BANCA D'ITALIA
EUROSISTEMA

Mercati, infrastrutture, sistemi di pagamento

(Markets, Infrastructures, Payment Systems)

Liquidity Optimization in Gross Settlement Systems with Quantum Reordering: Application to TARGET2

by Valerio Astuti, Adriano Baldeschi, Luca Bastianelli, Giuseppe Bruno, Ajit Desai, Danica Marsden, Riccardo Russo

March 2026

Number 78

BANCA D'ITALIA
EUROSISTEMA

Mercati, infrastrutture, sistemi di pagamento

(Markets, Infrastructures, Payment Systems)

Liquidity Optimization in Gross Settlement Systems with Quantum Reordering: Application to TARGET2

by Valerio Astuti, Adriano Baldeschi, Luca Bastianelli, Giuseppe Bruno, Ajit Desai, Danica Marsden, Riccardo Russo

Number 78 – March 2026

The papers published in the 'Markets, Infrastructures, Payment Systems' series provide information and analysis on aspects regarding the institutional duties of the Bank of Italy in relation to the monitoring of financial markets and payment systems and the development and management of the corresponding infrastructures in order to foster a better understanding of these issues and stimulate discussion among institutions, economic actors and citizens.

The views expressed in the papers are those of the authors and do not necessarily reflect those of Banca d'Italia.

The series is available online at www.bancaditalia.it.

Printed copies can be requested from the Paolo Baffi Library: richieste.pubblicazioni@bancaditalia.it.

ISSN 2724-6418 (online)
ISSN 2724-640X (print)

Banca d'Italia
Via Nazionale, 91 - 00184 Rome - Italy
+39 06 47921

Designed and printing by the Printing and Publishing Division of Banca d'Italia

LIQUIDITY OPTIMIZATION IN GROSS SETTLEMENT SYSTEMS WITH QUANTUM REORDERING: APPLICATION TO TARGET2

Valerio Astuti*, Adriano Baldeschi*, Luca Bastianelli*, Giuseppe Bruno*, Ajit Desai, Danica Marsden**, Riccardo Russo***

Abstract

In High-Value Payment Systems (HVPSs) with gross settlement, which require high levels of liquidity, optimizing the processing order could provide significant savings in the liquidity allocation needed for institutions to stay solvent. Building on McMahon et al. (2024), who showed that a hybrid quantum solver can improve intraday liquidity efficiency in Canada's HVPS, we apply a similar technique to payments between Italian institutions in TARGET2, using a Constrained Quadratic Model (CQM) Solver. Our application optimizes payment batches, yielding average daily liquidity savings between EUR 23 million and EUR 38 million, over a 35-day sample. The use of machine learning techniques also allows the identification of batch features that enhance these savings. Finally, we benchmark the results against those obtained using a Simulated Annealing Algorithm (SAA), finding comparable liquidity savings. The SAA is extended to handle larger batch sizes, which are still challenging for current quantum hardware, demonstrating an increase in liquidity savings more than proportional to the growth in batch size.

JEL Classification: C61, C63, C83, E42, E58.

Keywords: quantum annealing, simulated annealing, combinatorial optimization, NP-hard, high-value payment systems.

Sintesi

Nei sistemi di pagamento di importo rilevante con regolamento su base lorda, che richiedono livelli elevati di liquidità, ottimizzare l'ordine di esecuzione delle transazioni può ridurre in modo significativo le risorse che le istituzioni devono allocare per mantenere la propria solvibilità. Seguendo McMahon et al. (2024), che hanno mostrato come un *hybrid quantum* solver possa migliorare l'efficienza della *intraday liqudity* del sistema *high-value* canadese, applichiamo una tecnica simile ai pagamenti tra istituzioni italiane in TARGET2, utilizzando un *Constrained Quadratic Model (CQM) Solver*. L'applicazione ottimizza blocchi di pagamenti, generando risparmi di liquidità medi giornalieri compresi fra 23 e 38 milioni di euro, su un campione di 35 giorni. L'utilizzo di tecniche di *machine learning* consente inoltre di identificare le caratteristiche dei blocchi che incrementano tali risparmi. Infine, confrontiamo i risultati con quelli ottenuti tramite un algoritmo di *Simulated Annealing (SAA)*, riscontrando risparmi di liquidità di dimensioni analoghe. Il SAA viene esteso a blocchi di dimensioni più elevate, ancora proibitive per l'hardware quantistico attuale, mostrando un incremento dei risparmi di liquidità più che proporzionale rispetto alla crescita delle dimensioni dei blocchi.

* Banca d'Italia

** Bank of Canada

INDEX

1 Introduction

High-Value Payment Systems (HVPSs), such as T2 and TARGET2 in the Eurozone, Fedwire in the US, and Lynx in Canada, are vital components of financial systems. They facilitate the transfer of large-value transactions, support liquidity management, and aid in the implementation of monetary policy. Central banks, which either operate or oversee these systems, are responsible for ensuring their safety and efficiency. Most HVPSs are real-time gross settlement systems (RTGS), allowing settlement in real-time and on a gross basis, making the liquidity efficiency of these systems dependent on the order of transaction settlements.[1]

Traditionally, strategies to enhance liquidity efficiency in payment systems have focused on promoting timely coordination of payment submissions among participants and employing bypass or netting-based liquidity savings mechanisms (Bech and Garratt 2003; Martin and McAndrews 2008; Rivadeneyra and Zhang 2022). However, a recent work by McMahon *et al.* (2024) introduced a novel approach: liquidity recycling through optimal reordering of payment batches. This reordering process, however, is an NP-hard combinatorial optimization problem, with computational complexity growing rapidly with the number of payments in a batch. They demonstrated that employing a hybrid quantum solver to the problem of finding the optimal reordering can be effectively addressed within realistic time constraints, improving liquidity efficiency without significantly increasing payment delays. They evaluate the efficacy of this process using data from the Canadian HVPS, showcasing its potential to optimize payment processing.

Building on their work, we extend the application of the quantum reordering algorithm to the Italian segment of Eurozone's TARGET2 system using the Constrained Quadratic Model (CQM) solver provided by D-Wave (D-Wave Systems 2021).[2] The solver is used to identify the order that maximizes liquidity savings in each batch. The reordering optimization applies to batches of payments, collected in the order in which they are submitted to the system. The main steps of payments processing are illustrated by Figure 1.

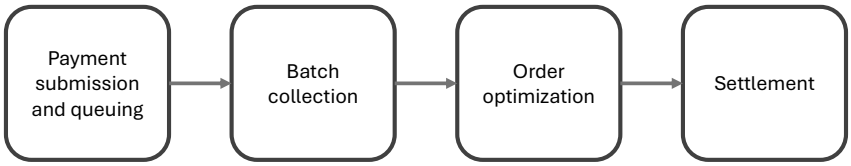

Figure 1: Steps involved in the processing of payments: after an accumulation time in which payments are grouped into batches, their order is optimized and finally they are settled.

Through this process, we achieve daily liquidity savings of up to EUR 167 million, with average savings of EUR 23.16 million and EUR 38.35 million for batches of 70 and 140 payments, respectively.

[1] Liquidity efficiency, also known as the Turnover or Liquidity Efficiency Ratio (LER), is the ratio of total value settled to total liquidity used (Benos et al. 2014; Alexandrova et al. 2023; Desai et al. 2023). As an example, let's take three payments $p_1 : €1$ from A to B, $p_2 : €3$ from A to B, and $p_3 : €2$ from B to A. If settled in this order, A needs €4 liquidity and B needs €0, yielding a liquidity efficiency of $\frac{€6}{€4} = 1.5$. If settled in the order p_1, p_3, p_2, A needs €2 liquidity and B needs €1, and the liquidity efficiency is $\frac{€6}{€3} = 2$.

[2] We also investigate the impact of the CQM solver's *time_limit* parameter, which defines the maximum runtime (in seconds) that the solver is permitted to spend on the simulation D-Wave Systems (2021), and how it affects solution quality and liquidity savings. This aspect was not studied in McMahon et al. (2024).

In relative terms the average savings are of 0.4% of the daily liquidity reserves for batch size 70, and of 0.7% for batch size 140, with maximum savings of 4.0% and 4.4%. The average time to collect a batch of 70 payments is 150 seconds, while the time required to optimize each batch is just about 5 seconds of quantum annealing time.[3] If we consider the classical compute overhead as well (hybrid total), the solution time is on the same order as the batch collection time.

While the study by McMahon et al. (2024) primarily demonstrated liquidity savings using quantum batch reordering algorithm, it did not delve into the underlying reasons for why certain payment batches are optimized through reordering, while others are not.[4] In particular, while proving that it is possible to achieve a meaningful saving on average, the optimization algorithm produces results which are heterogeneous across different batches. We address this gap by developing a Machine Learning (ML) classification framework that aims to identify the key features of payment batches that predict liquidity savings when processed through the reordering algorithm. This classification could be used in a refinement step of the optimization process: when limited resources are available, they could be assigned to the batches with highest probability to be optimizable, while retaining the original order for the remaining ones. We find that batches of 70 payments are less optimizable if fewer than 5 participants act as both senders and receivers. The optimizability of the batches increases steadily with the number of participants acting as both senders and receivers up to 15 participants, and then starts dropping slowly with further increases. Additionally, an increasing number of unique receivers enhances the optimizability of the batches. Interestingly, the other network properties of the batches did not significantly influence the optimizability prediction of our model. In essence, our ML framework adds a new dimension to the analysis of liquidity efficiency in payment systems by providing a deeper understanding of the factors that influence batch optimizability.

Finally we focus on the benchmarking and possible extensions of the CQM optimization algorithm. In McMahon et al. (2024) the authors focused on benchmarking the CQM solver against standard liquidity savings mechanisms (such as bypass and netting), and a classical exact solver which makes use of certain heuristics (SCIP).[5] In this paper, we benchmark against a classical heuristic solver - a simulated annealing algorithm (SAA) - due to its effectiveness in high-dimensional search spaces and its similarity with CQM as a heuristic solver (Kirkpatrick et al. 1983; Kadowaki and Nishimori 1998; Aarts et al. 2005). First, we demonstrate that the SAA, with a sufficiently long annealing schedule (i.e. with a sufficient number of iterations), can produce comparable savings to the one obtained with the CQM algorithm, both for batches of 70 and 140 payments. Next, we demonstrate that both the CQM and SAA consistently achieve savings close to the maximum optimizable amount, or upper bound. These upper bounds are determined by multilaterally netting all payments for all participants within each batch, similar to the process used in batch deferred net settlement systems. While these upper bounds using multilateral netting on each batch provide a benchmark for liquidity optimization, they cannot always be replicated by reordering the same batch of payments in RTGS systems.

The use of a flexible classical benchmark allows us to estimate the amount of savings actually realizable for larger batch sizes. While the current quantum hardware cannot achieve meaningful savings for larger batch sizes, we can estimate the effect of the optimization in larger batches with the classical benchmark. Even though the classical optimization algorithm can explore only a small

[3] It is important to note that the *real-time* nature of the payment systems does not imply the instantaneous settlement of payments. The only rigid temporal constraint is the settlement of the initiated payments before the end of the day. The effect of small delays during the day is to slightly increase the risks associated to an operation, but for delays of the order of minutes this increase is for all intents negligible.

[4] A batch is considered optimized if the liquidity requirement using the CQM (or SAA) solver-provided order is less than that of the original first-in-first-out (FIFO) order.

