Democratic and Popular Republic of Algeria

Ministry of Higher Education and Rsearch

University of Saad dahleb Blida 1

Faculty of science

Department mathematics

THESIS

PRESENTED FOR THE ATTAINMENT OF A DEGREE IN:

THEORETICAL MATHEMATICS

By

OULDYAHIA Kahina

THEME

Optimizing ATM Cash Replenishment using nonlinear programming

A thesis submitted for the degree of

Master in statistical and stochastic modeling

July 2023

Mr. BOUKHARI Mohamed	President
Mrs DJEMIA Noura	examiner
Mr. BOUDJEMAA Redoune	Promoter

2022/2023

Dedicace

This thesis is dedicated to...

My almighty Allah, the most gracious who guided me to the right path, enlighten my road and helped me to overcome all the hardships I encountered throughout this study.

To my dear parents who always picked me up on time and encouraged me to go on every adventure especially this one… To my sisters and Brother who led me with light of hope and support, encouraged me to keep me moving forward.

To my dear husband who were always here for me, the person who have done so much for me without any return.

To all people who gave me strenghth to continue.

Acknowledgements

First, I have to acknowledge my thanks to almighty Allah, for his help and bless. This work would not be completed without his guidance.

I wish to thank Mr. Redouane BOUDJEMAA, for his guidance, help and support throught this work.

I am also grateful to Mr. BOUKHARI Mohamed and Madam DJEMIA Noura for the honor of agreeing to preside over and judge this work.

Without forgetting to thank the TBA team for their professionalism.

Contents

List of Figures

List of Tables

ملخص

الهدف من هذا المشروع البحثي هو استخدام نموذج تحسين غير خطي يعتمد على البرمجة للتجديد النقدي لأجهزة الصراف الآلي التي:

1. تقلل من التكلفة الإجمالية للتجديد النقدي مع ضمان أن جميع أجهزة الصراف الآلي لديها مستويات نقدية كافية.
2. تأخذ بعين لإعتبار عوامل مثل أنماط استخدام أجهزة الصراف الآلي ، والطلب على النقد ، والنقد سعة التخزين وقيود الميزانية.
3. يمكن تنفيذها في شبكات أجهزة الصراف الآلي في العالم الحقيقي للحد من العمليات المكلفة وتحسين إدارة النقد.

تركز هذا العمل على إنشاء إطار مفاهيمي لتحسين تكلفة التجديد النقدي لشبكات أجهزة الصراف الآلي وتقليل حالات نقص الأموال فيها. الهدف هو تقليل التكلفة الإجمالية للخدمات اللوجستية والمخزون وتحسين خدمة العملاء و إرضائهم. يأخذ هذا النهج في الاعتبار الجمع بين التقنيات التالية:

- التنبؤ بالسلاسل الزمنية مع مراعاة الموسمية والإتجاهات والأحداث من التقويم المحلي (نظرية الجرد) تحديد تواتر ومبلغ النقد المراد تجديده لكل ماكينة صراف آلي.

Résume

L'objectif de ce travail de recherche est d'utiliser un modèle d'optimisation non linéaire basé sur la programmation pour le réapprovisionnement en espèces des guichets automatiques qui:

1. Minimise le coût total de réapprovisionnement en espèces tout en veillant à ce que tous Les guichets automatiques ont des niveaux de trésorerie adéquats.
2. Prend en compte des facteurs tels que les habitudes d'utilisation des guichets automatiques, la demande en espèces, les espèces capacité de stockage et contraintes budgétaires.
3. Peut être mis en œuvre dans des réseaux ATM du monde réel pour réduire les opérations coûts et améliorer la gestion de trésorerie.

Cette étude se concentre sur l'établissement d'un cadre conceptuel pour optimiser le coût de réapprovisionnement en espèces pour les réseaux de guichets automatiques et minimiser les cas de manque d'argent aux guichets automatiques individuels. L'objectif est de minimiser le coût total de la logistique, de l'inventaire et de l'amélioration de la satisfaction du service client. Cette approche considère la combinaison des techniques suivantes:
- Prévision des séries chronologiques tenant compte de la saisonnalité, des tendances et des événements du calendrier local-Théorie de l'inventaire
-détermination de la fréquence et de la quantité de liquidités à reconstituer pour chaque guichet automatique

Abstract

The objective of this work is to use a NonLinear programming based optimization model for ATM cash replenishment that:

1. Minimizes the total cost of cash replenishment while ensuring that all ATMs have adequate cash levels.
2. Takes into account factors like ATM usage patterns, cash demand, cash storage capacity, and budget constraints.
3. Can be implemented in real-world ATM networks to reduce operational costs and improve cash management.

This study focuses on establishing a conceptual frame work for optimizing the cost of cash replenishment for networks of ATMs and minimizing the cash-dry instances at individual ATM. The objective is to minimize the total cost of logistics, inventory, and improving customer service satisfaction.
This approach considers the combination of following techniques:
- Time series forecasting accounting for seasonality, trend and local calendar events - Inventory theory
– determining frequency and quantity of cash to be replenished for each ATM .

Introduction

Cash plays a pivotal role in everyday life and can be considered the backbone of the economy, at all times, government institutions must ensure a sufficient supply of banknotes and coins to the public.
Maintaining a minimum amount of cash in Automated Teller Machines (ATM) is often a controlled operating requirement for commercial banks, which are enforced by central banks to ensure that customer experience is consistently fulfilled. The number of cardholders is drastically increasing daily and with these rising numbers, cash withdrawal transactions have become predominantly relevant, especially in those countries which are in the process of being paperless.
Usually, banks tend to estimate the expected value without understanding the essence of cash inflows or outflows in a given ATM. The refill varies from ATM to ATM, depending on their location , peak times, Paydays, weekends, holidays, open hours and several other factors.
For example, an ATM located in an urban area will have a greater amount of transactional load than an ATM located in any rural area. This phenomenon can result in a stock out situation. One survey conducted in 2013 showed that 20% of people switch to other bank ATMs because of out of cash circumstances. Hence, there is a need for an efficient cash management solution that helps banks to anticipate the need for ATMs to replenish cash. The technology and the underlying algorithms should also be versatile enough to allow the bank to predict future demand and conduct what-if analysis to optimize the ATM supply network for cash distribution.
Another concern is that the topic of cash management is generally perceived and discussed only as a market management problem, not from the point of view of cost optimization . The key goal can therefore be set as an improvement of cash levels to reduce costs and ensure customer loyalty.

Implementing cash forecasting and optimization solutions helps the bank to take better advan-

tage of the funds available for investment .
In turn, introducing integrated cash optimization models would centralize the logistics phase of cash management for any ATM branch, and hence enable banks to reduce overall costs incurred in physical cash operations.
Consumer sales for each ATM must be estimated to assess the volume of cash that should be stored in these ATMs. However, modern forms of ATM make cash deposited by consumers eligible for dispensing until the money is validated, thereby, reducing the expense of cash re-supply and maintenance.
Although the problem of optimizing cash supply has many similarities with optimizing ATM cash, there are important variations that need to be considered when forecasting cash demand and transfers. For example, ATMs have a quota on banknotes, while branches are expected to keep as many banknotes as possible. The amount of money that can be withdrawn from an ATM is capped, but customers are entitled to withdraw money from branches below a predefined amount at any point in time. ATMs run at night, on weekends, and holidays, while branches are closed during certain days. These variations must be taken into account when developing both prediction and optimization models.
The main task of an ATM cash forecast model is to statistically link the volume withdrawn to the date, position, and additional variables. Although this is a classical continuous-value prediction problem over time, the above-mentioned auxiliary factors add substantial stochasticity and the prediction varies from one ATM to another. For banks, it is also difficult to build a statistical model for their entire ATM network.
A procedure is developed to statistically forecast customer demand which estimates the demand during replenishment order lead-times and replenishment cycles.
Nowadays ATM network and credit cards are the essential parts of modern lifestyle, consequently one of the most actual problem in the bank's ATM network is optimization of cash flow and organization of uninterrupted work. For the bank it is important to prevent the rush demand for cash withdrawals by customers, that can be provoked by the delayed loading of ATMs, and reduce the service expenses.

In this work we consider a problem in which a set of geographically dispersed ATMs with

known requirements must be served with a fleet of money collector teams stationed in the depot in such a way as to minimize some distribution objective.
This Serving the ATMs network is a costly task: it takes employees time to supervise the network and make decisions about cash management and it involves high operating costs (financial, transport, etc.) The increase of transportation and servicing cost can be substantial for banks.
Route optimization for the collector teams is allow to reduce bank expenses and to control the encashment process. Problem is combined with the problem of composition of service requests from the ATM network. We assume that the money collector teams are identical with the equal capacity and must start and finish their routes at the depot. At the present time more and more banks are turning their attention to have greater efficiency in how they manage their cash in ATMs.
In current section we consider a problem of cash flow forecasting in the ATM network, especially focused on forecasting of cash balance in ATMs and the moment of ATM upload.
Moreover, we find the moment of each ATM refusal and compile the requests for the bank's processing center. We analyzed cash flows of each ATM, using statistical data, and found that cash withdrawal is heterogeneous process which depends on following factors:

- Paydays, weekends, holidays, etc.
- ATM location.
- ATM open hours.

Chapter 1

Backround definition and notation

1.1 Optimisation models:

Optimization models are mathematical representations of real-world problems that aim to find the best possible solution among a set of feasible options. These models are used in various fields, Here are some common types of optimization models:

1.1.1 Linear programming

Also called **linear optimization**, is a method to achieve the best outcome (such as maximum profit or lowest cost) in a mathematical model whose requirements are represented by linear relationships. Linear programming is a special case of mathematical programming (also known as mathematical optimization).

More formally, linear programming is a technique for the optimization of a linear objective function, subject to linear equality and linear inequality constraints. Its feasible region is a convex polytope, which is a set defined as the intersection of finitely many half spaces, each of which is defined by a linear inequality. Its objective function is a real-valued affine (linear) function defined on this polyhedron. A linear programming algorithm finds a point in the polytope where this function has the smallest (or largest) value if such a point exists.

Linear programs are problems that can be expressed in canonical form as:

Find a vector x

That minimize $c^t x$

Subject to $Ax \leq b$

And $x \geq 0$

Here the components of x are the variables to be determined, c and b are given vectors (with ct indicating that the coefficients of c are used as a single-row matrix for the purpose of forming the matrix product), and A is a given matrix. The function whose value is to be maximized or minimized ($x \rightarrow c^t x$ in this case) is called the objective function. The inequalities $Ax \leq b$ and $x \geq 0$ are the constraints which specify a convex polytope over which the objective function is to be optimized. In this context, two vectors are comparable when they have the same dimensions. If every entry in the first is less-than or equal-to the corresponding entry in the second, then it can be said that the first vector is less-than or equal-to the second vector.

Linear programming can be applied to various fields of study. It is widely used in mathematics and, to a lesser extent, in business, economics, and some engineering problems. Industries that use linear programming models include transportation, energy, telecommunications, and manufacturing. It has proven useful in modeling diverse types of problems in planning, routing, scheduling, assignment, and design.

1.1.2 Integer linear programming:

An integer programming problem is a mathematical optimization or feasibility program in which some or all of the variables are restricted to be integers. In many settings the term refers to integer linear programming (ILP), in which the objective function and the constraints (other than the integer constraints) are linear.

Integer programming is NP-complete$^{[citation\ needed]}$. In particular, the special case of 0-1 integer linear programming, in which unknowns are binary, and only the restrictions must be satisfied, is one of Karp's 21 NP-complete problems.

If some decision variables are not discrete, the problem is known as a mixed-integer programming problem.

Canonical and standard form for ILP:

In integer linear programming, the *canonical form* is distinct from the *standard form*. An integer linear program in canonical form is expressed thus (note that it is the x vector which is to be decided):

Maximize $c^t x$

Subject to $Ax + s = b$,

$x \geq 0$

And $\quad x \in Z^n$

and an ILP in standard form expressed as:

Maximize $c^t x$

Subject to $Ax + s = b$,

$s \geq 0$

$x \geq 0$

And $\quad x \in Z^n$

Where $c \in R^n$ are vectors and $A \in R^{m*n}$ is a matrix. As with linear programs, ILPs not in standard form can be converted to standard form by eliminating inequalities, introducing slack variables (s) and replacing variables that are not sign-constrained with the difference of two sign-constrained variables.

