TORC3 - Token-Ring Clearing Heuristic for Currency Circulation

Carlos Humes Jr.[*] **Marcelo de Souza Lauretto**[+]
Fabio Nakano[+] **Carlos A. de Bragança Pereira**[*]
Guilherme F. G. Rafare[++] **Julio Michael Stern**[*,**]
[*]IME-USP and [+]EACH-USP , University of Sao Paulo.
[++]FinanTech Ltda.
[**] `jstern@ime.usp.br`

Symposium on Stochastic Modeling Applied to Finance,
5th LNCC Meeting on Computational Modeling,
National Laboratory for Scientific Computing,
July 16-19, Petrópolis, Brazil.

1.1- The *TORC3* Algorithm

Clearing algorithms are at the core of modern payment systems, facilitating the settling of multilateral credit messages with (near) minimum transfers of currency.
Traditional clearing procedures use batch processing based on *MILP* - mixed-integer linear programming algorithms.
The *MILP* approach has the following drawbacks:
- demands intensive computational resources;
- delays payments after batch processing; and is
- vulnerable to operational risks in the inter-batch period.

1.2- The *TORC3* Algorithm

This paper presents *TORC3* - the Token-Ring Clearing Algorithm for Currency Circulation.
In contrast to the *MILP* approach, *TORC3* is a real time heuristic procedure, demanding modest computational resources, and able to completely shield the clearing operation against the participating agents' risk of default.

Clearing systems handle multilateral credit messages between participating agents. For example, at an interbank clearing system all participating agents are state, commercial or investment banks.
The clearing system receives credit messages and tries to *clear* them, that is, to execute them all by *netting*, that is, by mutual multilateral cancellation, using actual transfers of currency only as a last resort.

2.1- Credit Messages and Reserves

A credit message M stipulates a payment, in the amount of v standard units, to be made by agent-i in favor of or to be received by agent-j.
The information pertinent to a credit message, M, is specified in a list, $[t, i, j, v]$, having the following fields:

- t **-** Time stamp and identification number;
- i **-** Origin;
- j **-** Destination; and
- v **-** Value, specified as an integer number of standard units.

2.2- Credit Messages and Reserves

In order to protect the clearing system, shielding it against risks of default, each participating agent must guarantee all its operations. This assurance is provided by three reserves maintained at the clearing system by each agent:

- $R1$ - **Monetary reserve.** It represents capital in the form of currency, a resource that cannot be invested or used outside the clearing system. Hence $R1$ has the cost of loss of opportunity to the agent. Therefore, the clearing system is designed to keep it at very low level, corresponding to a small fraction of the agent's daily operations.
- $R2$ - **Collateral reserve.** It consists of financial assets kept in custody of the clearing system.
- $R3$ - **Ticket reserve.** It is the collection of tickets held by a participant at a given time. A ticket is an IOU (I-owe-you), an acknowledgement of debt sent by another agent. Tickets and tokens are short-living objects that are strictly internal to *TORC3*, as explained in the sequel.

2.3- Credit Messages and Reserves

When agent-*i*, sends a credit message for clearing, the system makes sure that agent-*i*'s reserves can back it up. The following condition checks this situation, as it will become clear from further details of the clearing process.

$$\text{Condition I:}\quad R1(i) + \min[R3(i), R2(i)] \ \geq\ v \ ,$$

The system also prevents excessive concentration of monetary reserve at the credit message destination, *j*, a situation that could result in *TORC3* stalling or loss of efficiency. This is accomplished by checking if the execution of this credit message would make $R1(j)$ exceed a stipulated maximum,

$$\text{Condition II:}\quad R1(j) + v \ \leq\ \text{MaxR1}(j) \ .$$

If conditions I and II are satisfied, the credit message is accepted for clearing; otherwise, it is rejected. Rejected messages are handled by external sub-systems that can pay them directly or resubmit them for clearing at a later time.

3.1- Ticket Transactions

The first step in processing a message is its disassembling into unitary (value-1) transactions.
Each v-valued message from agent-i to agent-j is disassembled into v unitary transactions from i to j.
The first basic idea used to develop *TORC3* is to find sets of unitary transactions that can be matched in closed cycles.
The transactions in such a cycle cancel each other, having a null resultant, that is, they can be settled with no actual transfer of currency.
The second basic idea is to provide a “buffer” that stores these transactions for a limited time.
During this time, a stochastic optimization heuristic tries to reduce “noise” or fluctuations in the incoming messages by cancellation of cyclic transaction sets.