[5] The SCIP (Solving Constraint Integer Programs) solver is an open-source, versatile tool capable of solving mixed-integer programming and constraint integer programming problems (Bestuzheva et al. 2021).

part of the permutation space on which we want to optimize, the results available for smaller batches show comparable results with the quantum annealer. Given the quantum advantage of the CQM we can expect - when a powerful enough quantum hardware will be available - to obtain results bounded between the classical benchmark and the theoretical upper bound. We then extend the SAA to handle batch sizes of up to 700 payments, a scale that is challenging for current quantum hardware due to the size of the problem.[6] This extension demonstrates daily liquidity savings of up to EUR 3.9 billion, with an average of 270 million over the same 35-day sample period. This represents a more than tenfold increase in savings using the reordering approach, although it comes with the increased delay cost (on average 25 minutes) of collecting batches of 700 payments. The result highlights the potential of simulated annealing to efficiently manage larger payment batches and significantly improve overall liquidity efficiency in HVPSs. This is important given the current limitations of quantum hardware and the high cost of quantum computing. By using SAA on classical computing systems as a scalable alternative, we enhance the practicality of the approach, making it a more feasible solution for improving liquidity efficiency in HVPS and offering an estimation of the possible savings achievable by the quantum algorithm in larger batches. The scalability of the optimization algorithm is crucial for practical applications in HVPS, where batch sizes can vary significantly, and efficient liquidity management is essential for financial stability and operational efficiency.

The remainder of the paper is structured as follows: In section 2, we discuss the TARGET2 sample data used for the experiment. In section 3, we present the formulation of the CQM optimization approach, followed by the batch optimizability study in section 4. Next, in section 5, we present CQM results and benchmarking results using SAA for small batch sizes, followed by SAA results with large batch sizes in section 6. Finally, we conclude in section 7.

2 TARGET2 Italian Payments Data

The European System of Central Banks (ESCB) runs a single monetary policy since the inception of the euro. A payment system able to deliver liquidity within the euro-area countries is a basic pillar for the conduction of this single monetary policy. In Europe, the Trans-European Automated Real-Time Gross Settlement Express Transfer System (TARGET) was designed from the ground up with the goal of linking together existing national gross settlement systems through the Interlinking Network. These systems were then harmonized into the system known as TARGET2, to provide more flexibility for coping with the enlargement to handle all-EU payments. TARGET2 is a single shared platform (SSP) set up by three central bank providers, namely Banca d'Italia, Deutsche Bundesbank and Banque de France (the 3CB).[7] On March 20, 2023, TARGET2 was replaced by T2: the RTGS system launched by the Eurosystem to strengthen the technical and functional aspects of its predecessor, aiming at improving efficiency and reducing operating costs.

In 2021 TARGET2 handled 90% of the large-value payments (larger than €500,000), and has been one of the most important HVPSs in the world.[8] The average daily number of payments settled

[6] In McMahon et al. (2024), the authors solved the problem for batch sizes up to 700 using the CQM solver, but the time required to obtain reliable solutions was significantly higher, increasing delay costs and rendering an application of the strategy to large payment batches impractical.

[7] In 2008, the Banco de España joined, thereby completing the group of four central banks (4CB) responsible for managing the technical platform.

[8] The remaining 10% were processed via the Euro1 system, which is owned by the private company EBA Clearing (https://www.ebaclearing.eu/services/euro1/overview/). This is similar to the dual system used in the United States, composed of Fedwire, operated by the central bank, and CHIPS, operated by the private sector. In Canada, on the other hand, there is only one HVPS, referred to as Lynx.

in TARGET2 in 2021 was around 373,000 and the average daily amount was close to €1.9 trillion. Germany plays a predominant role in TARGET2 traffic between the different national components, accounting for more than half of the number of settled transactions and 38% of their value. Italy has a 9% share of the volume and 3% of the value.[9] In this work, we use a sample of 35 days in 2021 of data limited to TARGET2 transactions between Italian financial institutions only (i.e. both sender and receiver are Italian financial institutions). According to the ECB's payments statistics report,[10] in 2021 the Italian segment had a total of more than €15 trillion processed and 8.8 million transactions sent via TARGET2. The TARGET2 system operated Monday through Friday from 7 a.m. to 6 p.m. Central European Time (CET), and was closed on weekends as well as statutory holidays. Note that, for brevity, the sample days are referred to by their ordinal numbers, labeled as Day 1 through Day 35.

For the present work, the sample of transactions was extracted from the Italian segment of TARGET2. In particular, payments between Italian institutions with Normal priority have been selected ("urgent" payments have been discarded).[11] In the period considered in the sample, 92 financial institutions participated in the payment exchanges. Inbound and outbound traffic passing through the top five banks accounts for 86% of the value traded and 61% of the volume of transactions.[12]

Figure 2 shows a box plot of the distribution of submitted payments by hour, as a function of the time of day. The hours with the highest volume of activity are the beginning and middle of the day, while the volume drops off significantly in the final hours of the day. The payment amounts per batches of 70 payments are plotted (in logarithmic scale) as a function of the time of the day in Figure 3. It is apparent that the distribution of payment amounts spans many order of magnitudes, from tens of thousands of euro to tens of billions. In addition, this figure confirms the more intense activity at the beginning of the day, and the abating volume toward the end of the day.

Another relevant statistics in this context is the accumulation time for various batch sizes (i.e. the time needed to collect the number of payments necessary to complete a batch). This quantity dominates the delay necessary to optimize the payments queue: the computation times needed to find the optimal (or pseudo-optimal) orderings are most of the time negligible with respect to the time needed to accumulate enough payments in each batch. The distributions of accumulation times is shown in Figure 4. It can be observed that these times grow approximately linearly with the batch size. The wide tails of the distributions of accumulation times imply the presence of occasional outliers. To avoid excessive waiting times in these cases, a cut-off can be added to the optimization procedure. Finally, Figure 5 and Figure 6 present the distributions of the number of payers and payees per batch as a function of the hour of payment submission. While presenting a large variability, we see that the typical number of payers is small both at the beginning and at the end of the day, while the typical number of payees is approximately constant throughout the day, with the exception of the last hour.

[9] TARGET Annual Report 2021 https://www.ecb.europa.eu/press/targetar/html/ecb.targetar2021.en.html

[10] https://data.ecb.europa.eu/publications/payments-statistics/3075435

[11] The optimization process used here is potentially not suitable for urgent payments because it requires some waiting time until a batch of payments of a given size has accumulated. In the period considered, however, urgent payments represent only about 4% of the total number of payments.

[12] The sample considered is representative of the ordinary flows in the Italian portion of the TARGET2 system over longer time scales.

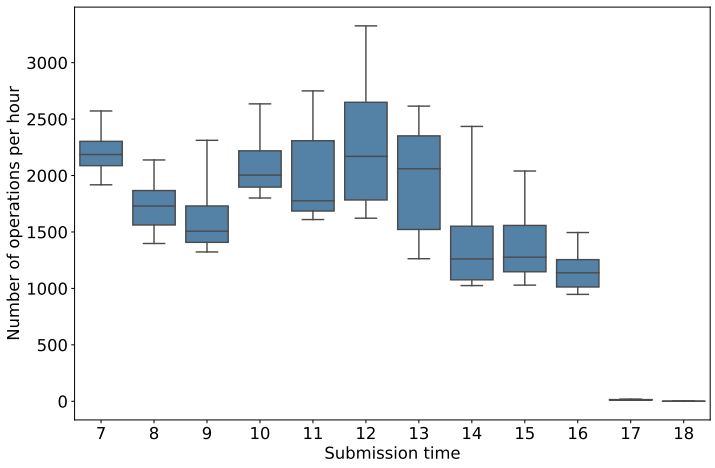

Figure 2: Statistics of the number of submitted payments per hour. The lines internal to the boxes denote the median of the distributions, the boxes themselves mark the 25^{th} and 75^{th} percentiles and the whiskers represent the 5^{th} and 95^{th} percentiles of the distribution.

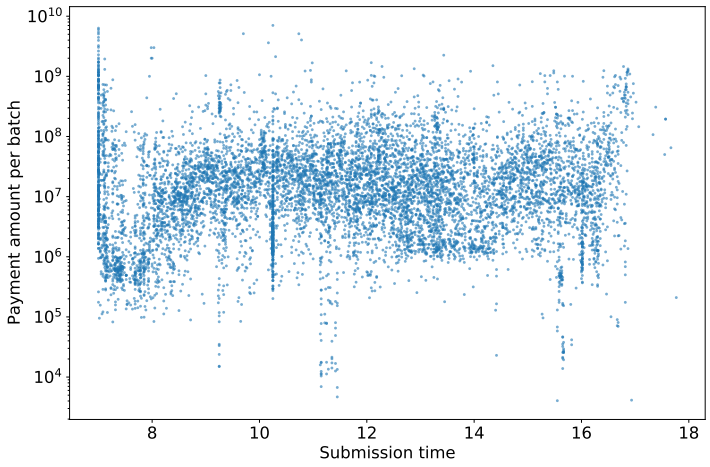

Figure 3: Distribution of payment amounts in euros per batch of 70 payments (log scale).

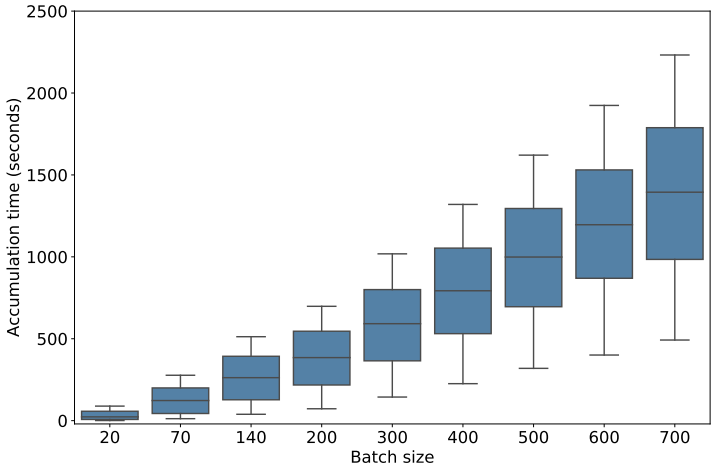

Figure 4: The time required (in seconds) to fill a batch of payments as a function of queue size. The lines internal to the boxes denote the median of the distributions, the boxes themselves mark the 25^{th} and 75^{th} percentiles and the whiskers represent the 5^{th} and 95^{th} percentiles of the distribution.

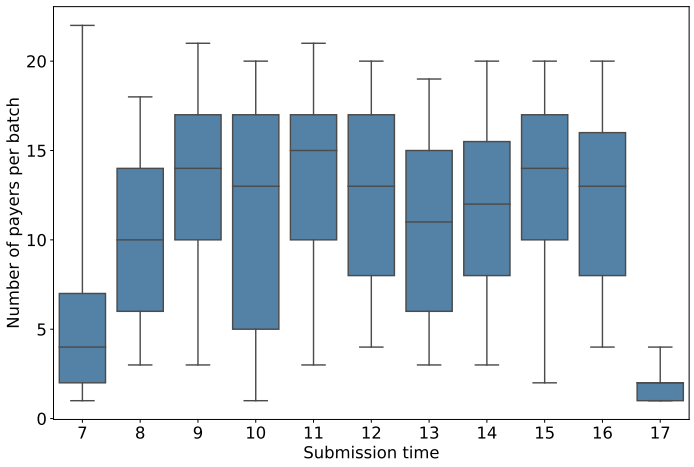

Figure 5: Number of payers per batch of 70 payments against submission hour of day. The lines internal to the boxes denote the median of the distributions, the boxes themselves mark the 25^{th} and 75^{th} percentiles and the whiskers represent the 5^{th} and 95^{th} percentiles of the distribution.