1.1.3 NONLINEAR PROGRAMMING:

In mathematics, **nonlinear programming (NLP)** is the process of solving an optimization problem where some of the constraints or the objective function are nonlinear. An optimization problem is one of calculation of the extrema (maxima, minima or stationary points) of an objective function over a set of unknown real variables and conditional to the satisfaction of a system of equalities and inequalities, collectively termed constraints. It is the sub-field of mathematical optimization that deals with problems that are not linear.

Let n, m, and p be positive integers. Let X be a subset of R^n, let f, g_i, and h_j be real-valued functions on X for each i in $\{1, ..., m\}$ and each j in $\{1, ..., p\}$, with at least one of f, g_i, and h_j being nonlinear.

A nonlinear minimization problem is an optimization problem of the form:

Minimize $f(x)$

Subject to $g_i(x) \leq 0 \quad \forall i \in \{1, ..., m\}$

$H_j(x) = \ \ for\ each\ \ j \in 1, ..., p$

$x \in X.$

A nonlinear maximization problem is defined in a similar way.

1.2 Decomposition Methods:

We introduce two main categories of methods that generate boundaries by iterative construction of polyhedral approximations of the convex hull of some MILP's feasible solutions. The first category, called conventional methods, considers the intersection of a polyhedron with a compact description and a polyhedron generated implicitly by solving an auxiliary problem. This happens because the size of the description of the second polyhedron is exponential, so it cannot be defined explicitly and efficiently. Traditional methods are further divided into outer methods, such as the cut plane methodTapez une équation ici., and inner methods, such as the Dantzig-Wolfe method and the Lagrangian method. The second category, called built-in methods, can exponentially increase the size of both polyhedra. This category includes algorithms that can simultaneously integrate both internal and external methods.

1.3 Traditional Decomposition Methods :

1.3.1 Cutting-Plane Method:

Using the cutting-plane method, the bound $Z_D = min_{x \in P' \cap Q''}\{c^T x\}$ can be obtained dynamically by generating the relevant portions of an outer description of $P^{'}$. Let $[D, d]$ denote the set of facet defining inequalities of $P^{'}$:

$$P^{'} = \{x \in R^n | Dx \geq d\}.$$

Then the cutting-plane formulation for the problem of calculating z_D can be written as

$$Z_{CP} = \min\{c^T x | Dx \geq d\} \qquad x \in Q^{''}$$

This is a linear program, but since the set $[D, d]$ of valid inequalities is potentially of exponential size, we dynamically generate them by solving a separation problem. An outline of the method is presented in Figure 2.3.1;2.

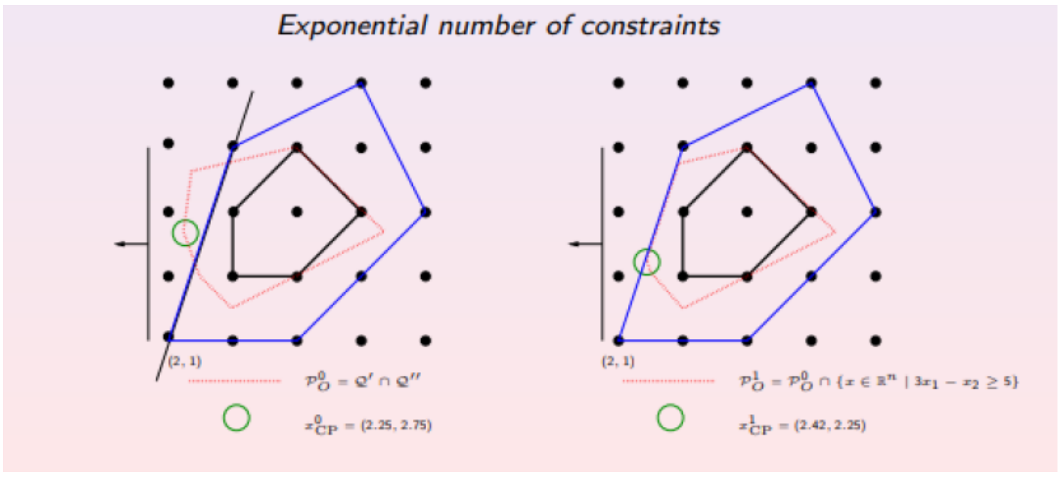

Exponential number of constraints
(2, 1)
$\mathcal{P}_O^0 = \mathcal{Q}' \cap \mathcal{Q}''$
$x_{CP}^0 = (2.25, 2.75)$
(2, 1)
$\mathcal{P}_O^1 = \mathcal{P}_O^0 \cap \{x \in \mathbb{R}^n \mid 3x_1 - x_2 \geq 5\}$
$x_{CP}^1 = (2.42, 2.25)$

Figure 1.1: cutting plane method

In Step 1, we need to initialize the set of valid inequalities to obtain the first approximation. Typically, if Q is compact, this is done by using the initial set of inequalities $[A, b]$. If Q is not compact, then we start with $[D^{'}, d^{'}] = [A^{''}, b^{''}]$ and define the initial outer approximation $P_O^{'} = Q^{''}$. In Step 2, the master problem is a linear program whose feasible region is the current outer approximation P_O^t.

Step 3. Solving the master problem in iteration t, we generate the relaxed (primal) solution x_{CP}^t and a valid lower bound. In the figure, the initial set of inequalities is taken to be those of $Q00$, since it is assumed that the facet-defining inequalities for $P^{'}$, which dominate those of $Q^{'}$, can be generated dynamically. In practice, however, this initial set may be chosen to include those of $Q^{'}$ or some other polyhedron, on an empirical basis.

1.3.2 Dantzig-wolfe Method:

The Dantzig-Wolfe method is an algorithm used for solving large-scale linear programming problems by exploiting the problem's block structure. It was proposed independently by George Dantzig and Philip Wolfe in the 1960s. The primary idea behind this method is to decompose a large linear programming problem into smaller subproblems and then iteratively solve them to obtain the optimal solution for the original problem.

In the Dantzig-Wolfe method, the bound z_D can be obtained by dynamically generating the relevant portions of an inner description of $P^{'}$ and intersecting it with $Q^{''}$. Consider Minkowski's

Cutting - Plane Method

Input: An instance OPT $(\mathcal{P}, c)$

Output: A lower bound z_{CP} on the optimal solution value for the instance, and $\hat{x}_{CP} \in \mathbb{R}^n$ such that $z_{CP} = c^T \hat{x}_{CP}$

1. **Initialize:** Construct an initial outer approximation

$$\mathcal{P}_O^0 = \{x \in \mathbb{R}^n | D^0 x \geq d^0\} \supseteq \mathcal{P} \tag{1.1}$$

   where $D^0 = A''$ and $d^0 = b''$ and set $t \leftarrow 0$.

2. **Master Problem:** Solve the linear program

$$z_{CP}^t = \min_{x \in \mathbb{R}^n} \left\{c^T x | D^t x \geq d^t\right\} \tag{1.2}$$

   to obtain the optimal value $z_{CP}^t = \min_{x \in \mathcal{P}_O^t} \left\{c^T x\right\} \leq z_{IP}$ and optimal primal solution x_{CP}^t

3. **Subproblem:** Call the subroutine SEP $(\mathcal{P}, x_{CP}^t)$ to generate a set $[\tilde{D}, \tilde{d}]$ of potentially improving valid inequalities for $\mathcal{P}$, violated by x_{CP}^t.

4. **Update:** If violated inequalities were found in Step 3 , set $[D^{t+1}, d^{t+1}] \leftarrow \begin{bmatrix} D^t & d^t \\ \tilde{D} & \tilde{d} \end{bmatrix}$ to from a new outer approximation

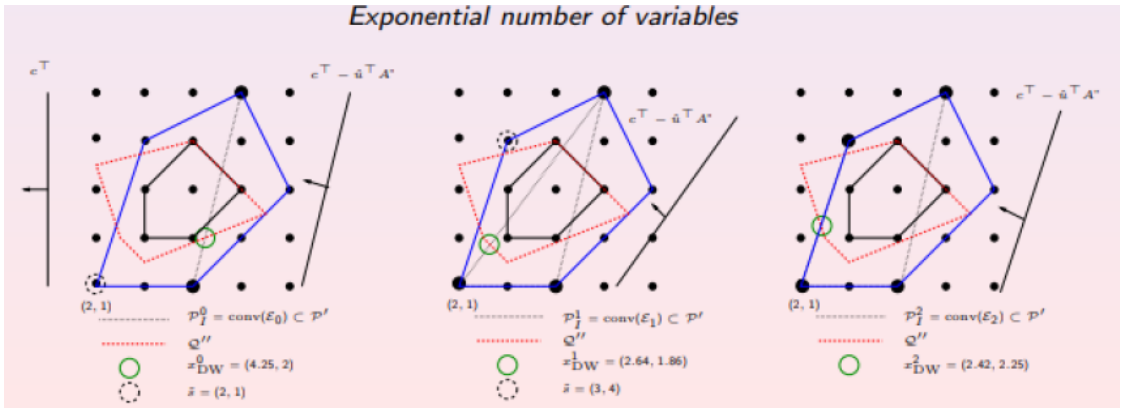
$$\mathcal{P}_O^{t+1} = \{x \in \mathbb{R}^n | D^{t+1} x \leq d^{t+1}\} \supseteq \mathcal{P} \tag{1.3}$$

   and set $t \leftarrow t + 1$. Go to step 2

5. If no violated inequalities were found, output $z_{CP} = z_{CP}^t \leq z_{IP}$ and $\hat{x}_{CP} = x_{CP}^t$

Figure 1.2: Outline of the cutting-plane method

Representation Theorem, which states that every bounded polyhedron is finitely generated by its extreme points and extreme rays. Since we assume the feasible region for the problem of interest is bounded.



Exponential number of variables
c^T
$c^T - \hat{u}^T A''$
(2, 1)
$\mathcal{P}_I^0 = \text{conv}(\mathcal{E}_0) \subset \mathcal{P}'$
$\mathcal{Q}''$
$x_{DW}^0 = (4.25, 2)$
$\tilde{s} = (2, 1)$
$c^T - \hat{u}^T A''$
(2, 1)
$\mathcal{P}_I^1 = \text{conv}(\mathcal{E}_1) \subset \mathcal{P}'$
$\mathcal{Q}''$
$x_{DW}^1 = (2.64, 1.86)$
$\tilde{s} = (3, 4)$
$c^T - \hat{u}^T A''$
(2, 1)
$\mathcal{P}_I^2 = \text{conv}(\mathcal{E}_2) \subset \mathcal{P}'$
$\mathcal{Q}''$
$x_{DW}^2 = (2.42, 2.25)$

Figure 1.3: Dantzing-wolf method

1.3.3 Lagrangian Method:

The Lagrangian method is a general approach for computing z_D that is closely related to the Dantzig-Wolfe method but is focused primarily on producing dual solution information. The Lagrangian method can be viewed as a method for producing a particular face of P', as in the Dantzig-Wolfe method, but no explicit approximation of P' is maintained. Although there are implementations of the Lagrangian method that do produce approximate primal solution information similar to the solution information that the Dantzig-Wolfe method produces our viewpoint is that the main difference between the Dantzig-Wolfe method and the Lagrangian method is the type of solution information they produce. This distinction is important when we discuss integrated methods. When exact primal solution information is not required, faster algorithms for determining the dual solution are possible. By employing a Lagrangian framework instead of a Dantzig-Wolfe framework, we can take advantage of this fact.

$$Z_{LD} = \max\{\alpha + b^{''T} u...^{''} | (c^T - u^T A^{'''}) s - \alpha \geq 0 \quad \forall s \in \xi\} = z_{DW}$$

Danzig -Wolfe Method

Input: An instance OPT $(\mathcal{P}, c)$

Output: A lower bound z_{DW} on the optimal solution value for the instance,a primal solution $\hat{\lambda} \in \mathbb{R}^{\xi}$, and a dual solution $(\hat{u}_{DW}, \hat{\alpha}_{DW}) \in \mathbb{R}^{m^n+1}$.

1. **Initialize:** Construct an initial outer approximation

$$\mathcal{P}_I^0 = \left\{ \sum_{s\in\xi^0} s\lambda_s \middle| \sum_{s\in\xi^0} \lambda_s = 1, \lambda_s \geq 0 \ \forall s \in \xi^0, \lambda_s = 0, \forall s \in \xi \setminus \xi^0 \right\} \subseteq \mathcal{P}' \tag{1.4}$$

from an initial set ξ^0 of extreme points of $\mathcal{P}'$ and set $t \leftarrow 0$.

