Liquidity-Saving Mechanisms for 24/7 Instant Payment Systems: Rolling Liquidity Bounds, Certified Offsetting, and an Open Benchmark

Ethan Li
Independent Researcher
poggies55555@gmail.com

Abstract—Liquidity-saving mechanisms (LSMs) let interbank payment systems settle with less intraday cash by queueing payments and settling offsetting sets together. Most analyses assume the system closes each day, and the few that study 24/7 settlement are simulations. Instant-payment systems never close and promise settlement within seconds. We give the first analytical bounds for this setting. We prove a liquidity floor that no settlement engine can beat when every payment must settle within a delay cap T; it equals the classic gross-settlement bound at $T = 0$. We identify classes where the floor is attained and give a two-bank instance where it is not. We define rolling liquidity bounds for systems with no end of day and prove they are ordered, monotone and consistent with daily bounds. We prove that settling a batch at once, netted, never needs more liquidity than any ordering of it; this certifies reordering heuristics, including those used in recent quantum-annealing studies. In simulations calibrated to published Fedwire and FedNow statistics, a one-minute cap leaves at most 0.55% of liquidity for any engine to save; across 81 further synthetic configurations the maximum is 0.73%. Code, data and 102 brute-force tests are public.

Index Terms—payment systems, liquidity-saving mechanisms, real-time gross settlement, instant payments, combinatorial optimization, simulation

I. Introduction

Real-time gross settlement (RTGS) systems settle each interbank payment individually and with finality, so every bank needs enough intraday liquidity to cover its outflows as they occur. When liquidity is scarce, payments queue, and the system can gridlock even when every bank is owed as much as it owes. Liquidity-saving mechanisms (LSMs) hold queued payments centrally and settle offsetting sets of them together [1], [2].

LSM research shares a frame: a settlement day that ends. The classic bounds are an upper bound (UB), the peak net debit when every payment settles on arrival, and a lower bound (LB), the end-of-day net debit [1], [3]. Simulation studies place engines between them and trace the trade-off between liquidity and delay [4]–[7]. Instant-payment systems break this frame twice. They run 24/7, so there is no end of day to define the LB. And they promise near-immediate settlement, which caps how long any payment may queue. Huang and Zhang simulate a hybrid queue-and-netting mechanism for 24/7 retail settlement on Swedish data and find liquidity savings [8], but no bounds exist for this setting.

We ask how much liquidity an LSM can save in such a system, for any engine. Our contributions are:

1) A liquidity floor under a delay cap (Theorem 4): for any engine that settles every payment within T, even one that knows all future payments, each bank needs at least a computable amount of liquidity. At $T = 0$ it equals the UB.
2) When the floor is attained (Proposition 5): always for $T = 0$, for caps longer than the horizon, and when every bank only sends or only receives; not always in general, by a two-bank, four-payment counterexample. An exact time-indexed program attains it in 94% of small random instances.
3) Rolling liquidity bounds for 24/7 systems (Proposition 3), proven ordered, monotone and consistent with daily bounds. Without a daily reset, the worst 24-hour window needed 3.5× the worst calendar-day LB in our simulations.
4) Netting dominates reordering (Proposition 6): for the batch-reordering objective of [9], [10], settling the batch at once is optimal over all orderings, takes linear time, and certifies any reordering heuristic's distance from the best order.
5) An open benchmark and certified offsetting: simulator, generators calibrated to public Fedwire and FedNow statistics, seven settlement methods, and CertOff, an anytime age-weighted offsetting routine that reports a certified optimality gap on every call. A short self-contained proof that offsetting stays strongly NP-hard at zero balances (Theorem 1) motivates certificates over approximation schemes.

The empirical headline is a tension for policy: a cap of a minute or less leaves under 1% of liquidity for any engine to save, in every configuration we tested. Real headroom appears only when payments may wait about an hour.

II. Related Work

Table I positions the closest work against our claims.

Bounds and simulation. Koponen and Soramäki define the UB/LB pair and compute, by simulation, a "real lower bound" when some time-critical payments must settle on entry or within a set period [3]. Their bound

is engine-specific and simulated; Theorem 4 is engine-independent, proven, and covers every payment. Leinonen and Soramäki [4] and Arjani [5] trace the liquidity–delay trade-off by simulation. McLafferty and Denbee find CHAPS savings of about 30%, "almost invariant" across offsetting algorithms [6]; Byck and Heijmans show LSMs help most when liquidity is scarce [7]; Garratt et al. study banks' use of Lynx's offsetting algorithm [11] and propose auction-based LSMs [12]. For continuous settlement, Huang and Zhang simulate a 24/7/365 retail mechanism combining queues, multilateral netting and risk controls on Swedish transaction data, and show that thresholds, initial liquidity and submission timing drive settlement and deferral costs [8]; our bounds apply to any such mechanism.