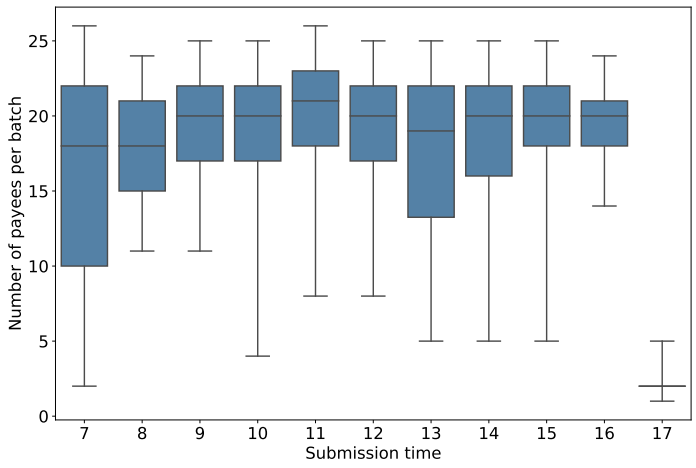

Figure 6: Number of payees per batch of 70 payments against submission hour of day. The lines internal to the boxes denote the median of the distributions, the boxes themselves mark the 25^{th} and 75^{th} percentiles and the whiskers represent the 5^{th} and 95^{th} percentiles of the distribution.

3 CQM Solver Formulation and Initial Conditions

In this section we briefly present the formulation of the payment queue batch quantum optimization problem. The CQM solver takes objectives and constraints directly as inputs. As described in McMahon et al. (2024), the problem is characterized as:

$$\text{Objective:} \quad \min \sum_{\alpha} b_{\alpha} \tag{1}$$

$$\text{Subject to:} \quad b_{\alpha} + N_{\alpha}(0) + mNDP_{\alpha} + \sum_{i} \sum_{\tau \leq t} f(\alpha, i)\, x_{i,\tau} \geq 0, \forall \alpha \tag{2}$$

$$\sum_{i} x_{i,t} = 1, \forall t \tag{3}$$

$$\sum_{t} x_{i,t} = 1, \forall i \tag{4}$$

where:

b_α	is the participant α's liquidity allocation. When minimized, it is the amount of additional liquidity that α would need to have to permit settlement of the payments in that batch (i.e., to avoid gridlock). Equivalently, it is the increase of $mNDP_\alpha$ after this batch is settled
$N_\alpha(0)$	is participant α's net position immediately before this batch is settled
$mNDP_\alpha$	is participant α's maximum net debit position incurred during previous batches earlier in the day
$x_{i,t}$	is a binary decision variable that indicates whether payment i is settled ($x_{i,t} = 1$) or not ($x_{i,t} = 0$) at position t in the final queue
$f(\alpha, i)$	is a function that returns v (the value or amount of payment i) if α is the payee; $-v$ if α is the payer; and 0 otherwise

In the above formulation of the problem the index i denotes the payment number in the original order, and the indices t and τ denote the payment number in the final order. Solving the problem amounts to finding the matrix that represents the reordering of the payments from i to t, i.e., $\{x_{i,t}\}$ which minimizes the aggregate up front liquidity required by participants. For more details, refer to McMahon et al. (2024). We apply it to the 35 days worth of illustrative data representing the Italian portion of the TARGET2 payments system described above, dividing each sample day into batches of size 70 and 140, and solving each batch on the CQM hybrid solver. While the former study used payments made earlier in the day to inform the starting conditions at 8am for each day in the Canadian sample, in this case we begin each day with zero net position and maximum net debit position (mNDP) for each participant.[13] The main reason for the different approach is that before the 7am start time there are not many payments in the Italian component of TARGET2, unlike the Canadian system before 8am, and testing revealed negligible effect on the outcome for the Italian case.

With this process we achieve daily the end of the day liquidity savings of up to EUR 167 million and averages of 23.16 and 38.35 million respectively for batches of size 70 and 140 (respectively 0.4% and 0.7% of the total daily liquidity needs). The summary of the results for the entire sample is presented in section 6 in Table 1 and 2.

3.1 CQM solver `time_limit` parameter

In the CQM hybrid solver, besides the objective function and the constraints fed to the CQM model, the only other optional parameter one can adjust is the `time_limit`, i.e., the maximum time the solver can use to perform the optimization. We examined the effect of increasing the CQM time limit parameter on typical batches of 70 or 140 payments. As shown in Figure 7 and Figure 8, in some cases there was an improvement in the aggregate mNDP of hundreds of thousands of euros in single batches.

In other cases, a batch is not optimizable (for example if in a batch there is a single payer such as on Day 33, batch 43 of batchsize 140, there can be no improvement even with increasing CQM `time_limit`). This suggests that the optimizability of the batches is a heterogeneous variable, as they are not equally optimizable even with extended solver time limits. See Section 4 for a more detailed explanation of this intuition, where we explore batch optimality and the characteristics of batches that influence total savings using the reordering approach.

[13] The maximum net debit position at a given time of the day is the maximum amount each financial institution needed before that moment to settle all the payments in which it was involved. The daily maximum of this quantity coincides with the minimum liquidity each institution has to lock in at the beginning of the day to avoid a default, and it is the quantity we want to optimize.

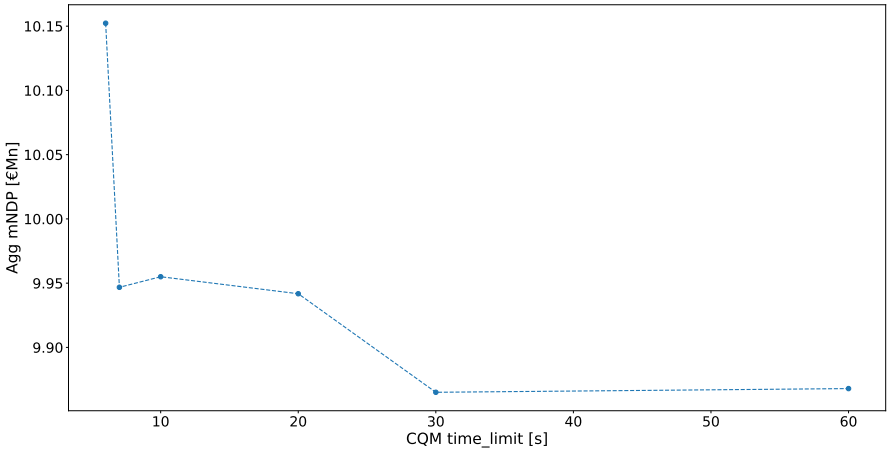

Figure 7: Aggregate mNDP as a function of CQM parameter `time_limit` for a single batch of 70 payments in TARGET2 data, Day 33 (batch 44).

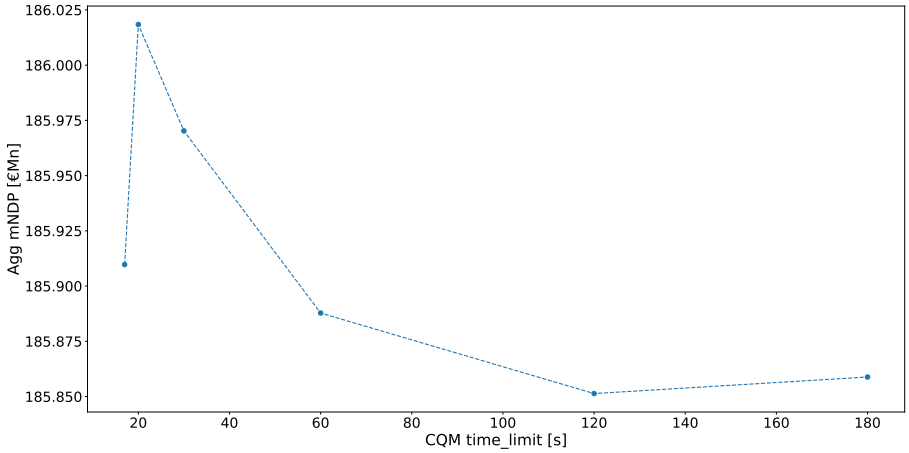

Figure 8: Aggregate mNDP as a function of CQM parameter `time_limit` for a single batch of 140 payments in the TARGET2 dataset, Day 33 (batch 88).

There is a third category: batches where the progression in performance as a function of CQM `time_limit` is less smooth (e.g. Figure 9). Due to the black-box nature of D-Wave's hybrid quantum solvers, we cannot know for sure, but only guess at why this may be the case - likely a combination of the prior scenario whereby optimizability is impaired for the specific batch of payments, and the effect of random noise of a certain scale combined with the probabilistic nature of the solver.

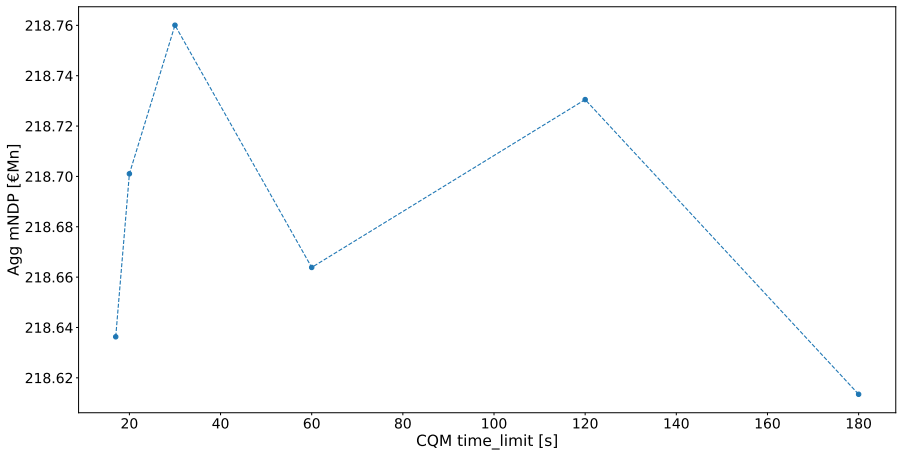

Figure 9: Aggregate mNDP as a function of CQM parameter `time_limit` for a single batch of 140 payments in the TARGET2 dataset, Day 33 (batch 87).

As shown in Tables 1 and 2, the CQM solver using the default time limit (5 seconds) is often achieving an optimal solution. However, we note a few days where the CQM solver lags behind the benchmark SAA optimization result - namely Days 20, 28 and 33 for batchsize 70. We thus ran those days using an increased CQM `time_limit` of 30s for batchsize 70, to see if this would close the gap between the SAA and CQM solver results. Indeed, as we see in Table 1, the asterisked values, which indicate those obtained by running with the time limit increased to 30s, effectively removes any gap in performance between the SAA and CQM solvers.

Remaining gaps in liquidity improvement between the maximum theoretical bound (from multilateral netting) and what the solvers achieve are due to the quantized nature of the payments themselves. This is the difference between payment reordering, but where payments still must each be settled in the system one by one, and netting, where payments can be balanced digitally and in a sense perfectly, since each individual payment need not be performed through the system, but which would introduces more challenges from both legal and risk management perspectives.

4 Batch Optimizability Study

In this section, our exploration focuses on the batches within our sample, aiming to pin down the essential attributes of these batches pertinent to the re-ordering problem. A binary classification framework is implemented to categorize batches into either optimized (class 1) or non-optimized (class 0). Optimized batches are characterized by instances where the re-ordering algorithm yields liquidity savings compared to the original FIFO order, while non-optimized batches refer to those where such savings are not observed.