2. **Master Problem:** Solve the Danzig -Wolfe reformulation

$$\bar{z}_{DW}^t = \min_{\lambda\in\mathbb{R}_+^{\xi}} \left\{ c^T \left(\sum_{s\in\xi} s\lambda_s \right) \middle| A'' \left(\sum_{s\in\xi} s\lambda_s \right) \geq b'', \sum_{s\in\xi} \lambda_s = 1, \lambda_s = 0, \ \forall s \in \xi \setminus \xi^t \right\} \tag{1.5}$$

to obtain the optimal value $\bar{z}_{DW}^t = \min_{\mathcal{P}_I^t \cap Q''} c^T x \geq z_{DW}$, an optimal primal solution $\lambda_{DW}^t \in \mathbb{R}_+^{\xi}$ and optimal dual solution $(u_{DW}^t, \alpha_{DW}^t) \in \mathbb{R}^{m''+1}$

3. **Subproblem:** Call the subroutine SPT $\left(\mathcal{P}', c^T - (u_{DW}^t)^T A'', \alpha_{DW}^t\right)$, generating a set $\bar{\xi}$ of improving membres of ξ with negative reduced cost, where the reduced cost of $s \in \xi$ is

$$rc(s) = \left(c^T - (u_{DW}^t)^T A'' \right) s - \alpha_{DW}^t \tag{1.6}$$

If $\bar{s} \in \bar{\xi}$ is a member of ξ with smallest reduced cost, then $\underline{z}_{DW}^t = rc(\bar{s}) + \alpha_{DW}^t + (u_{DW}^t)^T b'' \leq z_{DW}$ provides a valid lower bound.

4. **Update:** If $\bar{\xi} \neq \emptyset$, set $\xi^{t+1} \leftarrow \xi^t \cup \bar{\xi}$ to from the new inner approximation

$$\mathcal{P}_I^{t+1} = \left\{ \sum_{s\in\xi^{t+1}} s\lambda_s \middle| \sum_{s\in\xi^{t+1}} \lambda_s = 1, \lambda_s \geq 0 \ \forall s \in \xi^{t+1}, \lambda_s = 0, \forall s \in \xi \setminus \xi^{t+1} \right\} \subseteq \mathcal{P}' \tag{1.7}$$

and set $t \leftarrow t + 1$. Go to step 2

5. If $\bar{\xi} = \emptyset$, output the bound $z_{DW} = \bar{z}_{DW}^t = \underline{z}_{DW}^t, \hat{\lambda}_{DW} = \lambda_{DW}^t$ and $(\hat{u}_{DW}, \hat{\alpha}_{DW}) = (u_{DW}^t, \alpha_{DW}^t)$

Figure 1.4: Outline of the Dantzig-Wolfe method

Lagrangian method

Input: An instance OPT $(\mathcal{P}, c)$

Output: A lower bound z_{LD} on the optimal solution value for the instance, and a dual solution $\hat{u}_{LD} \in \mathbb{R}^{m''}$

1. **Initialize:** Let $s^0_{LD} \in \xi$ define some initial extreme point of $\mathcal{P}'$, u^0_{LD} some initial setting for the dual multipliers and set $t \leftarrow 0$.

2. **Master Problem:** Using the solution information gained from solving the pricing subproblem, and the previous dual setting u^t_{LD}. revise the dual multipliers u^{t+1}_{LD}.

3. **Subproblem:** Call the subroutine SPT $\left(\mathcal{P}', c^T - (u^t_{LD})^T A''\right)$, to solve

$$z^t_{LD} = \min_{s \in \xi} \left\{ \left(c^T - (u^t_{LD})^T A'' \right) s + b''^T u^t_{DW} \right\} \tag{1.8}$$

   Let s^{t+1}_{LD} be the optimal solution to this subproblem, if one is found.

4. If a prespecified stopping criterion is met , then output $z_{LD} = z^t_{LD}$ and $\hat{u}_{LD} = u^t_{LD}$ otherwise, go to step 2.

Figure 1.5: Outline of the Lagrangian method

1.4 Integrated Decomposition Methods:

In precedent section, we demonstrated that traditional decomposition approaches can be viewed as utilizing dynamically generated polyhedral information to improve the LP bound by building either an inner or an outer approximation of an implicitly defined polyhedron that approximates P. The choice between inner and outer methods is largely an empirical one, but recent computational research has favored outer methods. In what follows, we discuss two methods for integrating inner and outer methods. over those achieved by either approach alone. While traditional decomposition approaches build either an inner or an outer approximation, integrated decomposition methods build both an inner and an outer approximation. These methods follow the same basic logic as traditional decomposition methods, except that the master problem is required to generate both primal and dual solution information, and the subproblem can be either a separation problem or an optimization problem. The first two techniques we describe integrate the cutting-plane method with either the Dantzig-Wolfe method or the Lagrangian method.

1.4.1 Price-and-Cut:

The integration of the cutting-plane method with the Dantzig-Wolfe method results in a procedure that alternates between a subproblem that attempts to generate improving columns (the pricing subproblem) and a subproblem that attempts to generate improving valid inequalities (the cutting subproblem). Hence, we call the resulting method price-and-cut. When employed in a branch and bound framework, the overall technique is called branch-and-priceand-cut. This method has already been studied previously by a number of authors and more recently by Aragao and Uchoa . As in the Dantzig-Wolfe method, the bound produced by price-and-cut can be thought of as resulting from the intersection of two approximating polyhedra. However, the Dantzig-Wolfe method required one of these, $Q^{''}$, to have a short description. With integrated methods, both polyhedra can have descriptions of exponential size. Hence, price-and-cut allows partial descriptions of both an inner polyhedron PI and an outer polyhedron PO to be generated dynamically. To optimize over the intersection of PI and PO, except that the $[A^{''}, b^{''}]$ is replaced by a matrix that changes dynamically. The outline of this method is shown in Figures 2.4.1.

Price-and-cut method

Input: An instance OPT $(\mathcal{P}, c)$

Output: A lower bound z_{PC} on the optimal solution value for the instance,a primal solution $\hat{x}_{PC} \in \mathbb{R}^n$, an optimal decomposition $\hat{\lambda}_{PC} \in \mathbb{R}^{\xi}$, and a dual solution $(\hat{u}_{PC}, \hat{\alpha}_{PC}) \in \mathbb{R}^{m^t+1}$, and the inequalities $[D_{PC}, d_{PC}] \in \mathbb{R}^{m^t \times (n+1)}$.

1. **Initialize:** Construct an initial inner approximation

$$\mathcal{P}_I^0 = \left\{ \sum_{s\in\xi^0} s\lambda_s \middle| \sum_{s\in\xi^0} \lambda_s = 1, \lambda_s \geq 0 \ \forall s \in \xi^0, \lambda_s = 0 \forall s \in \xi \setminus \xi^0 \right\} \subseteq \mathcal{P}' \tag{1.9}$$

from an initial set ξ^0 of extreme points of $\mathcal{P}'$ and an initial outer approximation

$$\mathcal{P}_O^0 = \{x \in \mathbb{R}^n | D^0 x \geq d^0\} \supseteq \mathcal{P} \tag{1.10}$$

where $D^0 = A''$ and $d^0 = b''$ and set $t \leftarrow 0$, $m^0 \leftarrow m''$.

2. **Master Problem:** Solve the Danzig -Wolfe reformulation

$$\bar{z}_{PC}^t = \min_{\lambda \in \mathbb{R}_+^{\xi}} \left\{ c^T \left(\sum_{s\in\xi} s\lambda_s \right) \middle| D^t \left(\sum_{s\in\xi} s\lambda_s \right) \geq d^t, \sum_{s\in\xi} \lambda_s = 1, \lambda_s = 0, \ \forall s \in \xi \setminus \xi^t \right\} \tag{1.11}$$

of the LP over the polyhedron $\mathcal{P}_I^t \cap \mathcal{P}_O^t$ to obtain the optimal value $\bar{z}_{PC}^t$, an optimal primal solution $\lambda_{PC}^t \in \mathbb{R}^{\xi}$, an optimal fractional solution $x_{PC}^t = \sum_{s\in\xi} s(\lambda_{PC}^t)_s$, and optimal dual solution $(u_{PC}^t, \alpha_{PC}^t) \in \mathbb{R}^{m^t+1}$.

Figure 1.6: Outline of the price-and-cut method

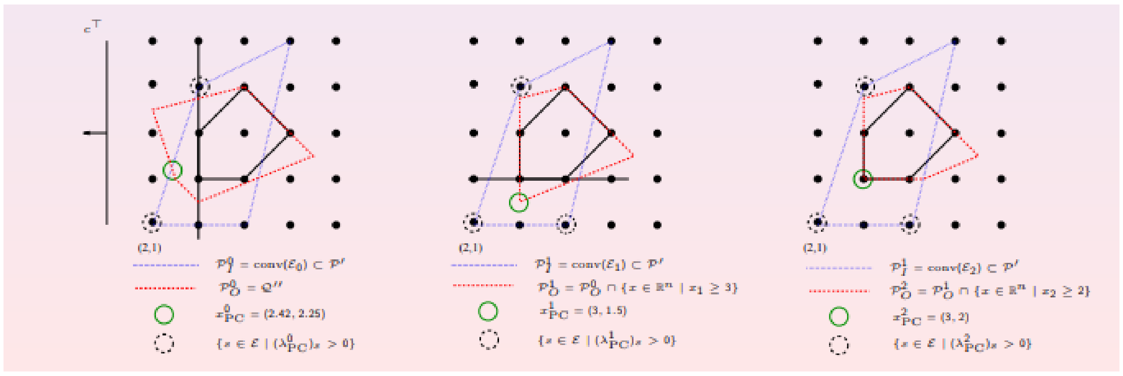
3. Do either (a) or (b).

a Pricing Subproblem and Update: Call the subroutine SPT $\left(\mathcal{P}', c^T - (u^t_{PC})^T D^t, \alpha^t_{PC}\right)$ generating a set $\tilde{\xi}$ of improving membres of ξ with negative reduced cost (defined in Figure 2.4), If $\hat{\xi} \neq \emptyset$, set $\xi^{t+1} \leftarrow \xi^t \cup \hat{\xi}$ to from a new inner approximation $\mathcal{P}^{t+1}_I$. If $\tilde{s} \in \xi$ is a member of ξ with smallest reduced cost, then $\underline{z}^t_{PC} = rc(\tilde{s}) + \alpha^t_{PC} + (d^t)^T u^t_{PC}$ provides a valid lower bound. Set $[D^{t+1}, d^{t+1}] \leftarrow [D^t, d^t]$, $\mathcal{P}^{t+1}_O \leftarrow \mathcal{P}^t_O$, $m^{t+1} \leftarrow m^t$, $t \leftarrow t+1$, and go to step 2.

b Cutting Subproblem and Update: Call the subroutine SPT $\left(\mathcal{P}', x^t_{PC}\right)$ to generate a set of improving valid inequalities $[\tilde{D}, \tilde{d}] \in \mathbb{R}^{\bar{m} \times n+1}$ for $\mathcal{P}$, violated by $\hat{x}_{PC}$. If violated inequalities were found, set $[D^{t+1}, d^{t+1}] \leftarrow \begin{bmatrix} D^t & d^t \\ \tilde{D} & \tilde{d} \end{bmatrix}$ to from a new inner approximation $\mathcal{P}^{t+1}_O$. Set $m^{t+1} \leftarrow m^t + \tilde{m}$, $\xi^{t+1} \leftarrow \xi^t$,$\mathcal{P}^{t+1}_I \leftarrow \mathcal{P}^t_I$, $t \leftarrow t+1$, and go to step 2

4. If $\bar{\xi} = \emptyset$ and no valid inequalities werz found, output the bound $z_{PC} = \bar{z}^t_{PC} = \underline{z}^t_{PC} = c^T x^t_{PC}$, $\hat{x}_{PC} = x^t_{PC}$,$\bar{\lambda}_{PC} = \lambda^t_{PC}$, $(\hat{u}_{PC}, \hat{\alpha}_{PC}) = (u^t_{PC}, \alpha^t_{PC})$, and $[D_{PC}, d_{PC}] = [D^t, d^t]$.