Mechanism design and behaviour. Bech and Garratt model the timing game [13]; LSMs can improve welfare and encourage earlier submission [2], [14]; Castro et al. estimate policies by reinforcement learning [15]. Duffie, Singh and Wang design a maximally efficient LSM that settles the best simultaneous combination of submitted payments, allowing partial payments, in an RTGS day [16]; it has no delay cap and no 24/7 setting.

Clearing algorithms and complexity. Güntzer et al. model clearing as a knapsack-type problem with deposits as the limiting resources and study heuristics on Germany's largest interbank system [17]; Shafransky and Doudkin [18] and Cambe and Xing [19] solve offsetting with optimization, the latter with mixed-integer programs that beat two classic LSM algorithms in over 91% of simulations. Multi-period clearing has been studied with model predictive control [20], convex optimal control [21] and dynamic programming [22]; these minimise queued value or costs in deferred-net settings and give no floor under a hard delay cap. Continuous-time clearing of defaults in Eisenberg–Noe networks [23] is a different problem.

Reordering. McMahon et al. and Astuti et al. reorder batches of 70–140 payments on a hybrid quantum annealer for Canadian and TARGET2 data, settling all payments gross; Astuti et al. find simulated annealing gives comparable savings [9], [10]. Neither reports optimality gaps. Proposition 6 supplies one.

TABLE I
Closest prior work against this paper's claims

Work	24/7 bounds	Delay-cap floor	Netting vs ordering	Hardness (0 bal.)	Certified gaps
[3]–[5]	–	simulated	–	–	–
[1], [6], [7]	–	–	–	–	–
[8]	simulated	–	–	–	–
[17]	–	–	–	related[a]	–
[19]	–	–	–	–	–
[20]–[22]	–	–	–	–	–
[9], [10]	–	–	–	claimed[b]	–
[16]	–	–	related[c]	–	–
This paper	✓	✓	✓	✓	✓

[a]Knapsack-type model with deposits as limits; the abstract states no zero-balance hardness result (full text not accessed). [b]NP-hardness of reordering asserted without proof. [c]Optimises a simultaneous combination; no comparison with orderings. Checked on abstracts, open full texts and citation lists, 31 Oct 2026.

III. Model and Theory

There are n banks. A payment $p = (s_p, r_p, v_p, \tau_p)$ has a sender, receiver, value $v_p > 0$ and submission time $\tau_p > 0$ (the clock starts before the first payment). Settlement happens in batches; balances change only through settlements, and after each batch every balance $B_i \geq 0$. For a queue Q, a set $S \subseteq Q$ is feasible if

$$\mathrm{Out}_i(S) - \mathrm{In}_i(S) \leq B_i \quad \forall i, \tag{1}$$

where $\mathrm{Out}_i(S)$ and $\mathrm{In}_i(S)$ sum the values bank i sends and receives in S. Over time, $\mathrm{Out}_i(a,b]$ and $\mathrm{In}_i(a,b]$ sum payments submitted in $(a,b]$, and $D_i(a,b) = \mathrm{Out}_i(a,b] - \mathrm{In}_i(a,b]$. Full proofs are in the Appendix.

Definition 1 (MVS). Given weights $w_p \geq 0$, max-value settlement finds a feasible S maximising $w(S) = \sum_{p\in S} w_p$.

Theorem 1. MVS is strongly NP-hard, even when every balance is zero.

Sketch. From 3-PARTITION [24]: a hub pays item banks, each item bank may pay any bin bank, each bin bank pays B back to the hub; with zero balances every bank must balance exactly, so the optimum reaches $3mB$ iff a partition exists. Hence MVS has no FPTAS unless P = NP. Unlike the deposit-limited model of [17], hardness here holds for pure offsetting, with no liquidity at all.

Proposition 2 (certified gap). If $\bar{z} \geq z^*$ is any valid upper bound on the MVS optimum (LP relaxation or MIP dual bound) and S_H is feasible, then $(z^* - w(S_H))/z^* \leq (\bar{z} - w(S_H))/\bar{z}$.

For a window W,

$$LB_i^W = \max_t \max\big(0, D_i(t-W,t)\big), \tag{2}$$

$$UB_i^W = \max_t \max_{s\in[t-W,t]} \max\big(0, D_i(t-W,s)\big). \tag{3}$$

Proposition 3. (i) $LB_i^W \leq UB_i^W$. (ii) $UB_i^W = \max_{0\leq s-a\leq W} \max(0, D_i(a,s))$, so it is non-decreasing in W. (iii) With day length Δ and days starting at multiples of Δ, LB_i^Δ is at least every day's classic LB.