The schematic of the three steps used for the batch optimizability study presented here are outlined in Figure 10. The process starts with looking at batches across each day in our sample. For each batch, we extract a target variable which is binary, i.e, 1 if current batch processed in order proposed by CQM saved some liquidity compared to FIFO otherwise 0. Next, for each payment batch, we

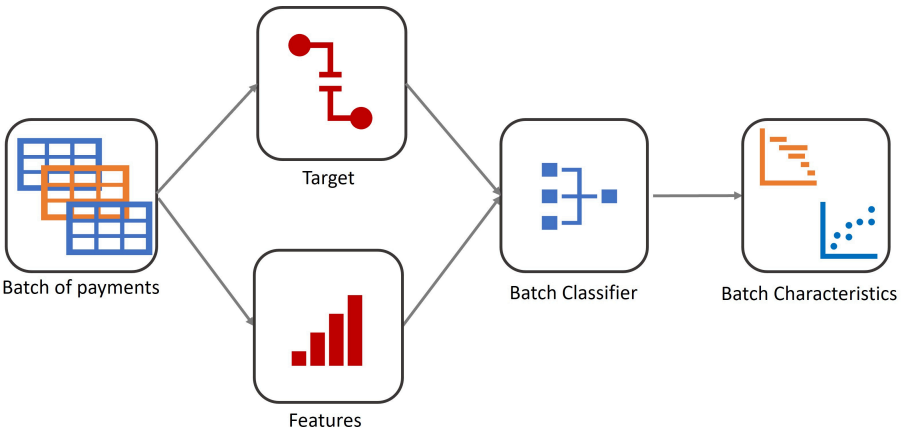

Figure 10: This schematic represents the sequential stages of data analysis in this batch optimization study.

extract a set of features categorized into five distinct groups:

- Liquidity Variables: These include 'saved amount' and 'lossed amount', which quantify the amount saved or lost by employing the CQM solver as opposed to First In, First Out (FIFO), measured in terms of Aggregate Maximum Net Debt Position (AggMaxNDP).
- Participant Variables: This group captures the number of distinct entities involved in transactions, with 'unique senders' and 'unique receivers' accounting for individual payers and payees, respectively, and 'both as sender and receiver' identifying entities that act in both capacities within the batch.
- Amount Variables: These are financial metrics of the batch, comprising 'amount total' for the aggregate sum, 'amount mean' for the average transaction value, 'amount max' and 'amount med' for the largest and median values, and 'amount var' for the variability in transaction amounts.
- Graph Variables: Reflecting the network structure of the batch, this set includes 'batch degree' for the number of connections per node, ''batch centrality' for the significance of nodes within the network (how much the node is a frequent transit point in the network traffic), 'batch clustering' for the degree to which nodes tend to cluster together (how much the neighbors of a node are also connected to each other), and 'batch reciprocity' for the proportion of bidirectional transactions.
- Time Variables: total time to collect a batch' represents the cumulative duration of the payment batch, which can be further categorized into 'short batch', 'medium batch', and 'long batch' to provide temporal context to the batch processing time.

These variables offer a multifaceted view of the payment batch, allowing for detailed analysis and optimization of the settlement process. Subsequently, using the extracted batch features and target variables, we train a classification model. This model can be mathematically represented as follows: Let $X = \mathbf{x}_i^1, \mathbf{x}_i^2, \ldots, \mathbf{x}_i^M$ denote a collection of M predictors (or features) each with N data points or samples, while $\mathbf{y}$ represents the target variable or dependent variable, also encompassing N data points or samples. In the context of our study, $\mathbf{y}_i = 0, 1$, where 1 signifies the category of optimized batches, and 0 signifies that of non optimized batches. We denote with $\hat{\mathbf{y}}$ the predicted target variable.

This prediction can be estimated using a classification model denoted as f, which can be expressed as $\hat{\mathbf{y}} = f(X)$.

A commonly used econometric model for classification is logistic regression, which analyzes the relationship between the target variable and predictor variables. Logistic regression is valued for its simplicity and interpretability. However, its performance can be limited with large and complex datasets, particularly when dealing with many continuous and categorical predictors, as well as nonlinearity (Hastie et al. 2009). Machine learning (ML) algorithms can model complex, non-linear relationships between features and the target variable, which traditional models like logistic regression may struggle with. Therefore, we employ decision-tree-based ensemble learning algorithms, specifically LightGBM (Ke et al. 2017), an advanced version of the standard gradient boosting model.

To train the ML model, we use standard K-fold cross-validation with 10 folds (see Figure 11). Cross-validation helps assess the model's performance by partitioning the dataset into training and validation sets multiple times, ensuring robustness and generalizability. The model that performs best across all folds is chosen. The selected model achieves an 89% accuracy in terms of R^2-scores, about 5% higher than the logistic regression model.

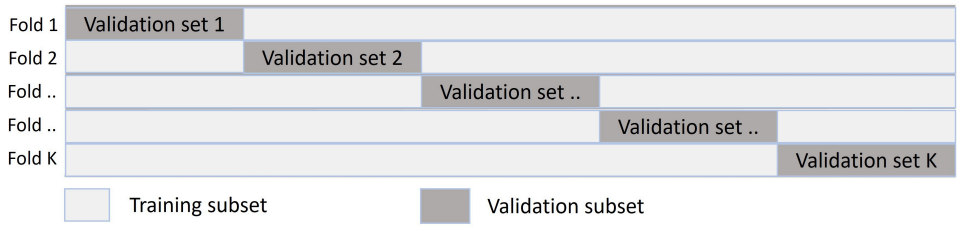

Figure 11: A schematic overview of a K-fold cross-validation procedure for the training and tuning of the classifier model. The dataset is divided into a K training (highlighted in light gray) and validation subsets (highlighted in dark gray).

In the last step, we utilize the classification model to conduct a feature importance analysis. This entails a relative ranking of the features based on their significance from the model's perspective during the classification process. By undertaking this analysis, we aim to understand the relative roles of different features in the model's decision-making process. To accomplish this, we utilize the SHAP (SHapley Additive exPlanations) tool (Lundberg et al. 2020), which is grounded in the concept of Shapley values from game theory (Shapley 1953).

The SHAP approach allows us to examine how the characteristics of payment batches influence liquidity savings. SHAP values, visualized in dependence plots or feature importance plots, explain how specific payment features on the model's predictions, providing an interpretable and robust measure of feature importance. It's important to note that the SHAP approach does not provide causal inference or any optimal statistical criterion. Instead, it explains the marginal contribution of each transaction feature to the difference between the model's predictions and the average prediction on the entire training sample. Therefore, its primary use is in intuitively explaining the models' predictions and feature importance (Lundberg et al. 2020).

In Figure 12 each bar represents a batch feature, and the length of the bar indicates the mean absolute SHAP value for that feature. The mean absolute SHAP value quantifies the average impact of a feature on the model's prediction. The features at the top, *both as sender receiver* and *unique receiver*, has the largest mean absolute SHAP value, which suggests they have the greatest impact on the model's output. In other words, this feature typically influences the model's predictions the most. As we move down the y-axis, the features have a progressively smaller impact on model output, with

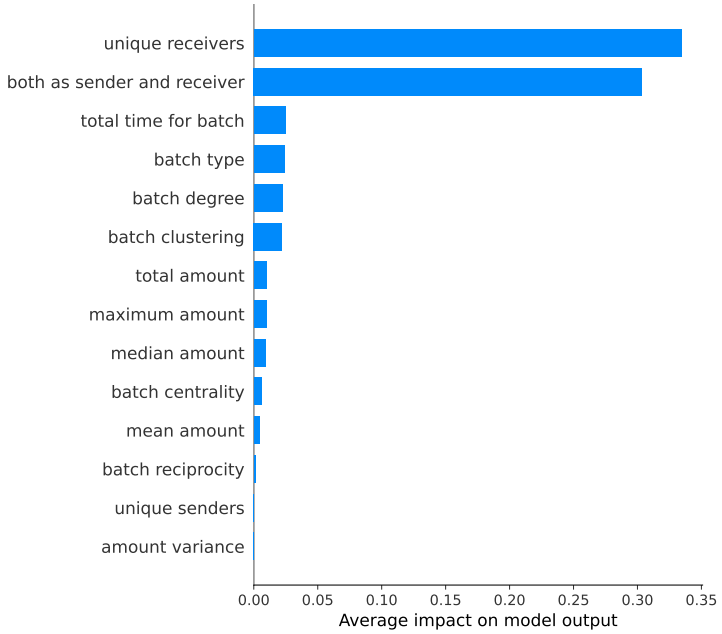

Figure 12: Features importance study using LightGBM model and SHAP. A higher mean SHAP value indicates a greater impact of that feature on the model's prediction.

amount variance at the bottom having the least impact among the shown features.

Figure 13 to 15 illustrates the relationship between the feature's value and its SHAP value on model output. First, we examine the import features, *both as sender and receiver* in Figure 13. The spread of dots vertically at each integer value of this variable indicates the range of SHAP values (or impact) this feature has at that specific value. From the plot we can infer that when a lower number of participants act as both sender and receiver of transactions within a batch of payments, this tends to produce negative SHAP values (as seen by the cluster of blue dots), indicating a lower or negative impact on the model's prediction. Conversely, as the number of participants acting as both sender and receiver in a batch increases, its impact becomes more positive (as seen by the cluster of pink/red dots). This type of plot can be helpful in understanding how the presence or value of one feature affects the contribution of another feature to the model's predictions. It's a valuable tool for interpreting complex models and for explaining why a model makes certain decisions. Likewise, in Figure 14, we show the influence of the number of *unique receivers* feature on a model's predictions. As the number of unique receivers in the batch increases, the SHAP value also tends to increase, indicating a positive relationship with the model's output. The variation in SHAP values at each level of this features suggests differing impacts on the model output across different observations.

In Figure 15, we show the influence of the feature *total time to collect a batch* of payments on the model's predictions. The plot suggests that when the total time to collect 70 payments is small (indicating a high pace of incoming payments), the SHAP value is low, indicating less opportunity to optimize that batch. However, as the collection time increases, the SHAP values suddenly become positive, indicating greater optimization potential.

To summarize, this study identifies and analyzes various batch features that impact the model's predictions. Notably, features such as *both as sender and receiver* and *unique receivers* significantly

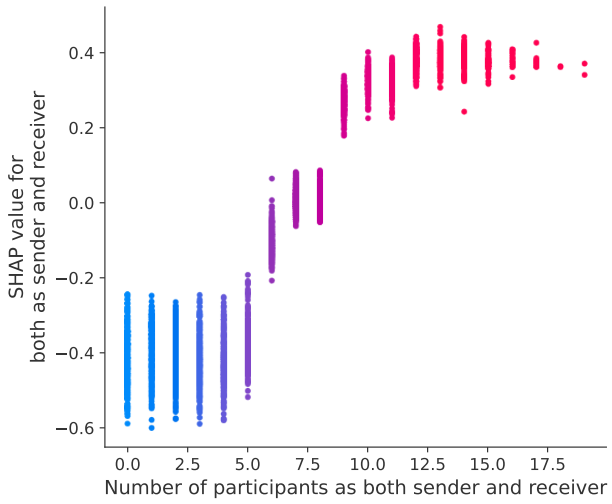

Figure 13: The SHAP dependence plot illustrates the impact of the feature "number of participants as both sender and receiver in a batch" on the model's predictions. Color intensity indicates the magnitude of SHAP values, with positive SHAP values corresponding to class-1 (optimized) and negative SHAP values corresponding to class-0 (not optimized).