Figure 1.7: Outline of the Dantzig-Wolfe method



c^T
(2,1)
$\mathcal{P}^0_I = \text{conv}(\mathcal{E}_0) \subset \mathcal{P}'$
$\mathcal{P}^0_O = \mathcal{Q}''$
$x^0_{PC} = (2.42, 2.25)$
$\{s \in \mathcal{E} \mid (\lambda^0_{PC})_s > 0\}$
(2,1)
$\mathcal{P}^1_I = \text{conv}(\mathcal{E}_1) \subset \mathcal{P}'$
$\mathcal{P}^1_O = \mathcal{P}^0_O \cap \{x \in \mathbb{R}^n \mid x_1 \geq 3\}$
$x^1_{PC} = (3, 1.5)$
$\{s \in \mathcal{E} \mid (\lambda^1_{PC})_s > 0\}$
(2,1)
$\mathcal{P}^1_I = \text{conv}(\mathcal{E}_2) \subset \mathcal{P}'$
$\mathcal{P}^2_O = \mathcal{P}^1_O \cap \{x \in \mathbb{R}^n \mid x_2 \geq 2\}$
$x^2_{PC} = (3, 2)$
$\{s \in \mathcal{E} \mid (\lambda^2_{PC})_s > 0\}$

Figure 1.8: Price and cut method

1.4.2 Relax-and-Cut:

Just as with the Dantzig-Wolfe method, the Lagrangian method of Figure 2.8 can be integrated with the cutting-plane method to yield a procedure several authors have recently termed relaxand-cut [26, 62, 56]. This is done in much the same fashion as in price-and-cut, with a choice in each iteration between solving a pricing subproblem and a cutting subproblem. In each iteration that the cutting subproblem is solved, the generated valid inequalities are added to the description of the outer polyhedron, which is explicitly maintained as the algorithm proceeds. As with the traditional Lagrangian method, no explicit inner polyhedron is maintained, but the algorithm can again be seen as one that computes a face of the implicitly defined inner polyhedron that contains the optimal face of solutions to a linear program solved over the intersection of the two polyhedra. When employed within a branch-and-bound framework, we call the overall method branch-and-relax-and-cut.

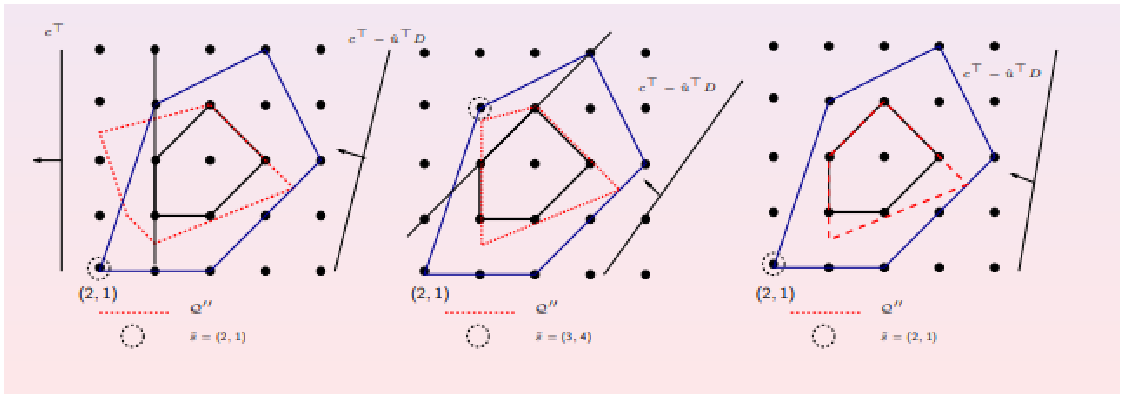


Figure 1.9: Relax and cut method

1.4.3 Decompose-and-Cut:

the use of decomposition in price-and-cut separation and decompose-and-cut, an algorithm related to integrated decomposition methods. It reviews structured separation and decomposeand-cut, a separation algorithm that derives decomposition cuts, a class of cutting planes.the use of decomposition in price-and-cut separation and decompose-and-cut, an algorithm related to integrated decomposition methods. It reviews structured separation and decomposeand-cut, a separation algorithm that derives decomposition cuts, a class of cutting planes.

Relax-and-cut method

Input: An instance OPT $(\mathcal{P}, c)$

Output: A lower bound z_{RC} on the optimal solution value for the instance and a dual solution $\hat{u}_{RC} \in \mathbb{R}^{m^t}$.

1. **Initialize:**Let s^0_{RC} define some initial extreme poit of $\mathcal{P}'$ and construct an initial outer approximation

$$\mathcal{P}^0_O = \{x \in \mathbb{R}^n | D^0 x \geq d^0\} \supseteq \mathcal{P} \tag{1.12}$$

   where $D^0 = A''$ and $d^0 = b''$.Let $u^0_{RC} \in \mathbb{R}^{m''}$ be some initial set of dual multipliers associated with the constraints $[D^0, d^0]$. Set $t \leftarrow 0$ and $m^t = m''$.

2. **Master Problem:** Using the solution information gained from solving the pricing subproblem, and the previous dual solution u^t_{RC}, update the dual solution (if the pricing problem was just solved) or initialize the new dual multipliers (if the cutting subproblem was just solved) to obtain $u^{t+1}_{RC} \in \mathbb{R}^{m^t}$.

3. Do either (a) or (b).

   (a) Pricing Subproblem: Call the subroutine SPT $\left(\mathcal{P}', c - (u^t_{RC})^T D^t\right)$ to obtain

$$z^t_{RC} = \min_{s \in \xi} \left\{ \left(c^T - (u^t_{RC}) D^t + d^t (u^t_{RC}) \right) \right\} \tag{1.13}$$

   Let $s^{t+1}_{RC} \in \xi$ be the optimal solution to this subproblem. Set $[D^{t+1}, d^{t+1}] \leftarrow [D^t, d^t]$, $\mathcal{P}^{t+1}_O \leftarrow \mathcal{P}^t_O$, $m^{t+1} \leftarrow m^t$, $t \leftarrow t+1$, and go to step 2.

   (b) Cutting Subproblem: Call the subroutine SPT $\left(\mathcal{P}', s^t_{RC}\right)$ to generate a set of improving valid inequalities $[\tilde{D}, \tilde{d}] \in \mathbb{R}^{\tilde{m} \times n+1}$ for $\mathcal{P}$, violated by s^t_{RC}. If violated inequalities were found, set $[D^{t+1}, d^{t+1}] \leftarrow \begin{bmatrix} D^t & d^t \\ \tilde{D} & \tilde{d} \end{bmatrix}$ to from a new outer approximation $\mathcal{P}^{t+1}_O$. Set $m^{t+1} \leftarrow m^t + \tilde{m}$, $s^{t+1}_{RC} \leftarrow s^t_{RC}$, $t \leftarrow t+1$, and go to step 2

4. If a pre-specified stopping criterion is met, then output $z_{RC} = z^t_{RC}$ and $\hat{u}_{RC} = u^t_{RC}$.

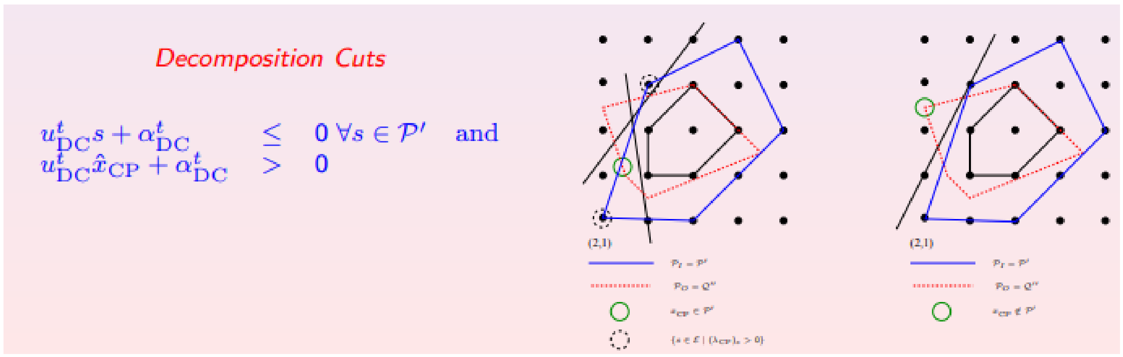
5. otherwise go to step 2

Figure 1.10: Outline of the relax-and-cut method

A template class with a known structure is used for identity verification procedures. Each class is individually injured, and a template class of valid inequalities in P is a set of related inequalities that describe the polyhedron containing P. Two known classes of valid inequalities in TSP are subtour elimination constraints and Comb inequality. Class C separation problems of valid inequalities for P are defined as facets. A valid inequality depends on the form of the inequality, and the worst-case execution time is independent of the isolation problem.

Inequality separation problems for certain classes can be easier, as they are solved as general facet identification problems for P. The convex hull facet identification problem of $TSP's$ feasible solutions is generally an NP-complete problem, with the intersection point being the polyhedron associated with it. The class of inequalities that can meaningfully solve the separation problem is not equal to P, and applying decoupled routines to increasingly difficult classes of inequalities often assumes a more difficult class of routines.

$$Min\{x^+ + x^- | \sum_{s\in\xi} S\lambda_s + x^+ - x^- =^\wedge x_{CP}, \sum_{s\in\xi} \lambda_s = 1\} \quad \lambda \in R_+^\xi, (x^+, x^-) \in R_+^n$$


Decomposition Cuts

$$\begin{array}{lll} u_{DC}^t s + \alpha_{DC}^t & \leq & 0 \ \forall s \in \mathcal{P}' \quad \text{and} \\ u_{DC}^t \hat{x}_{CP} + \alpha_{DC}^t & > & 0 \end{array}$$

(2,1)
$\mathcal{P}_I = \mathcal{P}'$
$\mathcal{P}_O = \mathcal{Q}''$
$\hat{x}_{CP} \in \mathcal{P}'$
$\{s \in \mathcal{E} \mid (\lambda_{CP})_s > 0\}$
(2,1)
$\mathcal{P}_I = \mathcal{P}'$
$\mathcal{P}_O = \mathcal{Q}''$
$\hat{x}_{CP} \notin \mathcal{P}'$

Figure 1.11: Decompose and cut method

1.4.4 Heuristic and metaheuristique:

Heuristic Methods:

Heuristic methods involve using intuitive approaches and rules of thumb to find approximate solutions to problems.

To apply a heuristic, you need to design a set of rules or algorithms that guide the search process

towards potentially good solutions.
Heuristics may not guarantee an optimal solution but can be useful for finding reasonably good solutions quickly.
Implementation of heuristic methods involves defining the heuristics, applying them to the problem, and refining the solution based on feedback.
Metaheuristic Methods:
Metaheuristic methods guide problem-solving by combining search techniques like Genetic Algorithms, Particle Swarm Optimization, Simulated Annealing, Ant Colony Optimization, and Tabu Search. They involve designing an algorithmic framework, defining operators, and setting parameters for exploration and exploitation.The effectiveness of these methods depends on the specific problem, the formulation of the objective function, the quality of the heuristics or operators used, and the tuning of algorithm parameters. Heuristics and metaheuristics are widely used in combinatorial optimization, global optimization, and other complex problems where finding exact solutions is challenging or computationally infeasible.
To apply heuristic or metaheuristic methods to a specific problem, you would typically write code in a programming language of your choice, define the problem's objective function and constraints, and implement the heuristic or metaheuristic algorithm. After running the algorithm for a certain number of iterations or until a termination criterion is met, you would analyze the results to see if the solutions obtained are satisfactory. If not, you might need to adjust the algorithm parameters or experiment with different heuristics or metaheuristics to improve the result.

Chapter 2

Definition of ATM cash replenishment Problem

2.1 Problem definition:

The increased cost of cash, the increased seriousness of security incidents, and the increased power of the customers drive the need for efficient, secure, and customer-oriented cash logistics. The affected institutions are required to anticipate the changing payment industry. Not all countries and cash supply chains across the globe experience the same gravity of this payment shift (yet), but in countries where the shift is paramount, a serious decline in cash usage is observed over the past decade. This shift, amplified by the financial crises that struck the world in 2008, has been the underlying Before 2008, commercial banks were not fully aware of the cost impact of cash and were not sufficiently interested in managing it efficiently. More attention was devoted to, for instance, developing and commercializing financial services (e.g., mortgages and loans). The lack of priority of the incurred cost by cash logistics is remarkable since the total cost of cash is primarily carried by these institutions. Before 2008, managing physical activities regarding cash was not considered to be a core business, but rather a burden that needed to be taken care of. The aforementioned shift in cash payments and the financial crises forced commercial banks to rethink their business processes. More and more institutions realize that cash has indeed had a pivotal impact on their statement of earnings and feel the need to improve the efficiency of cash logistics.