Theorem 4 (liquidity floor). Suppose an engine settles every payment p at a time $\sigma_p \in [\tau_p, \tau_p + T]$ and keeps all balances non-negative. Let $F_i^T(a,b) = \mathrm{Out}_i(a, b-T] - \mathrm{In}_i(a-T, b]$. Then bank i's balance just after time a satisfies $B_i(a) \geq \max_b \max(0, F_i^T(a,b))$. In particular its opening balance L_i satisfies

$$L_i \geq L_i^T = \max_b \max\big(0, \mathrm{Out}_i(0, b-T] - \mathrm{In}_i(0,b]\big). \tag{4}$$

Sketch. Every payment i sends with $\tau \in (a, b-T]$ must settle in $(a,b]$, and every payment it receives in $(a,b]$ was

Algorithm 1 CertOff$(Q, B, \alpha, T, \varepsilon, \Delta)$

$w_p \leftarrow v_p(1 + \alpha \min(a_p/T, 1))$ for $p \in Q$
$x_u \leftarrow$ greedily add payments with $a_p \geq T$, oldest first
$S_H \leftarrow$ multilateral offsetting on Q
$(\bar{z}, x^{LP}) \leftarrow$ LP relaxation of weighted MVS
$S_R \leftarrow$ greedily add from x_u in decreasing x^{LP}, oldest first
$S \leftarrow \arg\max\{w(S_H), w(S_R)\}$; $g \leftarrow (\bar{z} - w(S))/\bar{z}$
if $g > \varepsilon$ then
    $(S', \bar{z}') \leftarrow$ MIP for weighted MVS, time limit Δ
    $S \leftarrow$ better of S, S'; $\bar{z} \leftarrow \min(\bar{z}, \bar{z}')$; $g \leftarrow (\bar{z} - w(S))/\bar{z}$
return S, g

submitted in $(a-T, b]$. The floor binds clairvoyant engines, equals the UB at $T = 0$, and is non-increasing in T.

Proposition 5 (when the floor is attained). The minimum total opening liquidity over engines meeting cap T equals $\sum_i L_i^T$ if (i) $T = 0$; or (ii) $T \geq \max_p \tau_p - \min_p \tau_p$; or (iii) every bank only sends or only receives. It can exceed it: with banks A, B, $T = 1$ and payments $A \to B$ of 1 at $\tau = 1$, $B \to A$ of 3 and 4 at $\tau = 2$, $A \to B$ of 5 at $\tau = 3$, the floor sums to 1 but every schedule needs 2.

In the counterexample B wants to pay late, but A needs B's payment early to meet its own deadline; the floor treats each bank's best schedule as if it were free of the other's.

Proposition 6 (netting dominates reordering). Let a batch β settle one payment at a time in order π, bank i entering with position P_i and maximum net debit M_i, and let $d_i^\pi(k)$ be its net outflow after the first k payments. Then for every π,

$$\max\Big(0, \max_k \big(P_i + d_i^\pi(k)\big) - M_i\Big) \geq \max\big(0, P_i + \mathrm{net}_i(\beta) - M_i\big), \tag{5}$$

and the right side is the increase when β settles at once, netted.

The maximum over k includes $k = |\beta|$, whose value does not depend on π. The gain can be strict, as on any three-bank cycle. The netting cost is also a lower bound for every order, so it certifies any reordering heuristic.

IV. CertOff

Mixed-integer offsetting is known to beat classic LSM heuristics [19]. CertOff (Algorithm 1) adds two things suited to instant payments: weights $w_p = v_p(1 + \alpha \min(a_p/T, 1))$ that grow with a payment's age a_p, so old payments win ties; and an anytime certificate. A fast heuristic and an LP-rounded solution are compared; the exact solver runs only when the certified gap exceeds ε, within a time budget Δ, and the reported gap uses the tighter of the LP and MIP dual bounds.

V. Experimental Setup

Generator. Banks are linked by a core-periphery network (complete core, each periphery bank tied to two core banks), Erdős–Rényi ($p = 0.25$) or Barabási–Albert ($m = 2$). Payments arrive as a thinned Poisson process with morning and late-afternoon peaks; values are lognormal; each payment follows an edge chosen with probability proportional to $\sqrt{\deg u \deg v}$, in a random direction. Business-day mode runs 08:00–18:00 and resets balances daily; 24/7 mode runs continuously with 10% overnight and 20% weekend traffic. The synthetic baseline is $n = 25$, a 5-bank core, 150 payments per business hour and $\sigma = 1.5$.

Calibration. Two further scenarios use public statistics. Fedwire core: 66 banks around a 25-bank clique, the structure carrying 75% of Fedwire value [25], with 100,000 or 300,000 payments per weekday against Fedwire's 869,187 in 2025, and $\sigma \in \{2, 2.5\}$ consistent with near-lognormal values [25], [26]. FedNow: 69,309 payments per calendar day as in 2026:Q3 [26], 100 or 300 participants on a scale-free network, $\sigma \in \{2, 3\}$. Since all results are ratios to the UB, the value scale does not matter.

Methods. FIFO (per-sender), FIFO-bypass, bilateral and multilateral offsetting (each followed by a gross pass), exact MVS (HiGHS [27] via SciPy), CertOff ($\alpha = 1$, $\varepsilon = 0.01$, $\Delta = 0.5$–1 s), and for reordering, greedy and simulated annealing. Every engine attempts gross settlement on arrival and offsets every 5 minutes. Opening balances are $L_i = LB_i + \alpha_L(UB_i - LB_i)$.