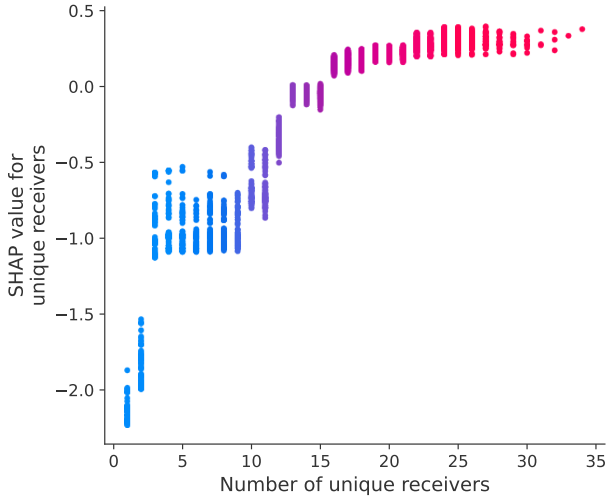

Figure 14: The SHAP dependence plot shows the impact of the feature "number of unique receivers in a batch" on the model's predictions. Color intensity indicates the magnitude of SHAP values, with positive SHAP values corresponding to class-1 (optimized) and negative SHAP values corresponding to class-0 (not optimized).

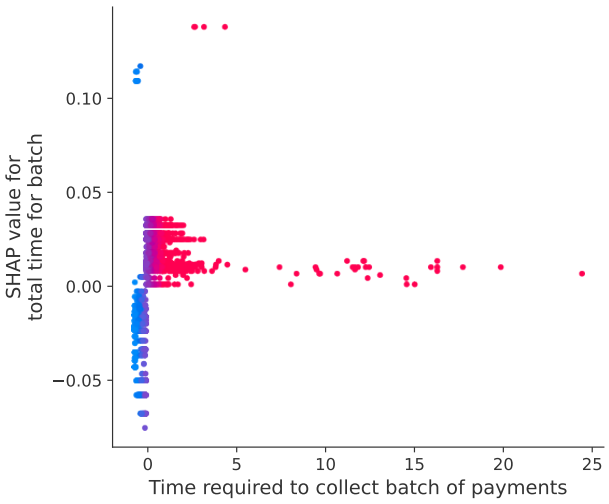

Figure 15: The SHAP dependence plot shows the impact of the feature "time required to collect the batch" of 70 payments on the model's predictions. Color intensity indicates the magnitude of SHAP values, with positive SHAP values corresponding to class-1 (optimized) and negative SHAP values corresponding to class-0 (not optimized).

influence the model outcomes, with the former having the greatest overall impact. SHAP value analysis reveals that a higher number of entities acting as both senders and receivers within a batch generally leads to positive liquidity savings, while lower numbers have a less favorable or even negative effect. The analysis uses a machine learning framework to categorize and examine these features, providing insights into the factors that affect liquidity savings in payment processing. The dependence plots created using SHAP values illustrate the intricate relationships between individual features and the model's performance, aiding in the interpretation of the predictive model.

A refinement of the optimization algorithm could include the adoption of a flexible batch optimization mechanism, whereby the algorithm dynamically evaluates the characteristics of each incoming batch to determine its potential for beneficial optimization. The algorithm would selectively engage the optimization process only for those batches exhibiting a high estimated likelihood of improvement. Such an adaptive strategy would enhance overall computational efficiency by focusing resources on instances where optimization is expected to yield meaningful gains, thereby avoiding the overhead associated with unproductive optimization attempts.

5 Benchmarking using Simulated Annealing

To provide a meaningful comparison for the results obtained with the CQM optimization, we consider here two different benchmarks. First, any batch-wise payment reordering cannot produce an arbitrary mNDP optimization: the debt position of each institution at the end of each batch is obviously permutation invariant, hence the least stringent liquidity constraint coincides with the one granting a non-negative liquidity at the end of each batch.[14] Any less liquidity set aside at

[14]The mNDP for an individual participant can vary depending on the order in which payments are processed within the batch. However, by the end of the batch, the final net debit positions remain invariant of the order.

the beginning of the day implies a default in at least one of the batches.[15] The liquidity constraint obtained in this way coincides with what would be obtained optimizing the payments resolution with the so-called *deferred batch multilateral netting*. With such netting all the payments in a batch are collected, and then netted for each single institution. After this netting, the only transactions finalized at the end of the batch are the ones in which the net amount is different than zero. In particular, the single original payments of the batch are not processed, and are substituted with new, synthetic transactions representing the net amounts. This feature of the algorithm renders it not suitable for Real-Time Gross Settlement systems like TARGET2, both for legal and risk-management considerations.

The upper bound obtained using such multilateral netting of the daily optimizability of mNDP could be very far from mNDP gains attainable with any optimization scheme based on payments reordering. For this reason we secondly perform an optimization study with a classical heuristic solver. Given the high dimension of the parameter space and the similarity with the quantum annealing, we used a Simulated Annealing Algorithm (SAA) (Kirkpatrick et al. 1983; Aarts et al. 2005).

Quantum annealing exploits the so-called quantum adiabatic theorem (Born and Fock 1928; Kato 1950) to find the ground state of a given Hamiltonian starting from an arbitrary starting state. This result is achieved by changing the Hamiltonian parameters slowly enough to allow the system to remain in its instantaneous ground state. In the SAA, on the other hand, the system starts with an arbitrary energy value, which is then lowered by a sequence of Markov chain Monte Carlo (MCMC) steps. While in the quantum annealing the slowly changing parameter is the Hamiltonian of the system, in the simulated annealing it is the acceptance probability of the MCMC proposals. The acceptance probability is equal to one if the system energy is decreased in the given MCMC step, but to allow the system to efficiently explore the state space a positive probability is given to transitions increasing the energy value. The acceptance probability for these transitions can be written as:

$$p\left(E_{\text{in}}, E_{\text{fin}}, T\right) = \exp \frac{\left(E_{\text{in}} - E_{\text{fin}}\right)}{T} \quad , \quad E_{\text{fin}} > E_{\text{in}} \tag{5}$$

where E_{in} and E_{fin} are the initial and final energies of the system, and T is a parameter gauging the freedom of the system to explore the search space (in analogy with physical annealing it is usually identified with a temperature). While in a standard MCMC algorithm the parameter T would be a constant throughout the optimization process, in simulated annealing it is slowly decreased, to enforce a greater bias toward energy-decreasing transitions near the end of the optimization process (in the limit $T \to 0$ only energy-decreasing transitions are accepted). In our implementation of the algorithm the temperature decrease at each iteration step is itself dependent on the initial temperature. The temperature as a function of the iteration step is given by:

$$T_i = T_0^{1-\frac{i}{n}} \tag{6}$$

[15] To prove the existence of this upper bound, we can consider the following fact: the liquidity constraint is imposed continuously in time (each payer must have sufficient funds to satisfy the constraint every time a payment is regulated). We can imagine a similar constraint, applied only in a discrete subset of times (i.e. each payer must have sufficient liquidity to satisfy its debt positions as evaluated hourly). Obviously the latter constraint is weaker than the original one, hence the liquidity necessary to comply with it is less than or equal to the liquidity necessary to meet the original constraint. Finally, we can select the times of enforcement of the weaker constraint to be the intervals between the conclusion of a batch and the beginning of the next one. The liquidity necessary to meet the constraint thus defined is independent of the payment order inside each batch, therefore this amount is smaller than or equal to the one needed to satisfy the continuous liquidity constraint with any payment order.

where T_i is the temperature at step i, T_0 is the initial temperature and n the total number of steps in the algorithm. The pseudo-code is given in algorithm 1.

Algorithm 1: Simulated Annealing Algorithm (SAA)

Input : Ordered set of payments B, initial temperature T_0, number of steps n
Output: Optimized set of payments B_{opt}, optimized maximum debt $mNDP_{\text{opt}}$

$mNDP_{\text{opt}} = mNDP(B)$;
$B_{\text{opt}} = B$;

for $i = 1$ **to** n **do**

  $B_{\text{temp}} =$ "small" permutation of B_{opt};
  $mNDP_{\text{temp}} = mNDP(B_{\text{temp}})$;
  $T_i = T_0^{1-\frac{i}{n}}$;
  $\Delta = mNDP_{\text{opt}} - mNDP_{\text{temp}}$;
  $P = \exp -\frac{\Delta}{T_i}$;
  $X =$ Random number with uniform distribution in $[0, 1]$;

  if $X < P$ **then**

    $mNDP_{\text{opt}} = mNDP_{\text{temp}}$;
    $B_{\text{opt}} = B_{\text{temp}}$;

return B_{opt}, $mNDP_{opt}$;

In this code the "small" permutation represents the MCMC step; while permutation spaces are not equipped with a natural distance, we define "small" a permutation of few elements with respect to the total number of payments in a batch. In particular, for batch size equal to 70, we select random permutations of 10 elements for each iteration step. Finally, we select the total number of iteration n to be equal to 200 in order to maintain the execution time short enough to be comparable with the CQM runtime. While an exact classical analogue of quantum annealing is of course impossible, SAA is close in terms of physical interpretation. Quantum fluctuations, responsible for the exploration of the parameter space in the CQM, are substituted by "thermal" fluctuations in the SAA. Both algorithms were first developed with an application to hard combinatorial problems in mind, and quantum fluctuations were explicitly introduced as an enhancement of the SAA in (Kadowaki and Nishimori 1998). Some indications of the theoretical superiority of quantum annealing with respect to classical optimization algorithms such as SAA exist in the literature (Wang et al. 2016; Bernaschi et al. 2024), further motivating our comparison.

The results of the SAA[16] along with quantum annealing using CQM solver are discussed below. First, we present a table for the batch of 70 payments, comparing the liquidity savings achieved by the SAA with those achieved by the CQM solver. Our results show that with a sufficient number of samples, the simulated annealing algorithm can achieve a similar level of performance within the same time constraints. Both CQM and SAA achieve about EUR 23 million in savings, as shown in Table 1. This amount is slightly smaller than the upper bound, defined at the beginning of the section.

[16]The SAA algorithm is run with Python 3.7 on an Intel Xeon E5 CPU.

Likewise, we present the end-of-day liquidity savings using SAA and CQM for batch size equal to 140 in Table 2: in this case the average saving amounts to EUR 38 million. In addition, the SAA can be scaled to even larger batch sizes. In table Table 3 the results are shown for batch size equal to 700, and in Appendix A the results for batch sizes from 200 to 600 are shown.[17] These tables highlight the consistently higher performance of the SAA and CQM over FIFO order across various batch sizes for each day in our sample. The increasing savings with larger batch sizes underscore the effectiveness of SAA in leveraging the additional reordering opportunities provided by larger sets of transactions. The summary of these results is discussed in Figure 16, which proves that as the batch size increases, the average savings across the days in our sample increase.