And who says bank, treasury, also says **Automate Teller Machines** the most important machine that makes life easier for people today easier on the banking service deposit, transfer and withdraw money. Cash management for ATMs is one of the important problems a bank faces. The costs of operating and maintaining ATMs are very high and include the cost of the ATM machines, renting facilities, networking, administration and the opportunity cost for depositing money in the machine. An ATM transaction is an expensive financial transaction and the banks can make little profit from it, but they must still support this service for customers. In this case, since the administrative cost and the opportunity cost for storing money in the machine are the crucial management costs that banks can plan and manage, we focus on the development of an efficient ATM cash inventory policy that satisfies customer demand. Inventory costs occur when the bank stores money in the ATMs and the cash centers, which are places for storage of cash moving to or from the central bank through commercial banks and cash centers to the bank branches and self-service devices (ATMs) to satisfy customer cash demand. The amount of money to place in ATMs and cash centers depends on future unknown demands. If the amount of money in ATMs and cash centers is higher than the customer demand, then an opportunity cost of holding cash will occur. The cost of money or the opportunity cost will be associated with interest rates and may also include insurance costs for the money held in the ATM. But if the amount of money is lower than the customer demand, the bank incurs a shortage cost.

when the ATM runs out of cash and is unable to meet customer demand. These costs can include both direct and indirect expenses. Here are a few examples: Transaction Reversals: When an ATM runs out of cash, customers may attempt to withdraw money, but the transaction fails. In such cases, the bank may need to reverse those failed transactions, which can result in administrative costs.

Customer Dissatisfaction: Cash shortages can lead to customer dissatisfaction and frustration. If customers are unable to withdraw money when needed, it may damage the reputation of the bank and result in customer attrition.

Customer Service and Support: When a customer encounters an out-of-cash ATM, they may seek assistance from the bank's customer service representatives. Handling customer complaints and inquiries related to ATM cash shortages requires additional resources and may incur costs.

Operational Efficiency: Cash management and replenishment processes become more complex when an ATM experiences frequent shortages. Additional efforts and costs are required to monitor ATM cash levels, schedule cash deliveries, and optimize the cash replenishment process to avoid shortages.
Lost Transaction Revenue: When an ATM is unable to dispense cash due to a shortage, the bank may lose potential transaction revenue. This can be especially significant during peak periods or in high-traffic locations.
Thus, the development of an advanced algorithm for ATM services to reduce the overall costs of running the ATMs is very important for banks.
The banks must pay an interchange fee when its customers use another bank's ATM. The bank may also have to pay a refill cost associated with replenishing the ATM. If the ATM is inside a bank branch, then there is no cost of moving cash to and from the ATM. However, if the ATM is not in a bank branch, then a refilling cost or ordering cost is incurred when the money in the ATM is refilled. This cost is independent of the amount of money refilled.
The goal of this research is to determine how much money to store in ATMs and cash centers and the frequency of cash replenishment in each period based on banks so that all demands are satisfied with minimum total costs for running the ATMs. Customer demand at cash ATM is assumed to be known and deterministic and shortages are not allowed.
Note that, since there is no cost for restocking cash at the ATMs in local branches they are considered part of cash management in branches and are not considered for this problem.
Through cash management optimization banks can avoid falling into the trap of maintaining too much cash and begin to profit by mobilizing idle cash. Effective cash management and control starts with accurate prediction of ATMs refusal, allowing banks to forecast cash demand for the network, and find an optimal routes, which manage to reduce servicing costs. The increase of transportation and servicing cost can be substantial for banks. Route optimization for the collector teams is allow to reduce bank expenses and to control the encashment process (Simutis et al., 2007).
In this work we consider a problem in which a set of geographically dispersed ATMs with known

requirements must be served with a fleet of money collector teams stationed in the depot in such a way as to minimize some distribution objective.

This problem is combined with the problem of composition of service requests from the ATM network. We assume that the money collector teams are identical with the equal capacity and must start and finish their routes at the depot.

Cash demand in ATMs require accurate prediction which is no different than in other vending machines. The only difference is the product which is cash needs to be replenished for a priory set period of time. If the forecast is wrong, it induces a considerable amount of costs. In the case of high forecast and high unused cash stored in the ATM incur costs to the bank. The bank pays different re-filling costs depending on its policy with the money transportation company. Banks normally pay a significant amount of fixed fees for the re-filling, additional extra cost for the transportation with security arrangement.

Some banks might store 40% more cash in ATMs than the actual demand and banks might have thousands of nationwide ATMs. Therefore, a small optimization in business operations would contribute to high earnings.

Business use-case ATMs should not be filled with large amounts of cash which may bring low transport/logistics cost but high freezing & high insurance costs. On the other hand, if banks do not have the proper mechanism to track the usage pattern, then frequent re-filling ATMs will reduce freezing and insurance cost but increase logistics cost.

It is quite obvious that daily cash withdrawal amounts are time series. Therefore, in this typical cash demand forecast models we will present time series and regression machine learning models to troubleshoot the above use case. We will work on the demand for a single ATM (a group of ATMs can also be worked that is treated as a single ATM) to develop a model for the given data set.

We have to remember that, cash withdrawals from an ATM are not only time dependent. There could be seasonality, e.g. 1) people will have a tendency to withdraw money on Friday for the weekend or 2) end of the month when people get their salaries or 3) between 7–10th day of each month some people get their pension. Therefore, developing cash demand forecasting model for ATM network is a challenging task. Also, the chronological cash demand for every ATM fluctuates with time and often superimposed with non-stationary behavior of users.

2.2 Cash Replenishment Optimization:

Banks need to find out a way to optimize how much cash and how frequent to load cash into each ATM machine. Loading cash to an ATM has a cost independent from the amount loaded. We can reduce this cost by trying to reduce the number of replenishments. However, that means loading larger amounts each time an ATM is loaded, which generates an interest cost for each day that cash stays in the ATM. Therefore, an optimized solution tries to reduce the number of replenishment to decrease the loading cost and reduce the amount loaded into an ATM to reduce the interest cost. These two objectives are contradictory and therefore the optimum solution should do these decisions to minimize the overall cost. The inputs of the cash replenishment optimization problem (ROP) are as follows:
predicted withdrawal amounts for N consecutive days for an ATM,
cash transportation/loading cost, and,
the daily interest rate,
location (optional).

2.3 Examples of models used to solve this problem:

In this section we have some examples from other thesis about the resolution of this problem

2.3.1 A stochastic programming approach to cash management in banking:

The methods used in this paper are stochastic programming techniques to develop models for cash management in ATMs and compensation of credit card transactions. The models take into account the uncertainty of future customer demand and provide optimal solutions for short-term and mid-term problems with fixed and staircase costs. The short-term model with fixed costs results in an integer problem which is solved by a fast algorithm. The short-term model with fixed and staircase costs is solved through its MILP equivalent deterministic formulation. The mid-term model with fixed and staircase costs gives rise to a multi-stage stochastic problem, which is also solved by its MILP deterministic equivalent.

In the two-stage stochastic programming problem a set of first-stage decisions $x \in R^{n1}$ must be taken before the values of some stochastic parameters, dependent of random variable n, are known. The cost of first-stage decisions is represented by $c(x)$. Once n is known, we must adjust another set of second-stage variables $y \in R^{n2}$ to minimize the costs we incurred by our choice of x and the particular value of n; the function of second-stage costs is $Q(x, \xi)$. We look

for the decisions x that, in average, according to the distribution of n, provide the minimum costs. The general model (with fixed recourse) is:

$$\textbf{Min} \qquad c(x) + \Phi(x) \tag{2.1}$$

$$\textbf{Subject to} \qquad x \in X \subset R^{n1}$$

$$\textbf{Where} \qquad \Omega(x, \xi) = \min q(y, \xi) \tag{2.2}$$

$$\Phi(x) = E\epsilon|\Omega(x, \xi) \qquad \text{and subject to} \qquad W_y = h(\xi) - T(\xi)x,$$

$$y \in Y \subset R^{n1}$$

$c(x)$ and $x \in X$ in (1.1) are usually a linear function cx and a convex polyhedron defined by $X = \{x : Ax = b, x\ 0\}$, respectively. Integrality constraints $x \in Z$, either for a subset or for all the first-stage variables, may also appear in (1.1). $\Phi(x)$, known as the recourse function, is the future average cost of our first-stage decisions x, for all scenario (i.e., for all realization of ξ). $q(y, \xi)$ and $y \in Y$ in (1.2) are usually a linear function $q(\xi)y$ and nonegativity constraints $y \geq 0$, respectively.

Integrality constraints $y \in Z$ can also appear in $y \in Y$, either for a subset or for all the second-stage variables; indeed we will need them in the cash management problem for ATMs. The stochastic parameters of this general formulation are $q(\xi), h(\xi)$ and $T(\xi)$. In the models of next sections only $h(\xi)$ is stochastic, and is defined as $h(\xi) = \xi$, ξ being the customers demand of money, either from the ATM or through the credit card.

For some particular problems we can obtain a closed form solution for $Q(x, \xi) = q(y^*; \xi), y^*$ being the optimum second-stage decisions. In these cases we may be able to "compute" $\Phi(x) = \Omega[(x, \xi)]$, either evaluating the expectation (e.g., for discrete distributions or some low dimensional random variable) or approximating it (e.g., for multi-dimensional continuous variables) . This allows the solution of (1.1) only in terms of the first-stage decisions.

In general, however, no closed form exists for $Q(x, \xi)$ and we are forced to solve the extensive form or deterministic equivalent of the stochastic problem. For this purpose we consider n is a discrete random variable of s values $\xi_1, ..., \xi_s$ with probabilities $p_1, ..., p_s$. Each particular value $n_i, i = 1, ..., s$ is usually known as a scenario. Replicating for each scenario the second-stage variables (i.e, yi, $i = 1, ..., s$) and constraints, and combining problems (1.1) and (1.2), we obtain the following problem:

$$\textbf{min} \qquad c(x) \sum_{I=1}^{S} (p_i q)(y_i; \xi)$$

subject to $x \in X$

$$W y_i = h(\xi_i) – T(\xi_i) x \qquad i = 1, ..., s \qquad y_i \in Y \qquad i = 1, ..., s \tag{2.3}$$

roblem (1.3) can be solved with standard linear, integer or nonlinear programming algorithms if it is of moderate size; otherwise we need to apply specialized procedures that exploit the particular problem structure.

2.3.2 An optimization-based heuristic for a capacitated lot-sizing model in an automated teller machines network:

This thesis proposes a mathematical model to solve the cash inventory problem in ATM networks by determining the amount of money to place in ATMs and cash centers for each period over a given time horizon.

The problem is formulated as a Mixed Integer Programming (MIP) for the resulting optimization of this problem.

Parameters and decision variables regarding the model are listed as follows:

T = Number of time periods in the planning horizon

$t =$ Time period index; $t = 1, 2, ..., T$

$m =$ The total number of cash centers

$i =$ Cash center index; $i = 1, 2, ..., m$

$n_i =$ The total number of ATMs in each cash center i

$j =$ ATM index; $j = 1, 2, ..., n_i$

$D_{ijt} =$ Amount cash demanded of ATM j managed by cash center i at period t

$q_i =$ Refilling cost (Baht per trip) from the bank to cash center i

$o =$ Opportunity cost (per day) of money stored in cash center or ATM

$r_{ij} =$ Refilling cost (Baht per trip) from cash center i to ATM j

$a_{ij} =$ Sufficiently large number for an upper bound of a cash order for each ATM j from cash center i

$b_i =$ Sufficiently large number for an upper bound of a cash order for each cash center i

$C_{ATM} =$ Available cash storage capacity of each ATM

$C_{cc} =$ Available cash storage capacity of each cash center

Decision Variables:

$Q_{it} =$ Order quantity of cash center i at period t

$X_{ijt} =$ Order quantity of ATM j delivered by cash center i at period t

$J_{it} =$ Inventory level of money stored in cash center i at the end of period t

$I_{ijt} =$ Inventory level of money stored in ATM j managed by cash center i at the end of period t.