Statistics and checks. Comparisons are paired over 30 seeded days, with Wilcoxon signed-rank tests, Holm correction, and median differences with 95% bootstrap intervals. 102 unit tests and an adversarial search over 7,200 edge-case instances (ties, payments at the opening, caps of zero or beyond the horizon) found no violation of any result; the search also produced the counterexample in Proposition 5. A FIFO engine given UB liquidity settled every payment on arrival on all 90 test days.

VI. Results

A. Validation against published findings

Two published patterns had to hold before any new result was trusted. First, LSMs help most when liquidity is scarce [7]: at LB liquidity ($\alpha_L = 0$) FIFO left 4.7% of value unsettled at the close and bypass 3.0%, while every offsetting engine settled everything; the advantage shrank as liquidity grew and was gone by $\alpha_L = 0.7$. Second, savings barely depend on the offsetting algorithm, which mainly changes delay [6]. Table II shows the liquidity each engine needs to settle every payment by the close (bisection over α_L, 30 days): all offsetting engines reach 37–38% below UB, close to the ∼30% reported for CHAPS, and differ mainly in delay.

TABLE II
Liquidity needed to settle everything by the close (V2)

Engine	Saving vs. UB	Mean delay (min)
FIFO	33.8%	28.0
FIFO-bypass	31.6%	23.6
Bilateral	37.3%	28.0
Multilateral	38.4%	30.4
Exact MVS	38.4%	29.0
CertOff	38.4%	29.0

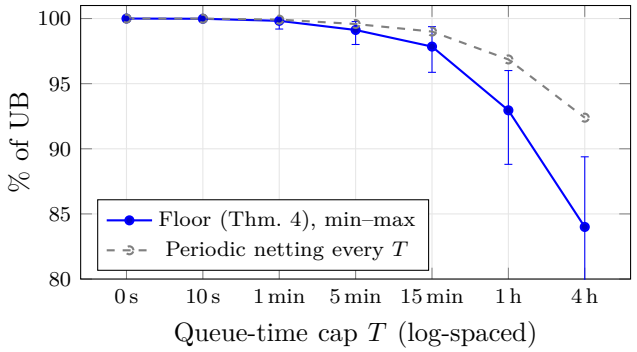


Fig. 1. Liquidity floor versus delay cap on 24/7 streams. The gap between the curves is the room left for a smarter engine than periodic netting.

B. Short delay caps leave almost nothing to save (E4)

Fig. 1 plots the Theorem 4 floor against the cap T on seven-day 24/7 streams (3 topologies, 5 seeds each). With $T = 1$ minute the floor averages 99.82% of the UB; with $T = 5$ minutes, 99.12%. Headroom grows only with long caps: 7.1% at one hour and 16.0% at four hours. A sensitivity sweep over 81 configurations (3 topologies, $n \in \{15, 25, 50\}$, 50–400 payments per hour, $\sigma \in \{1, 1.5, 2\}$) bounds the maximum possible saving at a one-minute cap by 0.73% (median 0.11%), and at five minutes by 2.9% (median 0.78%). These are bounds on every engine, so no algorithm, quantum or classical, can do better.

Calibrated to public data (E10). The pattern holds in the Fedwire- and FedNow-shaped scenarios (Section V; 24 seven-day runs). At a one-minute cap the maximum possible saving was 0.15–0.55% for the Fedwire core and 0.01–0.29% for FedNow; at five minutes, 0.46–3.03% and 0.22–1.36%; at one hour, 4.7–14.5% and 2.0–10.3%. Higher value dispersion (σ) lowered the headroom in every pairing, because a few large payments dominate each bank's peak.

Rolling bounds. Without a daily reset, net debits accumulate across days. The worst 24-hour window demanded on average 3.5× the worst calendar-day LB of the same stream (rolling $LB^{24\,\mathrm{h}}$ over the largest daily LB), and 1.05× the UB measured from the opening. A 24/7 operator who sizes liquidity from daily statistics will under-provision.

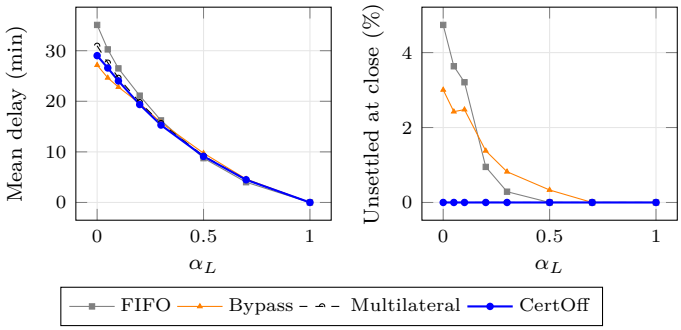


Fig. 2. Liquidity frontier on core-periphery business days (30 days per point). Exact MVS coincides with CertOff and is omitted.