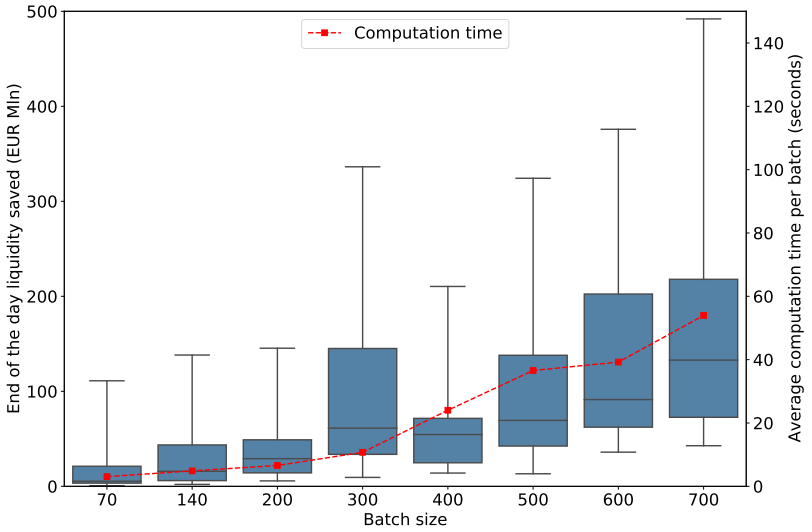

Figure 16: Boxplot statistics of total liquidity in EUR millions saved using SAA compared to FIFO with increasing batch size (left scale), and average computation time per batch in seconds (right scale). The boxes mark the 25^{th} and 75^{th} percentiles and the whiskers represent the 5^{th} and 95^{th} percentiles of the distributions.

These findings are significant as they illustrate the effectiveness of both SAA and CQM for the optimization of the daily liquidity constraints, and the scalability and robustness of the SAA approach in handling increasingly large payment batches. This scalability is crucial for practical applications in HVPS, where batch sizes can vary significantly, and efficient liquidity management is essential for financial stability and operational efficiency. The results validate the potential of SAA as a viable and efficient alternative to the hybrid quantum approach, which currently has limitations due to hardware constraints. However, as quantum hardware continues to advance, it is expected to significantly improve solution quality and processing times, making it a valuable long-term solution in this and other fields requiring high computational efficiency.

[17]For batch sizes from 200 to 700 the number of MCMC steps are selected as equal to the batch size, and the permutations swap one-tenth of the elements in the batches.

Table 1: Comparison of total liquidity saved by different solvers for batch size of 70, with an estimated upper bound (not necessarily attainable). Values for the CQM solver use the deafult solver time limit parameter (5 s), except in the cases of the values with an asterisk, which used a set time limit of 30 s (Section 3.1).

Date	SAA (€Mn)	CQM (€Mn)	Upper bound (€Mn)
Day 1	0.61	0.61	0.61
Day 2	6.28	6.27	6.28
Day 3	4.15	4.15	4.15
Day 4	150.95	150.95	158.67
Day 5	4.26	4.26	4.26
Day 6	3.95	3.94	3.95
Day 7	24.89	24.88	24.89
Day 8	5.33	5.33	5.33
Day 9	2.59	2.59	2.59
Day 10	3.88	3.88	3.88
Day 11	1.66	1.64	1.66
Day 12	33.08	33.08	33.08
Day 13	5.05	5.05	5.05
Day 14	40.82	40.82	40.82
Day 15	1.07	1.07	1.07
Day 16	0.41	0.40	0.41
Day 17	19.63	19.63	19.63
Day 18	8.69	8.68	8.69
Day 19	25.08	25.07	25.08
Day 20	10.68	9.67, 10.68*	10.68
Day 21	111.06	111.04	111.06
Day 22	0.30	0.30	0.30
Day 23	0.62	0.62	0.62
Day 24	22.63	22.63	22.63
Day 25	4.21	4.19	4.21
Day 26	10.88	10.88	10.88
Day 27	8.89	8.87	8.89
Day 28	159.02	158.94, 159.02*	159.02
Day 29	0.62	0.62	0.62
Day 30	7.05	7.04	7.05
Day 31	3.67	3.67	3.67
Day 32	2.98	2.98	2.98
Day 33	111.08	111.02, 111.07*	197.89
Day 34	4.57	4.56	4.57
Day 35	12.91	12.91	12.91
Average	**23.22**	**23.16, 23.22***	**25.92**

Table 2: Comparison of total liquidity saved by different solvers for batch size 140, with an estimated upper bound (not necessarily attainable).

Date	SAA (€Mn)	CQM (€Mn)	Upper bound (€Mn)
Day 1	1.94	1.93	1.94
Day 2	6.81	6.81	6.81
Day 3	9.99	9.99	9.99
Day 4	166.72	166.72	166.72
Day 5	15.84	15.69	15.84
Day 6	5.23	5.22	5.23
Day 7	140.18	140.18	140.18
Day 8	12.64	12.64	12.64
Day 9	4.03	4.03	4.03
Day 10	15.93	15.93	15.93
Day 11	6.8	6.61	6.8
Day 12	48.29	48.29	48.29
Day 13	63.96	63.96	63.96
Day 14	41.43	41.43	41.43
Day 15	2.04	2.04	2.04
Day 16	3.74	3.65	3.74
Day 17	32.51	32.49	32.51
Day 18	12.58	12.57	12.58
Day 19	27.64	27.58	27.64
Day 20	27.25	27.24	28.75
Day 21	114.01	114.01	114.01
Day 22	10.31	10.3	10.31
Day 23	1.27	1.27	1.27
Day 24	45.15	45.15	45.15
Day 25	41.96	41.95	41.96
Day 26	11.6	11.6	11.6
Day 27	16.16	16.16	16.16
Day 28	167.7	167.7	167.7
Day 29	3.39	3.39	3.39
Day 30	10.04	10.04	10.04
Day 31	4.72	4.71	4.72
Day 32	3.17	3.17	3.17
Day 33	111.69	111.51	198.5
Day 34	138.18	138.18	138.18
Day 35	18.28	18.25	18.28
Average	**38.38**	**38.35**	**40.9**

Table 3: Optimization results for batch size 700, with an estimated upper bound (not necessarily attainable).

Date	SAA (€Mn)	Upper bound (€Mn)
Day 1	13.96	13.97
Day 2	46.02	46.02
Day 3	96.86	103.25
Day 4	199.31	199.36
Day 5	58.69	58.71
Day 6	161.6	161.6
Day 7	353.7	353.74
Day 8	64.78	64.78
Day 9	734.89	734.89
Day 10	219.58	219.58
Day 11	338.67	338.67
Day 12	83.83	83.83
Day 13	208.28	208.28
Day 14	91.44	91.89
Day 15	73.89	73.89
Day 16	71.42	71.47
Day 17	202.37	202.47
Day 18	216.19	216.22
Day 19	74.59	74.59
Day 20	132.86	132.86
Day 21	152.19	155.86
Day 22	34.23	34.23
Day 23	59.26	59.26
Day 24	149.65	149.65
Day 25	74.99	74.99
Day 26	42.73	42.99
Day 27	59.27	62.81
Day 28	220.8	220.8
Day 29	148.16	148.17
Day 30	318.16	318.16
Day 31	90.44	90.47
Day 32	491.98	491.98
Day 33	3736.43	3859.76
Day 34	318.77	318.77
Day 35	85.53	85.86
Average	**269.30**	**273.25**

6 Optimization results

In this section we show the optimization results, with some examples of batch reorderings for batches of dimension 70. In Table 1, Table 2 and Table 3 we show the numerical results of the optimization process for batches of sizes 70 and 140 (both for the CQM and the SAA optimization) and size 700 (only for the SAA optimization). Results for intermediate batch sizes can be found in appendix Appendix A. Average daily savings are, respectively, 23, 38 and 269 million euro, following the increase in batch size (and also the time to accumulate a sufficient number of payments). In relative terms these values translate to 0.4%, 0.7% and 3.7% of the total daily liquidity needs of the institutions involved. All these values are close to the upper bound of the savings, which would have been obtained with the multilateral netting process.

For the illustrations of batch optimizations, we selected the batch causing the greatest discrepancy between the optimization results and the theoretical upper bound in day 33 ("unsuccessful" batch), and the batch from day 28 showing the greatest gains in terms of liquidity optimization ("successful" batch). We show only payments above 10 million euro, for the sake of plot readability. In Figure 17 we see the time structure of payments for the original unoptimized FIFO queue of the former case. The mNDP is driven by mainly four payments: 45 to 92, 96 to 26, 63 to 98 and 26 to 96. The last payment, from 26 to 96, decreases the debt position of entity 96 at the end of the batch, and hence the aggregate final debt position of the group. This change, however, leaves the mNDP constant, given the previous liquidity needs of entity 96. The gap between these two quantities is transferred in the distance between the optimization results and the theoretical upper bound given by the aggregate final debt position. In addition, this gap cannot be reduced, given that the difference is generated by two back-and-forth payments between two single entities. It is easy to see that any permutation of such pairs of payments leaves the mNDP constant, hence it cannot reduce the difference between maximum and final aggregate debt positions. In Figure 18 we see the payments order after the optimization process. As anticipated, the order of payments between 26 and 96 (the only pair of entities having a back-and-forth exchange) is unchanged, being neutral in terms of mNDP value.

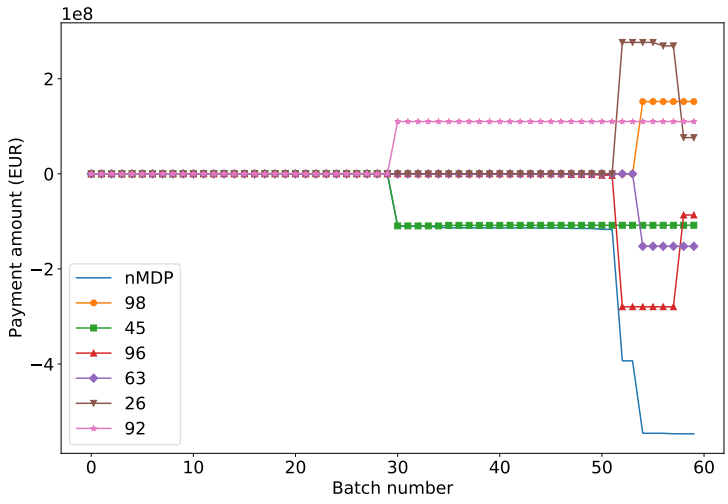

Figure 17: Time structure of payments with FIFO order for an "unsuccesful" batch.

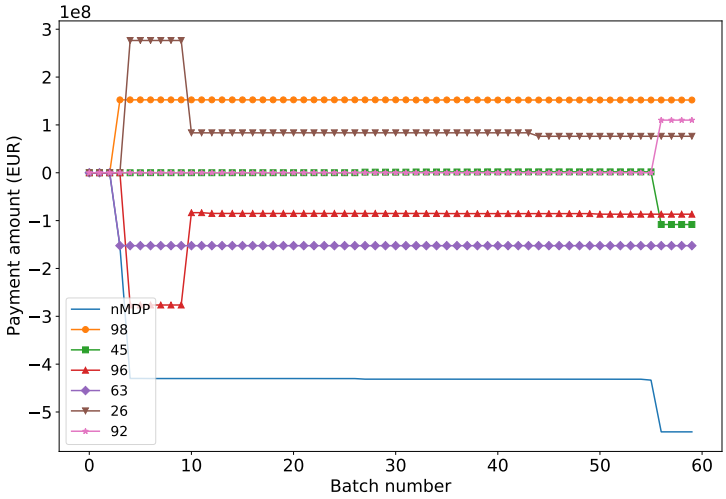

Figure 18: Time structure of payments with optimized order for an "unsuccesful" batch.