3.2- Ticket Transactions

When a transaction enters the buffer, a ticket is sent from agent-i to agent-j.

- **Case 1:** If $R3(i) > 0$ and $R2(i) > 0$ then agent-i forwards a ticket from its reserve to agent-j, $R3(i)$ is decremented and $R3(j)$ is incremented. Since a ticket is just an IOU from a third party, agent-i must back up this operation decrementing its collateral reserve, $R2(i)$. We represent these operations using the synthetic notation: $R3(i)--$, $R2(i)--$, $R3(j)++$.
- **Case 2:** Else, If $R1(i) > 0$, agent-i creates a ticket and sends it to agent-j. Creating a ticket implies decrementing $R1(i)$ and incrementing $R3(i)$. In synthetic notation: $R1(i)--$, $R3(i)++$, $R3(i)--$, $R3(j)++$, or, equivalently, $R1(i)--$, $R3(j)++$.
- **Case 3:** Else, there is an inconsistency with Condition I, that is, $R1(i) + \min[R3(i), R2(i)] < v$ (exception handling).

3.3- Ticket Transactions

A ticket's path is represented by its creation time, t, and a list of agents who owned it, $[a, b, \ldots x]$, starting with its creator, a, at the list's head, and ending with its current owner, x, at the tail. Notice that, according to the previously defined rules to create and forward tickets, the operation at the head of a ticket's path is guaranteed by $R1$, monetary reserves used in Case 1, while the following operations are backed up by $R2$, collateral reserves used in Case 2.

A circuit in the ticket's path corresponds to a zero-resultant financial exchange. *TORC3* accomplishes multilateral nettings by the elimination of circuits in tickets, and the elimination of the corresponding transactions. When a circuit covers the whole ticket's path, we have a cycle. Hence, when a cycle is eliminated, the whole ticket is eliminated.

3.4- Ticket Transactions

At one hand, the longer tickets are allowed to exist in the buffer, the longer will be their trajectories, and the better the chances for circuit or cycle formation and elimination.
At the other hand, existing tickets deplete the participating agents' reserves; hence, tickets should not be allowed to accumulate for too long in the buffer.
Therefore, in order to obtain a good performance, the *TORC3* clearing system operator must set and fine tune `MaxTicketLife` - the tickets' maximum permitted life-span.
When a ticket reaches the maximum life-span, it is eliminated. Hence, also tickets with non-cyclic (lists of) transactions will be generated and eliminated by *TORC3* in the clearing process.
The rules for ticket and circuit elimination are explained in the next section.

3.5- Ticket Transactions

TORC3 needs a criterion for choosing which ticket to forward in a transaction, when more than one is available, i.e. if $R3(i) \geq 2$. The following deterministic heuristic combines three simple criteria:

1. If possible, forward a ticket that creates a cycle; if more than one is available, choose the longest cycle.
2. Else, if possible, forward a ticket that creates a circuit; if more than one is available, choose the longest circuit.
3. Else, in the last case, forward the oldest available ticket.

Alternative, and more efficient, randomized heuristics choose among the available tickets according to probabilities that are proportional to a scoring function taking into account the three simple criteria of the deterministic heuristic.

4.1- Tickets, Tokens and Circuit Elimination

TORC3 can eliminate an entire ticket with no circuits, or eliminate a ticket's sub-list of transactions forming a circuit, or eliminate a cyclic ticket. The transactions in an eliminated list are transformed into tokens, that are released from the buffer. A token can be of one of five types, according to the position of the corresponding transaction in the original ticket.
The five token types are:

- *A* - Initial in a path; $a \Rightarrow b \rightarrow c \rightarrow d$
- *E* - Intermediate; $a \rightarrow b \Rightarrow c \rightarrow d$
- *I* - Terminal in a path; $a \rightarrow b \rightarrow c \Rightarrow d$
- *O* - Initial in a cycle; $a \Rightarrow b \rightarrow c \rightarrow a$
- *U* - Singular or unit-length path. $a \Rightarrow b$

4.2- Tickets, Tokens and Circuit Elimination

The information pertinent to a token, T, is specified in a list, $[i, j, k, l]$, having the following fields:

- i - Origin;
- j - Destination;
- k - Type;
- l - Value.