$\gamma_{it} =$ Binary setup variable indicating where order quantity is allowed for cash center i in period t (=1, if cash is refilled in cash center i in period t, 0 otherwise)

$\delta_{ijt} =$ Binary setup variable indicating where order quantity is allowed for ATM j under cash center i in period t (=1, if cash is refilled in ATM j managed by cash center i in period t, 0 otherwise)

Our formulation of the problem is described by the following mathematical program Equation:

$$\text{Minimize Total Cost} \quad \sum_{i=1}^{m}\sum_{t=1}^{T}(q_i\gamma_{it} + oJ_{it}) + \sum_{i=1}^{m}\sum_{j=1}^{ni}\sum_{t=1}^{T}(r_{ij}\delta_{ijt} + oI_{ijt}) \qquad (2.4)$$

Subject to

$$I_{ij(t-1)} + X_{ijt} - I_{ijt} = d_{ijt}; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.5}$$

$$X_{ijt} - (\alpha_{ij}\delta_{ijt}) \leq 0; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.6}$$

$$X_{ijt} - I_{ij(t-1)} \leq C_{ATM}; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.7}$$

$$J_{i(t-1)} + Q_{it} - J_{it} = \sum_{j=1}^{ni} X_{ijt}; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.8}$$

$$Q_{it} - (b_i\gamma_{it}) \leq 0; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.9}$$

$$Q_{it} - J_{i(t-1)} \leq C_{cc}; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.10}$$

$$X_{ijt}, I_{ijt} \geq 0; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.11}$$

$$Q_{it}, J_{it} \geq 0; \qquad \forall i \in \{1...m\}, t \in \{1...T\} \tag{2.12}$$

$$\delta_{ijt} \in 0, 1; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.13}$$

$$\gamma_{it} \in 0, 1; \qquad \forall i \in \{1...m\}, t \in \{1...T\} \tag{2.14}$$

The objective function (1.1) is to minimize the total costs for allocating cash in cash centers and ATMs (refilling costs and opportunity cost). Constraints (1.2) and (1.5) are the inventory balance constraints for ATMs and cash centers, respectively. Constraints (1.3) and (1.6) forces a set-up cost to be incurred for periods with positive ordering cash of each ATM and each center; it requires δ_{ijt} to be 1 if X_{ijt} is non-zero and it requires γ_{it} to be 1 if qit is non-zero. Constraints (1.4) and (1.7) represent the capacity constraints: the overall cash in ATM must remain lower than the available capacity. Constraints (1.8) and (1.9) are non-negative variables for order quantities and inventory level. Constraints (1.10) and (1.11) specify the binary setup variables.
For the large sizes of problem, the MIP formulation cannot solve or is very hard to solve in reasonable computational time. Thus, we develop an approach based on reformulating the model as a shortest path problem for finding a near-optimal solution.

2.3.3 The Shortest Path Reformulation:

Eppen and Martin (1987) presented the shortest path formulation approach to reformulate the capacitated lot-sizing problems based on a network diagram shown in Fig. 1. The new formulation proposes variables Z_{ijkl} that is the fraction of the accumulated demand from period k to period l.

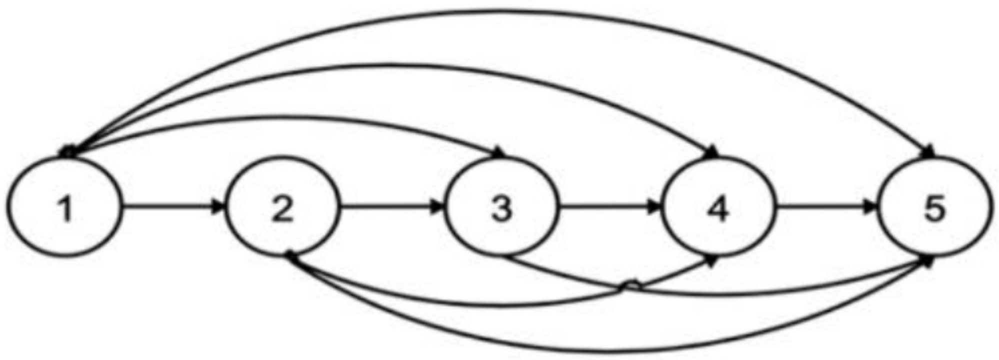


Figure 2.1: Shortest path representation of the lot sizing dynamic program for 1 item and 4 periods

Each node of network flow represents the period 1, 2, 3, 4 and a dummy period 5. The arc between nodes k and l means the choice of ordering the whole demand from period k to period $l - 1$ in period k.

The solution is to find the shortest path from node 1 to node 5.

They reformulate the MIP as described in Equation (1.1-1.11) as a SP reformulation. Only constraints that are related to capacity constraints, setup constraints and inventory balance constraints for managing ATMs will be relaxed. The rest of the constraints will not change from the original model. The decision variables and SP reformulation are as follows:

Decision Variables

Z_{ijkl} fraction of the total demand refilled in period k for demand in periods k to l of ATM j managed by cash center I Equation (1.12-1.26):

$$\text{Minimize Total Cost} \quad \sum_{i=1}^{m}\sum_{t=1}^{T}(q_i\gamma_{it} + o_t J_{it}) + \sum_{i=1}^{m}\sum_{j=1}^{ni}\sum_{t=1}^{T}(r_{ij}\delta_{ijt} + o_t I_{ijt}) \tag{2.15}$$

Subject to

$$\sum_{I=2}^{T+1} Z_{ij1I} = 1; \qquad \forall i \in \{1...m\}, j \in \{1...n\} \tag{2.16}$$

$$\sum_{k=2}^{I=1} Z_{ij1I} = \sum_{k=I+1}^{T+1} Z_{ij1I}; \qquad \forall I \in \{2, ...T\}, \forall i \in \{1...m\}, j \in \{1...n\} \tag{2.17}$$

$$\sum_{I=k+1}^{T+1} Z_{ij1I} \leq \delta_{ijkI}; \qquad \forall k \in \{1, ...T\}, \forall i \in \{1...m\}, j \in \{1...n\} \tag{2.18}$$

$$\sum_{I=k+1}^{T+1} \sum_{u=k} Z_{ij1I} = X_{ijt}; \qquad \forall k \in \{1, ...T\}, \forall i \in \{1...m\}, j \in \{1...n\} \tag{2.19}$$

$$I_{ij(t-1)} + X_{ijt} - I_{ijt} = d_{ijt}; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.20}$$

$$X_{ijt} + I_{ij(t-1)} \leq C_{ATM}; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.21}$$

$$J_{i(t-1)} + Q_{it} - j_{it} = \sum_{j=1}^{ni} X_{ijt}; \quad \forall i \in \{1...m\}, t \in \{1...T\} \tag{2.22}$$

$$Q_{it} - (bi\gamma_{it}) \leq 0; \qquad \forall i \in \{1...m\}, t \in \{1...T\} \tag{2.23}$$

$$Q_{it} - J_{i(t-1)} \leq C_{cc}; \qquad \forall i \in \{1...m\}, t \in \{1...T\} \tag{2.24}$$

$$X_{ijt}, I_{ijt} \geq 0; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.25}$$

$$Q_{it}, J_{it} \geq 0; \qquad \forall i \in \{1...m\}, t \in \{1...T\} \tag{2.26}$$

$$\gamma_{it} \in 0, 1; \qquad \forall i \in \{1...m\}, t \in \{1...T\} \tag{2.27}$$

$$\delta_{ijt} \in 0, 1; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, t \in \{1...T\} \tag{2.28}$$

$$Z_{ijkl} \geq 0; \qquad \forall i \in \{1...m\}, j \in \{1...n\}, k \in \{1...T\}, I \in \{2, ...T+1\} \tag{2.29}$$

The objective function (1.12) is to minimize the total costs for allocating cash in cash centers and ATMs (refilling costs and opportunity cost). Constraint (1.13) ensures that there is no more than one arc outgoing from node 1 for refilling each ATM. Constraint (1.14) ensures that if cash is placed in ATM in period l, it must exist in period +1. Constraint (1.15) forces the setup binary variables δijk to be one whenever money is refilled inATMj managed by cash center i in period t. Constraint (1.16) is used to find the order quantity for ATMj managed by cash center i at period t. Constraint (1.17) is used to find the inventory level of money stored in ATM j managed by cash center i at the end of period t. Constraints (1.17)-(1.25) are identical to Constraints (1.2), (1.4)-(1.11) in the original model. Constraint (1.26) enforces the non-negativity requirements for variables.

2.3.4 Computational results:

We use running time to evaluate the quality of the algorithms. Table 1 reports a solution, a solution time (CPU time in seconds) and a percent difference of the solutions of the two approaches for different numbers of ATMs with fixed number of cash centers and time periods.

No.of ATMs	MIP .... Total costs	.... CPU time(sec)	SPR .... Total costs	.... CPU time(sec)	% Diff of total costs
10	82.799	1.082	82.849	3.01	0.060
20	109774	2.19	109785	5.68	0.010
30	128.539	9.34	128.539	10.48	0.000
40	156.312	12.18	156.316	12.45	0.003
50	186.800	19.25	186.841	11.48	0.030
100	323.111	41.43	323.257	8.63	0.050
150	456.847	15556	456.847	13.07	0.000
200	-	> 8h	588.687	20.99	-
250	-	-	706.289	23.38	-
300	-	-	839.144	27.80	-
500	-	-	1.325.148	47.69	-
1000	-	-	2.631.580	106.60	-

Table 2.1: Comparison between the MIP formulation and the SPR approach

From Table 1, it can be seen that the MIP formulation can only solve small sizes of the problem and spends more computational time when the numbers of ATMs increase (Fig. 2). As the SPR algorithm can give solutions in less computational time although the problem is large. It was solved in a few minutes with 1,000 ATMs. Comparing the total costs of the two methods found that the SPR approach gave the optimal or close to optimal solution.

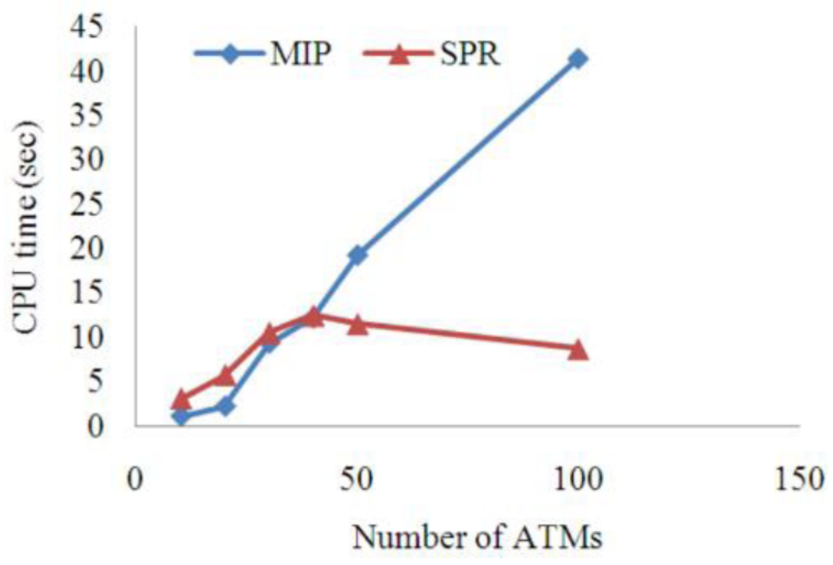


Figure 2.2: Illustrate a graph comparison of CPU Time between the MIP formulation and the spr approach.

The conclusion is that the proposed optimization-based heuristic approach (shortest path reformulation) is effective in solving the cash inventory problem in ATM networks. The approach is able to find near-optimal solutions for large-sized problems in a shorter amount of time compared to the traditional MIP formulation. The paper contributes to the field of inventory management and logistics by addressing the cash inventory problem in ATM networks and providing a new approach to solve it.

Chapter 3

Resolution of the problem:

In the scope of this thesis, two approaches are followed and applied to find an optimum solution for atm cash replenishment problem. The first one is mixt integer linear programming approach using SAS code and the second one non linear programming approach using MATLAB code. this section, I will give more detailed information about those approaches and how they are used for this specific solution.

3.1 ATM cash management Problem:

The goal of this application is to determine a schedule for allocation of cash inventory at bank branches to service a preassigned subset of automated teller machines (ATMs). Given historical training data per day for each ATM, we first define a polynomial fit for the predicted cash flow need. This is done using Casadi framework in Matlab environment to formulate the NLP model and determine the expected total daily cash withdrawals and deposits at each branch.

The modeling of this prediction depends on various seasonal factors, including the days of the week, the weeks of the month, holidays, typical salary disbursement days, location of the branches, and other demographic data. We then want to determine the multipliers that minimize the mismatch based on predicted withdrawals. The amount of cash allocated to each day is subject to a budget constraint. In addition, there is a constraint for each ATM that limits the number of days the cash flow can be less than the predicted withdrawal.