On the other hand, in Figure 19 we see the time structure of payments for a batch with a high degree of optimizability. In this case the mNDP trajectory is mainly driven by the payments from 45 to 96 and from 92 to 45. In particular the first two payments establish a debt position for entity 45, which is later converted into a positive position through a payment received from entity 92, which in turn goes into debt. This implies that the bulk of the mNDP is constituted by the sum of the (temporary) debt position of entity 45 and the final debt position of entity 92. This structure can be optimized as in Figure 20: if the first payment performed is the one from 92 to 45, the latter entity doesn't need any additional liquidity, and never reaches a debt position. Hence the mNDP is now composed mainly of the debt position of entity 92 (toward the end of the batch we can notice a small contribution from entity 26, which was less apparent in the FIFO ordering).

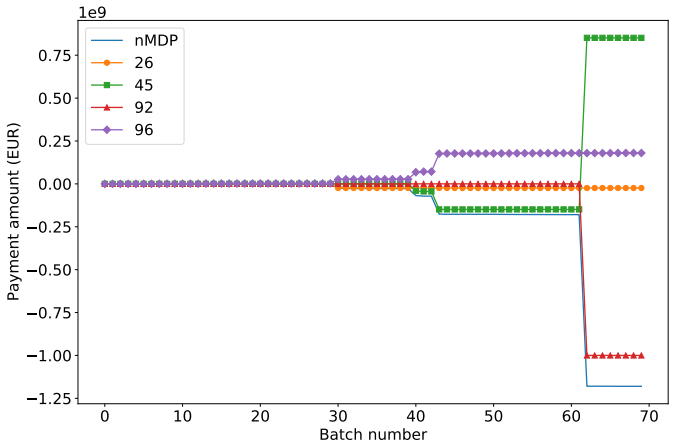

Figure 19: Time structure of payments with FIFO order for a "succesful" batch.

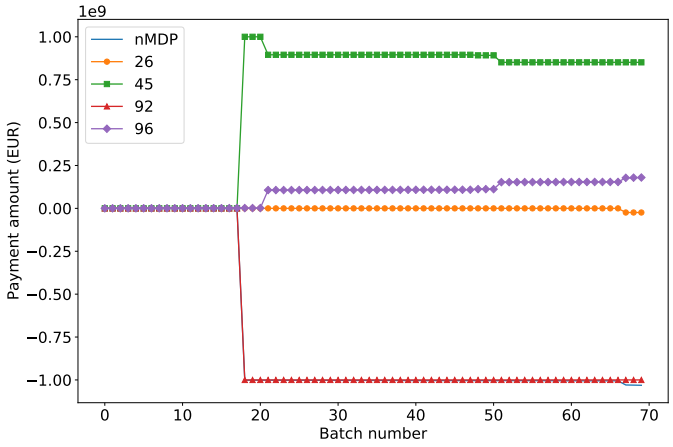

Figure 20: Time structure of payments with optimized order for a "succesful" batch.

7 Conclusion

In this paper we have examined the characteristics of the Italian segment of TARGET2, the HVPS operated by the Eurosystem until March 20, 2023. In particular we have explored two avenues to find an optimal value for the liquidity needs of TARGET2 participants. We showed the effectiveness of a hybrid quantum reordering solver and of a heuristic (classical) search algorithm for this application, comparing the features and limitations of the two approaches. Employing the CQM model available in the D-Wave software library, we achieved notable liquidity savings with a batch of 70 payments over a series of 35 consecutive days. These results are corroborated by the SAA, which provides a way to explore reorderings with batch sizes up to 700 payments, resulting in even more significant increases in liquidity savings (at the cost of greater waiting times). This is important given the current limitations of quantum hardware and the high cost of quantum computing. By using simulated annealing on classical computing systems as a scalable alternative, we enhance the practicality of the approach, making it a more feasible solution for improving liquidity efficiency in HVPSs in the near term. Finally, by employing a ML-based classification framework enhanced with Shapley values, we found the set of batch features having the biggest impact on the optimization possibilities. The most relevant variables for this purpose are the number of participants acting both as payers and payees, and the number of unique payees in the batch.

Future work could include searching for a more efficient Quadratic Unconstrained Binary Optimization (QUBO) formulation for the problem, investigating the impact of annealing features when running on the bare QPU, as well as exploring D-Wave's recently released Non-Linear Solver with updated QPU stacks and topologies (D-Wave Systems (2021)). These avenues could yield improved performance or even simply a reduction in computation time. Another possible enhancement of the algorithm could be flexible batch optimization, in which the algorithm takes as input the batch and decides whether it is optimizable, proceeding with the optimization only for batches having a high likelihood of profiting from it. If there is only a single payer/payee, for example, then it is not an optimizable batch, and we could save the time needed for the evaluation of an optimal ordering of payments. Otherwise, the solver is applied, using a time limit depending on the optimizability estimated for the given batch. Finally, our reordering algorithm could be integrated with other liquidity saving mechanisms, in order to provide both a buffer in cases of scarce liquidity and an increase in efficiency when abundant liquidity is available.

References

Aarts, E., J. Korst, and W. Michiels (2005). Simulated annealing. In E. Burke and G. Kendall (Eds.), *Search Methodologies*. Boston, MA: Springer.

Alexandrova, B., A. Badev, S. Benchimol Bastos, E. Benos, F. Cepeda-López, J. Chapman, M. Diehl, I. Duca-Radu, R. Garratt, R. Heijmans, et al. (2023). Intraday liquidity around the world. Technical report, Bank for International Settlements.

Bech, M. and R. Garratt (2003). The intraday liquidity management game. *Journal of Economic Theory 109*(2), 198–219. doi: 10.1016/S0022-0531(03)00016-4.

Benos, E., R. Garratt, and P. Zimmerman (2014). The role of counterparty risk in chaps following the collapse of lehman brothers. *Available at SSRN 2538237*.

Bernaschi, M., I. González-Adalid Pemartín, V. Martín-Mayor, and G. Parisi (2024). The quantum transition of the two-dimensional ising spin glass. *Nature*, 1–6.

Bestuzheva, K., M. Besançon, W.-K. Chen, A. Chmiela, T. Donkiewicz, J. van Doornmalen, L. Eifler, O. Gaul, G. Gamrath, A. Gleixner, L. Gottwald, C. Graczyk, K. Halbig, A. Hoen, C. Hojny, R. van der Hulst, T. Koch, M. Lübbecke, S. J. Maher, F. Matter, E. Mühmer, B. Müller, M. E. Pfetsch, D. Rehfeldt, S. Schlein, F. Schlösser, F. Serrano, Y. Shinano, B. Sofranac, M. Turner, S. Vigerske, F. Wegscheider, P. Wellner, D. Weninger, and J. Witzig (2021, December). The SCIP Optimization Suite 8.0. ZIB-Report 21-41, Zuse Institute Berlin.

Born, M. and V. Fock (1928). Beweis des adiabatensatzes. *Zeitschrift für Physik 51*(3), 165–180.

D-Wave Systems (2021). Hybrid solver for constrained quadratic models. Technical report, D-Wave Systems White Paper. https://www.dwavesys.com/media/rldh2ghw/14-1055a-a_hybrid_solver_for_constrained_quadratic_models.pdf.

Desai, A., Z. Lu, H. Rodrigo, J. Sharples, P. Tian, and N. Zhang (2023). From LVTS to Lynx: Quantitative assessment of payment system transition in Canada. *Journal of Payments Strategy & Systems 17*(3), 291–314.

Hastie, T., R. Tibshirani, J. H. Friedman, and J. H. Friedman (2009). *The elements of statistical learning: data mining, inference, and prediction*, Volume 2. Springer.

Kadowaki, T. and H. Nishimori (1998). Quantum annealing in the transverse ising model. *Physical Review E 58*(5), 5355–5363.

Kato, T. (1950). On the adiabatic theorem of quantum mechanics. *Journal of the Physical Society of Japan 5*(6), 435–439.

Ke, G., Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu (2017). LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pp. 3146–3154. doi: https://lightgbm.readthedocs.io/en/latest/.

Kirkpatrick, S., C. D. Gelatt Jr, and M. P. Vecchi (1983). Optimization by simulated annealing. *science 220*(4598), 671–680.

Lundberg, S. M., G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence 2*(1), 2522–5839. doi: 10.1038/s42256-019-0138-9.

Martin, A. and J. McAndrews (2008). Liquidity-saving mechanisms. *Journal of Monetary Economics 55*(3), 554–567.

McMahon, C., D. McGillivray, A. Desai, F. Rivadeneyra, J.-P. Lam, T. Lo, D. Marsden, and V. Skavysh (2024). Improving the efficiency of payments systems using quantum computing. *Management Science*. doi: https://doi.org/10.1287/mnsc.2023.00314.

Rivadeneyra, F. and N. Zhang (2022). Payment coordination and liquidity efficiency in the new Canadian wholesale payments system. Bank of Canada Staff Discussion Paper 2022-3, Bank of Canada.

Shapley, L. S. (1953). A value for n-person games. *Contributions to the Theory of Games 2*(28), 307–317.

Wang, C., H. Chen, and E. Jonckheere (2016). Quantum versus simulated annealing in wireless interference network optimization. *Scientific reports 6*(1), 25797.

A Optimization results using SAA for large batch sizes

Table 4: Optimization results for batch size 200, with an estimated upper bound (not necessarily attainable).

Date	SAA (€Mn)	Upper bound (€Mn)
Day 1	18.69	18.69
Day 2	17.77	17.78
Day 3	13.55	13.55
Day 4	187.99	188.47
Day 5	29.1	118.27
Day 6	14.76	14.76
Day 7	123.32	123.32
Day 8	23.58	23.58
Day 9	2.28	2.33
Day 10	28.0	28.0
Day 11	10.14	10.14
Day 12	47.41	47.41
Day 13	67.79	67.79
Day 14	75.75	75.75
Day 15	41.95	41.95
Day 16	13.15	13.15
Day 17	37.36	37.36
Day 18	50.47	50.48
Day 19	38.02	38.02
Day 20	44.09	45.6
Day 21	112.0	112.0
Day 22	12.32	12.32
Day 23	10.09	10.09
Day 24	47.36	47.36
Day 25	45.5	45.5
Day 26	24.44	24.46
Day 27	23.5	23.5
Day 28	175.28	175.28
Day 29	50.9	50.9
Day 30	29.35	29.35
Day 31	5.71	5.71
Day 32	4.28	4.28
Day 33	145.36	229.67
Day 34	9.75	9.75
Day 35	24.02	24.19
Average	**45.86**	**50.88**

Table 5: Optimization results for batch size 300, with an estimated upper bound (not necessarily attainable).

Date	SAA (€Mn)	Upper bound (€Mn)
Day 1	6.75	6.75
Day 2	19.05	19.05
Day 3	7.78	8.1
Day 4	189.56	190.84
Day 5	58.61	147.78
Day 6	77.6	77.6
Day 7	336.33	336.33
Day 8	35.53	35.55
Day 9	18.73	18.77
Day 10	188.63	188.63
Day 11	350.14	350.14
Day 12	68.97	68.97
Day 13	68.67	68.72
Day 14	79.44	79.44
Day 15	34.88	35.19
Day 16	54.46	54.47
Day 17	109.13	109.29
Day 18	120.1	120.17
Day 19	43.79	43.79
Day 20	73.58	73.58
Day 21	144.57	144.57
Day 22	13.8	13.8
Day 23	42.25	42.25
Day 24	57.17	57.17
Day 25	52.81	52.81
Day 26	20.26	20.27
Day 27	28.36	29.55
Day 28	188.92	193.99
Day 29	61.21	61.21
Day 30	171.36	171.39
Day 31	9.38	9.38
Day 32	488.99	488.99
Day 33	145.59	238.59
Day 34	294.74	294.74
Day 35	32.42	32.59
Average	**105.53**	**110.98**

Table 6: Optimization results for batch size 400, with an estimated upper bound (not necessarily attainable).