C. Is the floor tight? (E9)

For 270 small instances (6 random or core-periphery banks, or 2 banks; 18 payments; $T \in \{5, 30, 180\}$ min) we solved the exact minimum opening liquidity over all schedules meeting the cap, as a time-indexed MIP. Restricting settlement to the times $\{\tau_p\} \cup \{\tau_p + T\}$ is without loss: moving every settlement in an open interval between consecutive candidate times to the later one keeps all deadlines and, by Proposition 6, cannot raise any peak. The optimum equalled the floor in 255 instances (94.4%): 89 of 90 at $T = 5$ min, 84 at 30 min, 82 at 3 h. Where it did not, the mean gap was small but the largest was 41%, so the floor is not always attainable. Periodic netting needed 7.2% more than the optimum on average, which is the room CertOff-style engines have to win.

D. Engines on business days (E2, E5, E6)

On the liquidity frontier (Fig. 2), offsetting engines remove all end-of-day unsettled value at every α_L tested, and CertOff matches the exact engine. Against multilateral offsetting, CertOff cut delay by a median 0.48 min at $\alpha_L = 0$ and 0.54 min at $\alpha_L = 0.05$ (95% CIs $[0.12, 1.96]$ and $[0.29, 1.06]$; Holm-adjusted $p < 0.001$), with smaller but significant gains up to $\alpha_L = 0.5$. Lower delays for FIFO and bypass at low liquidity reflect only that they leave payments unsettled. The pattern held on Erdős–Rényi and Barabási–Albert networks (E5): at $\alpha_L = 0.05$ multilateral, exact and CertOff settled everything on all three, while FIFO left 2.7–3.9% and bilateral 0.4–0.7% unsettled. Offsetting frequency mattered little (E6): moving from every 10 s to every 30 min raised multilateral delay from 27.4 to 28.3 min and CertOff's from 26.3 to 27.4 min, at 0.3 ms and 5–7 ms per call.

E. Optimality and certificates (E3, E7)

Table III scores each method on 200 queue snapshots per size, with each bank holding 30% of its gross outflow. Classic heuristics settle 74–88% of the optimal value; CertOff reaches 99.97–100% and reports a certified gap of at most 2.4% (mean 0.04%) within 0.4 s. The LP

TABLE III
Share of the optimal settled value (E3), mean

Method	$\|Q\| = 20$	$\|Q\| = 70$	$\|Q\| = 200$
FIFO-bypass	0.817	0.742	0.824
Bilateral (+gross pass)	0.880	0.796	0.856
Multilateral (+gross pass)	0.867	0.793	0.871
LP rounding	0.695	0.595	0.685
CertOff	1.000	1.000	0.9997
CertOff certified gap, max	0	0	0.024
CertOff time (s), mean	0.011	0.068	0.396

relaxation alone is a weak certificate (integrality gap 10–63%), so CertOff's MIP fallback is what makes its gaps small. HiGHS proved optimality in 97.5% of snapshots within 10 s. On larger queues (E7) exact MVS took 5 s at 500 payments and hit the 60 s limit at 1,000–2,000, while CertOff stayed near its 1 s budget with certified gaps below 0.4%.

F. Reordering versus netting (E3)

On 100 batches of 70 payments, the reordering objective of [9] cost 138.9 under FIFO order, 134.7 after greedy reordering, and 110.8 after simulated annealing (3.3 s per batch on a laptop). Batch netting cost 110.5, never more than any ordering (Proposition 6), and strictly less in 40% of batches. Because netting is a lower bound for every order, it also certifies the classical annealer: on average it was within 0.28% of the best possible order (at most 3.7%).

G. Strategic sending (E8, exploratory)

In best-response dynamics where each bank chooses what share of its morning traffic to delay to 16:00, an LSM (5-minute periodic netting) lowered the equilibrium delayed share from 0.668 to 0.638 on average across three delay-cost ratios and five seeds. The direction matches theory [2], [14], but the effect is small and this experiment is not powered for inference.

VII. Threats to Validity

Data. Transaction-level RTGS data is confidential. We calibrate to published aggregates, reproduce two published findings, and sweep 81 synthetic configurations; per-payment value dispersion and 24/7 traffic shape are assumptions, swept rather than measured. Solver budgets are equal across methods. Objectives: we report delay, unsettled value and cap violations alongside liquidity. Behaviour is fixed except in the exploratory E8. Literature: for [8], [17], [22] we read abstracts and front matter, not full texts.

VIII. Conclusion

A delay cap puts a hard floor under the liquidity any settlement engine needs. For caps of a minute or less that floor sits within 1% of gross settlement in every configuration we tested, including ones calibrated to Fedwire and FedNow. LSMs remain valuable where payments can wait, and there certified offsetting is fast enough to run on every call. Batch netting makes reordering, quantum or classical, unnecessary wherever simultaneous settlement is allowed. Characterising exactly when the floor is attained remains open.