This scenario is referred to as a cash-out. Cash allocation plans are usually finalized at the beginning of the month, and any deviation from the plan is costly. In some cases, it may not even be feasible.

So, the goal is to determine a policy for cash distribution that balances the inventory levels while satisfying the budget and customer dissatisfaction constraints. By keeping too much cash-on-hand for ATM fulfillment, the banks will incur investment opportunity loss. In addition, regulatory agencies in many nations enforce a minimum cash reserve ratio at branch banks. According to regulatory policy, the cash in ATMs or in transit, do not contribute towards this threshold.

3.2 Mixed Integer Nonlinear Programming Formulation:

The most natural formulation for this model is in the form of a mixed integer nonlinear program (MINLP). Let A denote the set of ATMs and D denote the set of days used in the training data. The predictive model fit is defined by the following set of parameters: $(c^x_{ad}, c^y_{ad}, c^{xy}_{ad}, c^u_{ad})$ for each ATM a on each day d. Define variables (x_a, y_a, u_a) for each ATM that, when applied to the predictive model, give the estimated cash flow need per day, per ATM. In addition, define a surrogate variable fad for each ATM on each day that defines the net cash flow minus withdrawals given by the fit. Let B_d define the budget per day, K_a define the limit on cash-outs per ATM, and W_{ad} define the historical withdrawals at a particular ATM, on a particular day.

c^x and c^y are the costs associated with activating an ATM and making a replenishment, respectively.

A the set of atms **a**.

D the set of days **d**.

c^y_{ad} is the cost associated with making a replenishment for ATM a on day d.

c^{xy} and c^{xy}_{ad} are the costs associated with activating an ATM and making a replenishment simultaneously for ATM a on day d.

W_{ad} is the cash withdrawn by customers from ATM a on day d.

f_{ad} is the actual cash demand for ATM a on day d.

B_d is the maximum cash limit for day d.

u_a represents the amount of cash present in an ATM machine denoted by a.

Then the following NLP models this problem.

$$\textbf{Minimize} \qquad \sum_{a\in A}\sum_{d\in D}|f_{ad}|$$

Subject to:

$$c^x_{ad}x_a + c^y_{ad}y^a + c^{xy}_{ad}x_a y_a + c^u_{ad}u_a - W_{ad} = f_{ad} \quad \forall a \in A, \forall d \in D \tag{3.1}$$

$$|\{d \in D | f_{ad} < 0\}| \leq K_a \tag{3.2}$$

$$\sum_{a\in A}(fad + Wad) \leq B_d \tag{3.3}$$

$$(x_a, y_a) \in [0,1] \qquad \forall a \in A \tag{3.4}$$

$$u_a \geq 0 \qquad \forall a \in A \tag{3.5}$$

$$fad \geq -W_{ad} \forall a \in A, \qquad \forall d \in D \tag{3.6}$$

Inequalities (2) and (3) ensure that the solution satisfies the budget and cash-out constraints, respectively. Constraint (1) defines the surrogate variable fad, which gives the estimated net cash flow. In order to put this model into a more standard form, we first must use some standard model reformulations to linearize the absolute value and the cash-out constraint (3).

Linearization of Absolute Value. A well-known reformulation for linearizing the absolute value of a variable is to introduce one variable for each side of the absolute value.

The following system:

$$\min |y|, \qquad \text{is equivalent to} \qquad \min y^+ + y^-,$$

$$s.t. Ay \leq b \qquad s.t. A(y^+ - y^-) \leq b,$$

$$y^+, y^- \geq 0$$

Let f^+_{ad} and f^-_{ad} represent the positive and negative parts, respectively, of the net cash flow.Then, we can rewrite the model, removing the absolute value, as the following:

$$\textbf{Minimize} \qquad \sum_{a\in A}\sum_{d\in D}|f^+_{ad} + f^-_{ad}|$$

Subject to:

$$
\begin{aligned}
&c^x_{ad}x_a + c^y_{ad}y^a + c^{xy}_{ad}x^ay^a + c^u_{ad}u_a - W_{ad} = f^+_{ad} + f^-_{ad} && \forall a \in A, \forall d \in D \\
&\sum_{a \in D}(f^+_{ad} - f^-_{ad} + Wad) \leq B_d && \forall d \in D, \\
&|\{d \in D | (f^+_{ad} - f^-_{ad}) < 0\}| \leq K_a && \forall a \in A, \\
&x_a, y_a \in [0,1] && \forall a \in A \\
&u_a \geq 0 && \forall a \in A \\
&f^+_{ad} \geq 0 && \\
&f^-_{ad} \in [0, W_{ad}] && \forall a \in A, \forall d \in D
\end{aligned}
$$

Modeling the Cash-Out Constraints In order to count the number of times a cash-out occurs, we need to introduce a binary variable to keep track of when this event occurs. Let v_{ad} be an indicator variable that takes value 1 when the net cash flow is negative. We can model the following implication $f^-_{ad} > 0 \Rightarrow v_{ad} = 1$, or its contrapositive $v_{ad} = 0 \Rightarrow f^-_{ad} \leq 0$, by adding the constraint.

$$f^-_{ad} \leq W_{ad}v_{ad} \qquad \forall a \in A, \forall d \in D$$

Now, we can model the cash-out constraint simply by counting the number of days the netcash flow is negative for each ATM, as follows:

$$\sum_{d \in D}(v_{ad}) \leq K_a \qquad \forall a \in A$$

The MINLP model can now be written as follows:

Minimize:

$$\sum_{a \in A}\sum_{d \in D}(f^+_{ad} - f^-_{ad})$$

Subject to:

$$c^x_{ad}x_a + c^y_{ad}y^a + c^{xy}_{ad}x^a y^a + c^u_{ad}u_a - W_{ad} = f^+_{ad} - f^-_{ad} \quad \forall a \in A, \forall d \in D$$

$$\sum_{a\in A}(f^+_{ad} - f^-_{ad} + Wad) \leq B_d \qquad \forall d \in D,$$

$$f^-_{ad} \leq W_{ad}V_{ad} \qquad \forall a \in A, \forall d \in D$$

$$\sum_{d\in D} v_{ad} \leq K_a \qquad \forall a \in A,$$

$$x_a, y_a \in [0,1] \qquad \forall a \in A$$

$$u_a \geq 0 \qquad \forall a \in A$$

$$f^+_{ad} \geq 0 \qquad \forall a \in A, \forall d \in D$$

$$f^-_{ad} \geq 0 \in [0, W_{ad}] \qquad \forall a \in A, \forall d \in D$$

$$V_{ad} \in [0,1] \qquad \forall a \in A, \forall d \in D$$

We tried using this model with several of the available MINLP solvers on the NEOS Server for Optimization . However, we had little success solving anything but models of trivial size. We also solicited the help of several researchers doing computational work in MINLP, but thus far, none of the solvers have been able to successfully solve this problem. Presumably the difficulty comes from the non-convexity of the prediction function, then we testes an MILP model by changing the data (5 ATM and 10 Days). (Appendix)
Another approach is to formulate a nonlinear problem, I have used casadi framework in Matlab environment.

3.3 Nonlinear programming:

This model is a forecast model taken form a 'Mathew.Galati' book section 5 , so the search for the optimal multiplier is based on non-deterministic methods. Data is the least important. Rather, we want to provide the best possible solution in a reasonable amount of time.
therefore, it is perfectly acceptable to use NLP to approximate MINLP. In the original problem, we have the cash-out constraint in order to count the number of times a cash-out occurs, to count the number of payouts, to get a nonlinear programming we ignored this constraint and we worked

with NLP model:
This is the nonlinear programming:

$$\textbf{Minimize} \qquad \sum_{a\in A}\sum_{d\in D}|fad|$$

Subject to:

$$c^x_{ad}x_a + c^y_{ad}y^a + c^{xy}_{ad}x^a y^a + c^u_{ad}u_a - W_{ad} = fad \qquad \forall a \in A, \forall d \in D \tag{3.7}$$

$$\sum_{a\in A}(f_{ad} + W_{ad}) \leq B_d \qquad \forall d \in D \tag{3.8}$$

$$x_a, y_a \in [0,1] \qquad \forall a \in A \tag{3.9}$$

$$u_a \geq 0 \qquad \forall a \in A \tag{3.10}$$

$$fad \geq -W_a \qquad \forall a \in A \tag{3.11}$$

The objective function is to minimize the sum of the absolute values of the difference between the actual cash demand (fad) and the cash replenishment for all ATMs (a) and days (d).
The constraints are as follows:
The optimization problem is subject to a set of constraints, which are also represented as mathematical equation:
The first constraint is a linear equation that relates the decision variables xad, yad, and uad to the coefficients c^x_{ad}, c^x_{ad}, c^x_d, and c^u_{ad}, as well as a constant value Wad and the variable fad. The equation must hold true for all combinations of a and d in the sets A and D, respectively.
The second constraint The total cash replenishment for each day should not exceed the maximum cash limit (B_d) for that day.
The decision variables x and y are binary variables representing whether an ATM is active or not and whether a replenishment is made or not, respectively.
The decision variable u represents the amount of cash replenishment made for each ATM and day.
The fifth constraint ensures that the difference between the available and required cash is always greater than or equal to the negative of the maximum withdrawal amount for all ATMs and time periods.
In this model we have suppressed the the constrain of cash-out limit which count the number of

payouts, to get a nonlinear programming.
this is the nonlinear programming, we have used casadi framework in matlab environment to formulate it, after that ipopt slover has been used to solve this problem and find the optimal solution.

3.4 Why using CasADI framework:

CasADi is a software framework that provides a symbolic modeling language and numerical optimization methods for solving dynamic optimization problems. While CasADi is primarily written in $C++$, it also provides interfaces for various programming languages, including MATLAB.
There are several reasons why one might choose to use CasADi in the MATLAB environment:
Symbolic modeling: CasADi allows you to define optimization problems using a symbolic syntax. This makes it easier to express complex mathematical models in a concise and readable manner. You can leverage the power of symbolic computation while working in a familiar MATLAB environment.
Numerical optimization: it offers a wide range of numerical optimization algorithms, including both local and global optimization methods. These algorithms are specifically designed for solving dynamic optimization problems, where the objective and constraints depend on time or other variables. you can access these optimization algorithms and efficiently solve dynamic optimization problems.
Automatic differentiation: CasADi provides automatic differentiat ion capabilities, which enable the efficient computation of gradients and Hessians of mathematical expressions. This feature is particularly useful for optimization algorithms that rely on gradient-based methods, such as nonlinear programming. And you can easily compute derivatives of complex objective functions and constraints, which are essential for efficient optimization.
Integration with other MATLAB tools: MATLAB is a widely used platform for scientific and engineering computations. you can seamlessly integrate your optimization models with other MATLAB functionalities, such as plotting, data analysis, and simulation tools. This integration allows you to take advantage of the rich ecosystem of MATLAB for pre-and postprocessing tasks related to your optimization problems.
In summary, CasADi provides a powerful framework for symbolic modeling and numerical op-

timization of dynamic optimization problems. you can combine the benefits of symbolic computation, numerical optimization, automatic differentiation, and integration with other MATLAB tools, making it a popular choice for researchers and engineers working on optimization problems in MATLAB.

3.5 Why using Ipopt:

The **Ipopt** (Interior Point OPTimizer) solver is a popular open-source optimization solver that is widely used for nonlinear optimization problems with constraints. It is known for its effectiveness in solving large-scale optimization problems, and that is what we need.
Here are a few reasons why you might consider using Ipopt solver in MATLAB:
Nonlinear optimization: Ipopt is specifically designed for solving nonlinear optimization problems, which involve finding the optimal values for variables in the presence of nonlinear objective functions and constraints. If you have a problem that falls into this category, Ipopt can be a suitable choice.
Large-scale problems: Ipopt is efficient in handling large-scale optimization problems, wherethe number of variables and constraints is significant. It uses an interior point method, which allows it to handle problems with a large number of variables and constraints more effectively compared to some other solvers.
Constraint handling: Ipopt can handle both inequality and equality constraints in the optimization problem. It is capable of handling nonlinear constraints as well, making it suitable for a wide range of optimization applications.
Open-source and widely used: Ipopt is an open-source solver that is widely used in the optimization community. It has an active development community and is regularly updated with new features and bug fixes. Being open-source, it allows users to modify and customize the solver as per their requirements.
To use the Ipopt solver in MATLAB, you would typically need to install the Ipopt software package separately and then integrate it with MATLAB using the appropriate interface or wrapper provided by the Ipopt developers. You can refer to the Ipopt documentation and resources for detailed instructions on how to install and use Ipopt in MATLAB.