For the sake of simplicity we consider in this article only tickets and tokens of unitary value. However, *TORC3* can be easily generalized to handle tickets in a convenient range of pre-selected values, like
$l \in \{1, 2, 5, 10, 20, 50, 100\}$; or
$l \in \{1, 3, 10, 30, 100\}$.

4.3- Tickets, Tokens and Circuit Elimination

The tokens with destination *j* released by the buffer go to agent-*j*'s assembly line, where the credit messages with the same destination wait in a queue to be reassembled. Notice that a token arriving to agents-*j*'s assembly line is used to reassemble the first message in line, independent of its origin. When a message is reassembled, it is released by *TORC3* sub-system, and passed to external sub-systems that shall handle actual financial operations, accounting reports, etc. Used tokens are destroyed as the message is released. When a token is destroyed, reserves used to back-up corresponding ticket transactions are released.

4.4- Tickets, Tokens and Circuit Elimination

Table 1 summarizes the rules used to increment and decrement the pertinent reserves at the time of a ticket transaction and at the time of the corresponding token destruction.
The destruction of tokens of type E and O, just restore the reserves used at the corresponding ticket transaction; these operations cancel each other, having a null resultant.
The destruction of a pair of tokens, of type A and I, at the head and the tail of a tickets (non-cyclic) path, $[a, b, \ldots x]$, have the net effect of transferring one unit of monetary reserve from the agent at the path's head, a, to the agent at the path's tail, x.
The token of type U handles the special case of a singular or unit-length path.

4.5- Tickets, Tokens and Circuit Elimination

Table 1: Reserve operation for each ticket transaction, token destruction and ticket rejection (only for exception handling).

Type	Ticket transac.	Token destruc.	Rejection
A	$R1(i)--$	nop	$R1(i)++$
E	$R2(i)--$	$R2(i)++$	$R2(i)++$
I	$R2(i)--$	$R2(i)++$ and $R1(j)++$	$R2(i)++$
O	$R1(i)--$	$R1(i)++$	$R1(i)++$
U	$R1(i)--$	$R1(j)++$	$R1(i)++$

The five token types are:
(A) Initial in a path; (E) Intermediate; (I) Terminal in a path; (O) Initial in a cycle; (U) Singular or unit-length path.

4.6- Tickets, Tokens and Circuit Elimination

Finally we give some illustrative examples for the elimination process. Arrows represent ticket transactions. Reserve decrements at ticket transaction time are annotated above each arrow. Corresponding token types and reserve increments at the time of token destruction are annotated under the arrows.

Path:

$$\begin{tikzcd}
a \arrow[r, Rightarrow, "R1(a)--", "A"'] & b \arrow[r, Rightarrow, "R2(b)--", "\substack{E \\ R2(b)++}"'] & c \arrow[r, Rightarrow, "R2(c)--", "\substack{I \\ R2(c)++,\ R1(d)++}"'] & d
\end{tikzcd}$$

4.7- Tickets, Tokens and Circuit Elimination

Circuit:

$$\begin{array}{ccccccc}
 & R1(a)-- & & R2(b)-- & & R2(c)-- & \\
a & \Longrightarrow & b & \Longrightarrow & c & \Longrightarrow & b \\
a & \Longrightarrow & b & E & & E & \\
 & & & R2(b)++ & & R2(c)++ &
\end{array}$$

Cycle (left) and Singular or unit-length Path (right):

$$\begin{array}{ccccccccc}
 & R1(a)-- & & R2(b)-- & & & & R1(a)-- & \\
a & \Longrightarrow & b & \Longrightarrow & a & \qquad\qquad & a & \Longrightarrow & b \\
 & O & & E & & & & U & \\
 & R1(a)++ & & R2(b)++ & & & & R1(b)++ &
\end{array}$$

4.8- Tickets, Tokens and Circuit Elimination

“Real time” operation of the clearing system requires that credit messages can only wait a limited time to be executed.
If a credit messages time in *TORC3* sub-system exceeds `MaxMessageLife` the order is rejected.
Table 1 summarizes the rules used to increment and decrement the pertinent reserves for tokens destroyed when a partially assembled message is rejected.
This kind of message rejection should not occur if the clearing system operates under coherent conditions, including $\texttt{MaxTicketLife} < \texttt{MaxMessageLife}$.
Hence, the last column of Table 1 is only important for exception handling and error recovery situations.