Appendix A
Full Proofs

Standing assumptions. (A1) Balances change only through settlements. (A2) Payments settled at the same instant are applied jointly, so only the post-batch balances must be non-negative. (A3) Every submission time is positive, and bank i's opening balance is $L_i = B_i(0)$. $B_i(a)$ denotes the balance after all settlements at times $\le a$. Intervals $(a, b]$ are left-open; $\mathrm{Out}_i(a, b]$ and $\mathrm{In}_i(a, b]$ sum values of payments submitted in $(a, b]$, and are 0 when $b \le a$.

A. Theorem 1

Membership. A set S is a certificate checkable in $O(|S|+n)$ time.

Hardness. 3-PARTITION is strongly NP-complete [24]: given positive integers $a_1, \dots, a_{3m}$ with $\sum_j a_j = mB$, decide whether $\{1, \dots, 3m\}$ splits into m groups each summing to B. (The usual bounds $B/4 < a_j < B/2$ force groups of size three; the argument below does not need them.) Build banks H, $I_1, \dots, I_{3m}$, $K_1, \dots, K_m$, all with balance 0, and payments $h_j : H \to I_j$ of value a_j; $e_{jl} : I_j \to K_l$ of value a_j for every j, l; and $k_l : K_l \to H$ of value B. There are $3m + 3m^2 + m$ payments of value at most B, so the instance has size polynomial in the unary encoding.

Let S be feasible. Each bank has $\mathrm{Out}_i(S) \le \mathrm{In}_i(S)$, and $\sum_i \mathrm{Out}_i(S) = \sum_i \mathrm{In}_i(S) = w(S)$, so every inequality is an equality. For I_j: $\mathrm{In} = a_j[h_j \in S]$ and $\mathrm{Out} = a_j\,|\{l : e_{jl} \in S\}|$, so I_j pays exactly one bin if $h_j \in S$ and none otherwise. For K_l: $\mathrm{In} = \sum_{j:e_{jl}\in S} a_j$ and $\mathrm{Out} = B[k_l \in S]$, so each bin receives 0 or exactly B. Let $J = \{j : h_j \in S\}$. Then $w(S) = \sum_{j\in J} a_j + \sum_{j\in J} a_j + B \cdot |\{l : k_l \in S\}| = 3\sum_{j\in J} a_j \le 3mB$, with equality iff J is everything, in which case the bins form m groups of sum B. Conversely, a partition $G_1, \dots, G_m$ gives the feasible set $\{h_j\} \cup \{e_{jl} : j \in G_l\} \cup \{k_l\}$ of value $3mB$. So the optimum is $3mB$ iff a partition exists.

Corollary. Since w is integral and bounded by a polynomial in the unary size, an FPTAS run with $\varepsilon < 1/(3mB)$ would decide 3-PARTITION in polynomial time [24].

Two banks. With $B_1 = B$, $B_2 = 0$ and payments $1 \to 2$ of sizes a_j, feasibility is $\sum_S a_j \le B$, so subset-sum embeds and the problem is weakly NP-hard. Conversely, with two banks feasibility is $-B_2 \le \mathrm{Out}_1(S) - \mathrm{In}_1(S) \le B_1$, and a dynamic program over the attainable values of $\mathrm{Out}_1 - \mathrm{In}_1$ solves it in time polynomial in $|Q|$ and $\sum_p v_p$.

B. Proposition 2

S_H is feasible, so $w(S_H) \le z^*$; $\bar{z}$ is an upper bound, so $z^* \le \bar{z}$. If $z^* > 0$ then $w(S_H)/z^* \ge w(S_H)/\bar{z}$, which rearranges to the claim. The LP relaxation has $|Q|$ variables and n constraints and is solved in polynomial time; the MIP dual bound is valid by construction of branch-and-bound.

C. Proposition 3

(i) For each t, the choice $s = t$ in the definition of UB_i^W gives the term $\max(0, D_i(t-W, t))$, so $UB_i^W \ge LB_i^W$. (ii) Let $P_W = \{(t-W, s) : t \in \mathbb{R},\ s \in [t-W, t]\}$. If $(a, s) \in P_W$ then $0 \le s - a \le W$. If $0 \le s - a \le W$, put $t = a + W$; then $s \in [a, a+W] = [t-W, t]$, so $(a, s) \in P_W$. Hence $P_W = \{(a, s) : 0 \le s - a \le W\}$, which grows with W, and the maximum over a larger set cannot be smaller. (iii) Take $t = (k+1)\Delta$: the term is $\max(0, D_i(k\Delta, (k+1)\Delta))$, day k's classic LB.

D. Theorem 4

Fix bank i and $a < b$. By (A1)–(A2), $B_i(b) = B_i(a) - P + R \ge 0$, where P and R are the values i pays and receives in settlements at times in $(a, b]$. Outgoing. If $s_p = i$ and $\tau_p \in (a, b-T]$, then $\sigma_p \ge \tau_p > a$ and $\sigma_p \le \tau_p + T \le b$, so p settles in $(a, b]$. Hence $P \ge \mathrm{Out}_i(a, b-T]$. Incoming. If $r_p = i$ and $\sigma_p \in (a, b]$, then $\tau_p \le \sigma_p \le b$ and $\tau_p \ge \sigma_p - T > a - T$. Hence $R \le \mathrm{In}_i(a-T, b]$. So $B_i(a) \ge P - R \ge F_i^T(a, b)$, and $B_i(a) \ge 0$ by assumption; maximise over b. For the opening balance take $a = 0$: by (A3) no payment has $\tau \le 0$, so $\mathrm{In}_i(-T, b] = \mathrm{In}_i(0, b]$ and $L_i = B_i(0) \ge L_i^T$.