3.6 Digital application:

This section presents a computational analysis of the worked done in MATLAB of the nonlinear programming, After the execution we obtained this optimal objective value, which is the ob-value:

ob-value =

2.1424e+006

And the results for MILP model (5ATMs – 10days) is:

fval =

5.9704e+04

Conclusion

Which tools and approaches offer a long-term solution for controlling automated teller machine (ATM) inventories and planning cash deliveries, thereby improving the costefficiency of cash supply chains, the ATM users" satisfaction, and the security of cash logistics?" This question is divided into several sub-questions per research study, we have seen this in the chapters of our thesis.
The success of decomposition-based methods in real applications shows the potential for this area to make a positive impact on the field of mathematical programming, in this thesis we have seen various methods based on the decomposition methods for generating convex hull approximations of feasible solutions of an integer linear program.
Then find an optimal for ATM cash replenishment using Nonlinear programming, a model reformulated from an MINLP, we have used the Ipopt solver in MATLAB environement, we presented the computational results in the last section, we also tried an MILP model by changing data (5atms and 10days). Our research aim to ensure that the atms has enough cash available to dispense to customers. As people withdraw money from ATMs, the cash inside the machine decreases, and eventually, it needs to be replenished to maintain its functionality.

Appendix 1

```
addpath('C:\Users\salem\Desktop\MATLAB IPOPT CODE\casadi-windows
-matlabR2016a-v3.5.5')
 import casadi.* % import casadai framework

clear
clc
A = 20; % number of ATMs
D = 100; % number of days
%% import data. reshape it intomatrices (ATMs, DAYS).
full_data = importdata('ATM_data.mat'); % cx cy cz cu c w
Budget_data = importdata('Budget.mat'); %budget data
data_cx = reshape(full_data(:,1),100,20)';
data_cy = reshape(full_data(:,2),100,20)';
data_cz = reshape(full_data(:,3),100,20)';
data_cu = reshape(full_data(:,4),100,20)';
data_c = reshape(full_data(:,5),100,20)';
data_W = reshape(full_data(:,6),100,20)';
%%%%%%%%%%%%%Take parts from the data. A atm and D day.
%%%%%%%%%%%%%for full data. we use A=20 and D=100.
Cx = data_cx(1:A,1:D);
Cy = data_cy(1:A,1:D);
Cz = data_cz(1:A,1:D);
Cu = data_cu(1:A,1:D);
```

```
C = data_c(1:A,1:D);
Withdrawal = data_W(1:A,1:D);
B = Budget_data(1:D); % Budget
withdrawal= reshape(Withdrawal,A*D,1);
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 fp = SX.sym('fp',A,D); %%% f
 x = SX.sym('x',A);
 y = SX.sym('y',A);
 u = SX.sym('u',A);


Bin = []; %contains constraints and inquality constraints
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
costf = 0;% cosf function :
for i = 1:A

for j = 1:D
 costf = costf + abs(fp(i,j)); % calculate cost function

 %put the equality constraints on the first A*D element
 Bin = [Bin;Cx(i,j)*x(i) + Cy(i,j)*y(i) + Cz(i,j)*x(i)*y(i)
  + Cu(i,j)*u(i) + C(i,j) - Withdrawal(i,j) - fp(i,j)];
 end
end
 for j = 1:D
 %put the inequality constraints after the equality constraint
 Bin = [Bin;sum(fp(:,j)) + sum(Withdrawal(:,j))];

 end
 %%%%%% parameters constraints
lb = zeros(A*D+3*A,1); %% lower bounds set to zero.
lb(1:A*D) = -withdrawal; % lower bound for Fe is set to -w
%%%%%%%%%%
ub = ones(A*D+3*A,1)*inf; %% upper bound is set to inf(infinity)
which means there is no upper bound
ub(A*D+1:A*D+2*A) = 1; %% upper bound for x and y is set to 1 (x,y<1)
 %%% nonlinear constraints
 %put the lower and upper bounds to 0.
 %which means that only 0 is accpeted.
  lbg(1:A*D+D) = 0;
  ubg(1:A*D+D) = 0;
 %%% inquality constraints:
  lbg(A*D+1:A*D+D) = -ones(1,D)*inf;
  ubg(A*D+1:A*D+D) = B;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%chossing the solover
  OPT_var = [reshape(fp,A*D,1);x;y;u];%reshape x,y,u and Fe into one
    big vector
  nlp_prob = struct('f', costf , 'x' ,OPT_var ,'g' ,Bin); %put costf
  and the set of variables and the constraint
vector to solve

 %Solver options
  opts = struct;
  opts.ipopt.max_iter = 1000; %%max iteration number
 %%%% Ipopt solver
  solver = nlpsol('solver', 'ipopt' , nlp_prob, opts);
 %%%% initialising:
  x0 = [withdrawal',zeros(1,3*A)]; % choosing intialpoint to start from
```

```
%%%%solving
 sol = solver('x0', x0, 'lbx', lb, 'ubx', ub,...
 'lbg', lbg, 'ubg', ubg);
%%%%% optimal variables :

 F_optimal = reshape(sol.x(1:A*D),A,D); % f from 1 to A*D
 x_optimal = sol.x(A*D+1:A*D+A); % x from A*D+1 to A*D+A
 y_optimal = sol.x(A*D+A+1:A*D+2*A); % y from A*D+A+1 to A*D+2*A
 u_optimal = sol.x(A*D+2*A+1:A*D+3*A); % u from A*D+2*A+1 to A*D+3*A
%%%%% objective value
 ob_value = sol.f;
```

Appendix 1 Detailed

Importing the necessary libraries and clearing the workspace.
Importing data from two .mat files: 'ATM-data.mat' and 'Budget.mat'. The data represents information related to ATMs and their daily operations.
Reshaping the imported data into matrices based on the number of ATMs (A) and the number of days (D).
Setting up the initial parameters and separating data for a specific ATM (A) and specific days (D).
The main optimization problem is formulated as follows:
Objective:
Minimize the cost function, which is the sum of the absolute values of the variables fp (A $\times$ D matrix), representing some decision variables.
Constraints:
For each ATM and day, there is an equality constraint related to the balance:

$$C_x(i,j)^*x(i)+C_y(i,j)^*y(i)+C_z(i,j)^*x(i)^*y(i)+C_u(i,j)^*u(i)+C(i,j)-Withdrawal(i,j)-fp(i,j)=0$$

where x, y, and u are decision variables for ATM i, and fp represents the decision variable for a function.

This equality constraint ensures that the balance at the end of each day is equal to the initial balance, considering the withdrawals and the function fp.
For each day, there is an inequality constraint related to the budget:
sum(fp(:,j)) + sum(Withdrawal(:,j)) <= B(j)
This inequality constraint ensures that the total of all fp and withdrawals on each day doesn't exceed the available budget B(j).
The code then defines the decision variables, bounds, and constraints to set up the optimization problem. It uses the IPOPT solver from CasADi to find the optimal solution.
The optimal solution provides the values of the variables fp, x, y, and u that minimize the cost function while satisfying the given constraints. The optimal value of the cost function (ob-value) represents the optimized objective value.
Note: Without specific data for the 'ATM-data.mat' and 'Budget.mat' files, we cannot assess the actual results or the real-world implications of this optimization problem. The explanation above is based solely on the code's structure and provided comments.

Appendix 2

```
clear
clc
prob = mpsread('atm_5_10_1.mps')
[x,fval ,exitflag ,output] = intlinprog(prob);
```

Appendix 1 results

The code is solving an integer linear programming problem defined by the MPS file 'atm_5_10_1.mps'. The objective function and constraints of the optimization problem are likely specified in that file. The intlinprog function is used to find the optimal values of the decision variables that satisfy the constraints and optimize the objective function. The result is stored in the variables x, fval, exitflag, and output.

Bibliographie

[1] "Mixed-Integer Linear Programming (MILP): Model Formulation" (PDF). Retrieved 16 April 20182. Papadimitriou, C. H.; Steiglitz, K. (1998). Combinatorial optimization: algorithms and complexity. Mineola, NY: Dover. ISBN 0486402584.

[2] J. Czyzyk, M. Mesnier, and J. More. The NEOS server. IEEE Journal on Computational Science and Engineering, 5:68–75, 1998.

[3] (http://www.thescipub.com/jmss.toc).

[4] https://www.google.com/search?q=trust+bank+algeria+histoire&oq=TRU&aqs=chrome.0.69i59j69i57j46i39i199i465i650j0i131i433i512l2j69i60l2j5.1759j0j4&sourceid=chrome&ie=UTF-8 .

[5] Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., Diehl, M.: Casadi: a soft-Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., Diehl, M.: Casadi: a software framework for nonlinear optimization and optimal control. Mathematical Programming Computation 11, 1–36 (2019)ware framework for nonlinear optimization and optimal control. Mathematical Programming Computation 11, 1–36 (2019)rol. Mathematical Programming Computation 11, 1–36 (2019).

[6] C. Barnhart, C. A. Hane, and P. H. Vance. Using branch-and-price-and-cut to solve origindestination integer multi-commodity flow problems. Operations Research, 48:318–326, 2000.

[7] J.E. Beasley. Lagrangean relaxation. In C.R. Reeves, editor, Modern Heuristic Techniques for Combinatorial Optimization. Wiley, 1993.

[8] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch and price: Column generation for solving huge integer programs. Operations Research, 46:316–329, 1998.

[9] M. Poggi de Aragao and E. Uchoa. Integer program reformulation for robust branch-andcut-and-price. Working paper, Pontif'ica Universidade Catolica do Rio de Janeiro, 2004. Available from http://www.inf.puc-rio.br/uchoa/doc/rbcp-a.pdf.

[10] M. de Moraes Palmeira, A. Lucena, and O. Porto. A Relax and Cut Algorithm for Quadratic Knapsack Problem. Technical report, Universidade Federal do Rio de Janeiro, 1999.

[11] M.L. Fisher. The Lagrangian relaxation method for solving integer programming problems. Management Science, 27:1–18, 1981.

[12] M. Guignard. Lagrangean relaxation. Top, 11:151–228, 2003.

[13] N. Kohl, J. Desrosiers, O.B.G. Madsen, M.M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. Transportation Science, 33:101–116, 1999.

[14] G. Laporte, Y. Nobert, and M. Desrouchers. Optimal routing with capacity and distance restrictions. Operations Research, 33:1050–1073, 1985.

[15] A. Lucena. Non-delayed relax-and-cut algorithms. Working paper, Departmaneto de Administrac¸ao, Universidade Federal do Rio de Janeiro, 2004.

[16] C. Martinhon, A. Lucena, and N. Maculan. A relax and cut method for the vehicle routing problem. Unpublished working paper, 2001.

[17] G.L. Nemhauser and L.A. Wolsey. Integer and Combinatorial Optimization. Wiley, New York, 1988.

[18] J.M. van den Akker, C.A.J. Hurkens, and M.W.P. Savelsbergh. Time-indexed formulations for machine scheduling problems: Column generation. INFORMS Journal on Computing, 12:111–124, 2000.

[19] F. Vanderbeck. Lot-sizing with start-up times. Management Science, 44:1409–1425, 1998.

[20] https://coral.ise.lehigh.edu/ ted/files/papers/MatthewGalatiDissertation09.pdf.

[21] Jordi Castro,' A stochastic programming approach to cash management in banking Department of Statistics and Operations Research, Universitat Politecnica de Catalunya, Jordi Girona 1-3, 08034 Barcelona, Catalonia, Spain.

[22] Supatchaya Chotayakul, Peerayuth Charnsetthikul, Juta Pichitlamken and John Kobza, 'AN OPTIMIZATION-BASED HEURISTIC FOR A CAPACITATED LOT-SIZING MODEL IN AN AUTOMATED TELLER MACHINES NETWORK', 1Department of Industrial Engineering, Faculty of Engineering, Kasetsart University, Chatuchak, Bangkok 10900, Thailand Department of Industrial Engineering, Faculty of Engineering, The University of Tennessee, Knoxville, Tennessee 37996, US 2013.

Abreviation

LP:	Linear Programming
ILP:	Integer Linear Programming
MILP:	Mixt Integer Linear Programming
NLP:	Nonlinear Programming
TBA:	Trust Bank Algeria
CRO:	Cash Replenishment Optimization