Date	SAA (€Mn)	Upper bound (€Mn)
Day 1	23.11	23.11
Day 2	22.76	22.76
Day 3	21.49	21.5
Day 4	210.36	210.62
Day 5	37.48	130.12
Day 6	16.99	16.99
Day 7	148.51	148.52
Day 8	55.6	55.6
Day 9	37.45	37.48
Day 10	58.34	58.34
Day 11	16.1	16.1
Day 12	69.88	69.88
Day 13	114.71	114.71
Day 14	79.84	79.85
Day 15	47.47	47.47
Day 16	26.43	26.45
Day 17	59.61	59.61
Day 18	54.5	54.5
Day 19	57.17	57.17
Day 20	73.19	74.71
Day 21	158.62	158.62
Day 22	14.21	14.25
Day 23	13.96	13.96
Day 24	57.13	57.13
Day 25	53.65	54.35
Day 26	62.43	62.43
Day 27	42.05	44.18
Day 28	201.76	210.24
Day 29	55.82	55.82
Day 30	47.89	47.93
Day 31	13.27	13.34
Day 32	8.96	9.0
Day 33	345.16	429.48
Day 34	268.66	268.66
Day 35	27.14	27.14
Average	**74.33**	**79.77**

Table 7: Optimization results for batch size 500, with an estimated upper bound (not necessarily attainable). Negative results like the one on Day 27 can be generated by processes like this: a payer P in a given batch has a FIFO debt position of X. After the optimization process, this debt is reduced to $X - d$, with a debt $d' < d$ transferred to a group of new debtors (thus reducing the mNDP for the batch of the quantity $d - d'$). If, however, in a following batch the payer P has an unoptimizable debt position $X' \geq X$, the net effect of the optimization is to increase the daily mNDP at least of the amount d'.

Date	SAA (€Mn)	Upper bound (€Mn)
Day 1	44.2	44.2
Day 2	26.73	26.73
Day 3	21.43	21.43
Day 4	211.25	213.66
Day 5	39.32	131.95
Day 6	90.19	90.19
Day 7	325.49	325.49
Day 8	51.65	51.65
Day 9	88.46	88.46
Day 10	215.62	215.62
Day 11	11.86	11.86
Day 12	69.39	69.39
Day 13	76.64	76.64
Day 14	47.86	47.86
Day 15	57.32	57.33
Day 16	48.8	49.29
Day 17	104.46	104.46
Day 18	131.01	131.01
Day 19	58.31	58.31
Day 20	103.52	103.52
Day 21	148.58	152.29
Day 22	40.57	40.58
Day 23	33.42	33.43
Day 24	127.15	127.15
Day 25	54.45	54.45
Day 26	57.02	57.02
Day 27	-21.32	65.96
Day 28	275.0	279.8
Day 29	95.53	95.57
Day 30	324.23	324.23
Day 31	13.21	13.21
Day 32	490.4	490.4
Day 33	144.85	231.67
Day 34	298.02	298.02
Day 35	28.45	35.88
Average	**112.37**	**120.53**

Table 8: Optimization results for batch size 600, with an estimated upper bound (not necessarily attainable).

Date	SAA (€Mn)	Upper bound (€Mn)
Day 1	35.96	35.96
Day 2	20.63	20.63
Day 3	103.91	109.72
Day 4	219.17	219.17
Day 5	66.9	156.09
Day 6	137.29	137.29
Day 7	361.87	361.87
Day 8	62.89	62.9
Day 9	79.61	79.61
Day 10	223.22	223.22
Day 11	380.23	380.23
Day 12	71.71	71.91
Day 13	105.59	105.59
Day 14	88.88	88.88
Day 15	91.3	91.3
Day 16	71.87	71.95
Day 17	135.04	135.04
Day 18	208.66	208.67
Day 19	61.6	61.6
Day 20	104.86	104.87
Day 21	172.89	172.89
Day 22	26.04	26.04
Day 23	53.86	53.87
Day 24	59.29	59.29
Day 25	67.33	67.33
Day 26	50.56	50.56
Day 27	53.76	62.77
Day 28	196.13	196.13
Day 29	109.34	109.34
Day 30	328.58	328.61
Day 31	76.64	76.64
Day 32	490.56	490.56
Day 33	375.81	499.42
Day 34	315.88	315.88
Day 35	38.19	38.51
Average	**144.17**	**150.70**

Recently published papers in the 'Markets, Infrastructures, Payment Systems' series

n. 41 Assessing credit risk sensitivity to climate and energy shocks, *by Stefano Di Virgilio, Ivan Faiella, Alessandro Mistretta and Simone Narizzano*

n. 42 Report on the payment attitudes of consumers in Italy: results from the ECB SPACE 2022 survey, *by Gabriele Coletti, Alberto Di Iorio, Emanuele Pimpini and Giorgia Rocco*

n. 43 A service architecture for an enhanced Cyber Threat Intelligence capability and its value for the cyber resilience of Financial Market Infrastructures, *by Giuseppe Amato, Simone Ciccarone, Pasquale Digregorio and Giuseppe Natalucci*

n. 44 Fine-tuning large language models for financial markets via ontological reasoning, *by Teodoro Baldazzi, Luigi Bellomarini, Stefano Ceri, Andrea Colombo, Andrea Gentili and Emanuel Sallinger*

n. 45 Sustainability at shareholder meetings in France, Germany and Italy, *by Tiziana De Stefano, Giuseppe Buscemi and Marco Fanari* (in Italian)

n. 46 Money market rate stabilization systems over the last 20 years: the role of the minimum reserve requirement, *by Patrizia Ceccacci, Barbara Mazzetta, Stefano Nobili, Filippo Perazzoli and Mattia Persico*

n. 47 Technology providers in the payment sector: market and regulatory developments, *by Emanuela Cerrato, Enrica Detto, Daniele Natalizi, Federico Semorile and Fabio Zuffranieri*

n. 48 The fundamental role of the repo market and central clearing, *by Cristina Di Luigi, Antonio Perrella and Alessio Ruggieri*

n. 49 From Public to Internal Capital Markets: The Effects of Affiliated IPOs on Group Firms, *by Luana Zaccaria, Simone Narizzano, Francesco Savino and Antonio Scalia*

n. 50 Byzantine Fault Tolerant consensus with confidential quorum certificate for a Central Bank DLT, *by Marco Benedetti, Francesco De Sclavis, Marco Favorito, Giuseppe Galano, Sara Giammusso, Antonio Muci and Matteo Nardelli*

n. 51 Environmental data and scores: lost in translation, *by Enrico Bernardini, Marco Fanari, Enrico Foscolo and Francesco Ruggiero*

n. 52 How important are ESG factors for banks' cost of debt? An empirical investigation, *by Stefano Nobili, Mattia Persico and Rosario Romeo*

n. 53 The Bank of Italy's statistical model for the credit assessment of non-financial firms, *by Simone Narizzano, Marco Orlandi and Antonio Scalia*

n. 54 The revision of PSD2 and the interplay with MiCAR in the rules governing payment services: evolution or revolution?, *by Mattia Suardi*

n. 55 Rating the Raters. A Central Bank Perspective, *by Francesco Columba, Federica Orsini and Stefano Tranquillo*

n. 56 A general framework to assess the smooth implementation of monetary policy: an application to the introduction of the digital euro, *by Annalisa De Nicola and Michelina Lo Russo*

n. 57 The German and Italian Government Bond Markets: The Role of Banks versus Non-Banks. A joint study by Banca d'Italia and Bundesbank, *by Puriya Abbassi, Michele Leonardo Bianchi, Daniela Della Gatta, Raffaele Gallo, Hanna Gohlke, Daniel Krause, Arianna Miglietta, Luca Moller, Jens Orben, Onofrio Panzarino, Dario Ruzzi, Willy Scherrieble and Michael Schmidt*

n. 58 Chat Bankman-Fried? An Exploration of LLM Alignment in Finance, *by Claudia Biancotti, Carolina Camassa, Andrea Coletta, Oliver Giudice and Aldo Glielmo*

n. 59 Modelling transition risk-adjusted probability of default, *by Manuel Cugliari, Alessandra Iannamorelli and Federica Vassalli*

n. 60 The use of Banca d'Italia's credit assessment system for Italian non-financial firms within the Eurosystem's collateral framework, *by Stefano Di Virgilio, Alessandra Iannamorelli, Francesco Monterisi and Simone Narizzano*

n. 61 Fintech Classification Methodology, *by Alessandro Lentini, Daniela Elena Munteanu and Fabrizio Zennaro*

n. 62 The Rise of Climate Risks: Evidence from Expected Default Frequencies for Firms, *by Matilde Faralli and Francesco Ruggiero*

n. 63 Exploratory survey of the Italian market for cybersecurity testing services, *by Anna Barcheri, Luca Bastianelli, Tommaso Curcio, Luca De Angelis, Paolo De Joannon, Gianluca Ralli and Diego Ruggeri*

n. 64 A practical implementation of a quantum-safe PKI in a payment systems environment, *by Luca Buccella and Stefano Massi*

n. 65 Stewardship Policies. A Survey of the Main Issues, *by Marco Fanari, Enrico Bernardini, Elisabetta Cecchet, Francesco Columba, Johnny Di Giampaolo, Gabriele Fraboni, Donatella La Licata, Simone Letta, Gianluca Mango and Roberta Occhilupo*

n. 66 Is there an equity greenium in the euro area?, *by Marco Fanari, Marianna Caccavaio, Davide Di Zio, Simone Letta and Ciriaco Milano*

n. 67 Open Banking in Italy: A Comprehensive Report, *by Carlo Cafarotti and Ravenio Parrini*

n. 68 Report on the payment attitudes of consumers in Italy: results from ECB SPACE 2024 survey, *by Gabriele Coletti, Marialucia Longo, Laura Painelli, Emanuele Pimpini and Giorgia Rocco*

n. 69 A solution for cross-border and cross-currency interoperability of instant payment systems, *by Domenico Di Giulio, Vitangelo Lasorella, Pietro Tiberi*

n. 70 Do firms care about climate change risks? Survey evidence from Italy, *by Francesca Colletti, Francesco Columba, Manuel Cugliari, Alessandra Iannamorelli, Paolo Parlamento and Laura Tozzi*

n. 71 Demand and supply of Italian government bonds during the exit from expansionary monetary policy, *by Fabio Capasso, Francesco Musto, Michele Pagano, Onofrio Panzarino, Alfonso Puorro and Vittorio Siracusa*

n. 72 Statistics on tokenized financial instruments: A challenge for central banks, *by Riccardo Colantonio, Massimo Coletta, Riccardo Renzi*

n. 73 Credit Risk Assessment with Stacked Machine Learning, *by Francesco Columba, Manuel Cugliari, Stefano Di Virgilio*

n. 74 What if Ether Goes to Zero? How Market Risk Becomes Infrastructure Risk in Crypto, *by Claudia Biancotti*

n. 75 The Cyber Risk of Non-Financial Firms, *by Francesco Columba, Manuel Cugliari, Marco Orlandi, Federica Vassalli*

n. 76 Sustainability and financial innovation: The emerging role of Fintech for Good (F4G), *by Alessandro Lentini and Daniela Elena Munteanu*

n. 77 Hydrogeological and credit risk: the Italian firms' physical risk-adjusted probability of default, *by Manuel Cugliari, Simone Narizzano and Federica Vassalli*