder um diese herum geleitet werden.

Mögliche Ansätze der Verbesserung könnten zum Beispiel die Bestimmung der Kantenkosten (Reisezeit) sein. Diese hängt aktuell nur vom Netzwerkzustand ab, nicht jedoch von der Anzahl der Personen, die sich auf einer Kante befinden. Zudem muss der zu betrachtende Zeitraum T einer Evakuierung stets im Voraus angegeben werden. Dies kann unter Umständen zu einem Problem führen, wenn diese Zeitraum zu klein gewählt wird.
Da das vorgestellte Verfahren allgemein gehalten ist, kann es leicht auf andere Einsatzgebiete übertragen werden. Denkbar ist beispielsweise die Evakuierung einer geografischen Region aufgrund eines Militärangriffs, einer Naturkatastrophe oder einer durch Menschen verursachten Katastrophe (Atomunfall).
Schließlich böte sich noch die Möglichkeit zur Entwicklung heuristischer Verfahren, die deutlich schneller ein annährend gutes Ergebnis für sehr große Netzwerke liefern, um somit möglichst wenig Zeit dabei zu verlieren Menschen aus Gefahrensituationen zu führen. Die optimale Lösung des *BEPSI* könnte dazu als Richtwert (Benchmark) dienen, an dem sich die Heuristiken messen.

Literaturverzeichnis

[CFS82] Chalmet, L. G. ; Francis, R. L. ; Saunders, P. B.: Network models for building evacuation. In: *Management Sci* 28 (1982), S. 86–105

[CMH08] Chen, Lichun ; Miller-Hooks, Elise: The building evacuation problem with shared information. In: *Naval Research Logistics (NRL)* 55 (2008), Nr. 4, 363–376. http://dx.doi.org/10.1002/nav.20288. – DOI 10.1002/nav.20288. – ISSN 1520–6750

[CP08] Christensen, Uffe G. ; Pedersen, Anders B.: Lecture Note on Benders' Decomposition. (2008). http://www.andersbp.dk/studier/dat/OPPP/benders.pdf

[DCWW88] DiNenno, Philip J. ; Custer, R. L. P. ; Walton, W. ; Watts, J. M.: *The SFPE handbook of fire protection engineering.* 1. The National Fire Protection Association, 1988

[NFPA01] National Fire Protection Association, The: *NFPA 101 Life Safety Code.* Inc.1 Batterymarch Park, 2001