Properties. At $T = 0$, $L_i^0 = \max_b \max(0, D_i(0, b])$, the classic UB. If $T' \ge T$ then $\mathrm{Out}_i(0, b-T'] \le \mathrm{Out}_i(0, b-T]$ for every b, so $L_i^{T'} \le L_i^T$.

E. Settlement at candidate times (used in E9)

Let $\Theta = \{\tau_p\} \cup \{\tau_p + T\}$, sorted. Given any feasible schedule, move every settlement in an open interval (θ_k, θ_{k+1}) to θ_{k+1}. Such a payment had $\tau_p \le \theta_k$ and $\tau_p + T \ge \theta_{k+1}$, because both are candidate times bracketing σ_p; so the deadline still holds. Between θ_k and θ_{k+1} balances now stay at their values after θ_k, which were non-negative; at θ_{k+1} the jointly applied batch ends at the same balances as before. Opening balances are unchanged, so restricting to Θ loses nothing.

F. Proposition 5

Let L^* be the minimum total opening liquidity over schedules meeting the cap. Theorem 4 gives $L^* \ge \sum_i L_i^T$; we exhibit schedules achieving it. (i) $T = 0$ forces $\sigma_p = \tau_p$. Bank i then needs exactly $\max_b \max(0, D_i(0, b]) = L_i^0$. (ii) Let $\bar{\tau} = \max_p \tau_p$. Since $\tau_p + T \ge \min_q \tau_q + T \ge \bar{\tau}$, settling every payment at $\bar{\tau}$ meets the cap, and bank i needs $\max(0, \mathrm{net}_i)$, where net_i is its total outflow minus inflow. Taking $b = \bar{\tau} + T$ in L_i^T gives $L_i^T \ge \max(0, \mathrm{net}_i)$. (iii) A bank that only receives needs 0 under any schedule, and $L_i^T \ge 0$. A bank that only sends must cover its total outflow under any schedule, and L_i^T equals that total (take b large; $\mathrm{In}_i = 0$). Receivers' balances never fall, so the banks do not interact.

Counterexample. Let $T = 1$ and payments $p_1 : A \to B$ of 1 at $\tau = 1$; $p_2, p_3 : B \to A$ of 3 and 4 at $\tau = 2$; $p_4 : A \to B$ of 5 at $\tau = 3$. For A, the shifted outflows are $+1$ at 2 and $+5$ at 4, and inflows -7 at 2: the running sums after each time are $-6, -1$, so $L_A^1 = 0$. For B, inflows are -1 at 1 and -5 at 3, and shifted outflows $+7$ at 3: running sums $-1, 1$, so $L_B^1 = 1$. Now consider any schedule. p_1 settles in $[1, 2]$, p_2, p_3 in $[2, 3]$ and p_4 in $[3, 4]$. Case $L_A < 1$. A needs an inflow no later than $\sigma_1 \le 2$, and B's payments cannot settle before 2; so $\sigma_1 = 2$ and at least one of p_2, p_3 settles at 2. By then B has received at most 1 (p_4 is not yet submitted) and paid at least 3, so $L_B \ge 2$. Case $L_A \ge 1$. B must pay 7 by time 3 and can have received at most $1 + 5$ by then, so $L_B \ge 1$. Either way $L_A + L_B \ge 2 > 1$; the schedule with $L_A = L_B = 1$, $\sigma_1 = 1$ and p_2, p_3, p_4 all at time 3 achieves 2.

G. Proposition 6

Under order π bank i's position after k payments is $P_i + d_i^\pi(k)$, so its new maximum net debit is $\max(M_i, \max_k P_i + d_i^\pi(k))$ and the increase is the left side. Since $d_i^\pi(|\beta|) = \mathrm{net}_i(\beta)$ for every π, the inner maximum is at least $P_i + \mathrm{net}_i(\beta)$. Settling β at once moves the position from P_i to $P_i + \mathrm{net}_i(\beta)$ with no intermediate state, giving the right side. Summing over banks, simultaneous netting is optimal over all $|\beta|!$ orders and is computed in $O(|\beta| + n)$ time. For three banks paying v in a cycle, every order raises the first payer's peak by v, while netting raises none.

References

[1] M. L. Bech and K. Soramäki, "Gridlock resolution in interbank payment systems," Bank of Finland, Research Discussion Papers 9/2001, 2001.

[2] A. Martin and J. McAndrews, "Liquidity-saving mechanisms," Journal of Monetary Economics, vol. 55, no. 3, pp. 554–567, 2008.