5.1- Simulations and Performance

FinanTech and *Banco do Brasil* have established three benchmark scenarios used to test a *TORC3* prototype.
These scenarios were built according to statistical profiles typical of the 1999-2000 operations at *CIP-SBP*, the Clearing House of the Brazilian Interbank Payment System.
Scenarios 1, 2 and 3 are presented in Table 2 and have, respectively, 30, 100 and 173 participants.
Participating agents are ranked according to their size, measured by typical transaction volume for clearing.
In each of the tree scenarios, a stream of 300,000 random credit messages is generated so that:
- The value of each message is distributed uniformly between 1 and 100 standard units.
- The origin and destination of each message are distributed with weights proportional to the agents' size.
- These random streams of credit messages where inputed to *TORC3* prototype over 480 minutes (8 hours).

5.2- Simulations and Performance

Table 2.1: Simulation Scenario 1 (30 Agents)

Group	Agents	Agent Size %	Group Size %	Accumulated %
1	1	23	23	23
2	1	17	17	40
3	1	15	15	55
4	1	13	13	68
5	2	10	20	88
6	24	0.5	12	100

Table 2.2: Simulation Scenario 2 (100 Agents)

Group	Agents	Agent Size %	Group Size %	Accumulated %
1	4	15	60	60
2	6	5	30	90
3	10	0.2	2	92
4	80	0.1	8	100

5.3- Simulations and Performance

Table 3 gives the minimum (for the larger agents), maximum (for the smaller agents) and average required monetary reserve, as a percentage of the agent's expected transaction volume ($R1$ is set by an auxiliary Bayesian procedure).
The collateral reserves are always assumed to be sufficient.
Hence, in these simulations, rejections may occur by lack of $R1$, never by insufficient $R2$.
CPU (Pentium III) idle time shows that *TORC3* could handle the task even using very modest computational resources.
Moreover, the message rejection rate was below 3%, using a total amount of $R1$ (monetary reserves) below 0.4% of total transacted value (considered an excellent performance).
Nevertheless, as expected, the simulations show that the required $R1$ must be relatively higher for small participants.
This observation suggests the possibility of organizing cooperating groups of small agents.

5.4- Simulations and Performance

Table 3: Simulation Results

Data	Scenario 1	Scenario 2	Scenario 3
Agents	30	100	173
Cred.Messages	300,000	300,000	300,000
Accepted Messages (% number)	97.0	95.8	95.7
Accepted Messages (% volume)	97.8	96.9	96.1
Average Message size	50	50	50
Max. Message Life (minutes)	15	15	15
Simulated clearing period (mnts)	480	480	480
CPU idle time (%)	95	81	58
$R1$ (% of agent's volume), min	0.3	0.3	0.3
$R1$ (% of agent's volume), mean	0.38	1.15	5.1
$R1$ (% of agent's volume), max	0.4	1.3	13.0

6- Final Remarks

Future Developments:

TORC4, the Token-Ring Clearing Heuristic for Complementary Currencies Circulation, is presently under development.
It extends the scope of *TORC3* for multiple circulating currencies under complex and dynamic conversion rules.

Direct References:

- C.Humes Jr., M.S.Lauretto, F.Nakano, G.F.G.Rafare, C.A.B.Pereira, J.M.Stern (2012). *TORC3 - Token-Ring Clearing Heuristic for Currency Circulation.* EBEB 2012, XI Bayesian Statistics Brazilian Meeting. To appear in AIP Conference Proceedings.
- G.F.G.Rafare, J.M.Stern, C.A.B.Pereira, F.Nakano (2001). Algoritmo de Concretização e Compensação de Mensagens de Pagamento. Tipo de Programa: AT03, AT04, TC02; Campos de Aplicação: EC04, EC06, FN03, FN04. Número de registro no INPI: 00042036 (patent number at the Brazilian Institute of Industrial Property).