[3] R. Koponen and K. Soramäki, Intraday Liquidity Needs in a Modern Interbank Payment System: A Simulation Approach, ser. Bank of Finland Studies. Bank of Finland, 1998, no. E:14.

[4] H. Leinonen and K. Soramäki, "Optimizing liquidity usage and settlement speed in payment systems," Bank of Finland, Discussion Papers 16/1999, 1999.

[5] N. Arjani, "Examining the trade-off between settlement delay and intraday liquidity in Canada's LVTS: A simulation approach," Bank of Canada, Working Paper 2006-20, 2006.

[6] J. McLafferty and E. Denbee, "Liquidity saving in CHAPS: A simulation study," in Simulation in Computational Finance and Economics: Tools and Emerging Applications, B. Alexandrova-Kabadjova et al., Eds. Hershey, PA: IGI Global, 2012.

[7] S. Byck and R. Heijmans, "How much liquidity would a liquidity-saving mechanism save if a liquidity-saving mechanism could save liquidity? A simulation approach for Canada's large-value payment system," Journal of Financial Market Infrastructures, vol. 9, no. 3, 2021.

[8] Y. Huang and J. Zhang, "Fintech innovations in service: 24/7/365 interbank settlement," Journal of the Knowledge Economy, vol. 16, pp. 18 559–18 582, 2025.

[9] C. McMahon, D. McGillivray, A. Desai, F. Rivadeneyra, J.-P. Lam, T. Lo, D. Marsden, and V. Skavysh, "Improving the efficiency of payments systems using quantum computing," Management Science, vol. 70, no. 10, pp. 7325–7341, 2024, arXiv:2209.15392.
[10] V. Astuti, A. Baldeschi, L. Bastianelli, G. Bruno, A. Desai, D. Marsden, and R. Russo, "Liquidity optimization in gross settlement systems with quantum reordering: Application to TARGET2," Journal of Economic Dynamics and Control, p. 105385, 2026.
[11] R. Garratt, Z. Lu, and P. Tian, "How banks create gridlock to save liquidity in Canada's large value payment system," Bank of Canada, Staff Working Paper 2023-26, 2023.
[12] R. Garratt, M. L. Bech, M. N. Petric, and C. Ates, "Auction-based liquidity saving mechanisms," Bank for International Settlements, BIS Working Papers 1318, 2025.
[13] M. L. Bech and R. Garratt, "The intraday liquidity management game," Journal of Economic Theory, vol. 109, no. 2, pp. 198–219, 2003.
[14] M. Jurgilas and A. Martin, "Liquidity-saving mechanisms in collateral-based RTGS payment systems," Federal Reserve Bank of New York, Staff Reports 438, 2010.
[15] P. S. Castro, A. Desai, H. Du, R. Garratt, and F. Rivadeneyra, "Estimating policy functions in payments systems using reinforcement learning," Bank of Canada, Staff Working Paper 2021-7, 2021.
[16] D. Duffie, S. F. Singh, and C. Wang, "An efficient liquidity savings mechanism," National Bureau of Economic Research, Working Paper 35679, 2026.
[17] M. M. Güntzer, D. Jungnickel, and M. Leclerc, "Efficient algorithms for the clearing of interbank payments," European Journal of Operational Research, vol. 106, no. 1, pp. 212–219, 1998.
[18] Y. M. Shafransky and A. A. Doudkin, "An optimization algorithm for the clearing of interbank payments," European Journal of Operational Research, vol. 171, no. 3, pp. 743–749, 2006.
[19] J. Cambe and Z. Xing, "Just solve it: A simple method to improve the design and performance of liquidity-saving mechanisms," Journal of Financial Market Infrastructures, 2025, rePEc:rsk:journ7:7959350; confirm volume, issue and year.
[20] Z. Chen, "An online-decision algorithm for the multi-period bank clearing problem," Journal of Industrial and Management Optimization, vol. 18, no. 4, pp. 2783–2803, 2022.
[21] S. Barratt and S. Boyd, "Multi-period liability clearing via convex optimal control," 2020, arXiv:2005.09066.
[22] S. Chen, O. Baron, and N. Chen, "Optimal dynamic clearing for interbank payments," Management Science, vol. 71, no. 4, pp. 2953–2974, 2025.
[23] I. M. Sonin and K. Sonin, "A continuous-time model of financial clearing," Annals of Operations Research, 2025.
[24] M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, 1979.
[25] K. Soramäki, M. L. Bech, J. Arnold, R. J. Glass, and W. E. Beyeler, "The topology of interbank payment flows," Physica A, vol. 379, no. 1, pp. 317–333, 2007.
[26] Federal Reserve Financial Services, "Fedwire funds service annual statistics; fednow service volume and value statistics," https://www.frbservices.org, 2026, accessed 31 Oct 2026.
[27] Q. Huangfu and J. A. J. Hall, "Parallelizing the dual revised simplex method," Mathematical Programming Computation, vol. 10, pp. 119–142, 2018.