# *Feasibility of fluid transport modelling for buffer and processing systems : Information Fluid Dynamics*

Florian De Vuyst[1,2]

[1]Université de Cergy-Pontoise
Département de Mathématiques
BP 222 Pontoise
95011 Cergy-Pontoise cedex
FRANCE

[2]Centre de Mathématiques et de leurs applications CMLA
ENS de Cachan
61, avenue du Président Wilson
94235 Cachan cedex FRANCE
e-mail:
devuyst@cmla.ens-cachan.fr

*Abstract*

Finding numerical tools for rapidly prototyping very large communication networks or complex information value chains is a topical question. The motivation of this paper is to propose algorithms that achieve robustness, accuracy and rapid simulations. Here we present a fluid model for a general family of buffer and processing systems. The model leads to a set of nonclassical hyperbolic partial differential equations (PDEs). One of the main advantage of this model is that it is not sensible to the well-known ripple effect, that makes it an unconditionally outperforming candidate compared to event-driven algorithms. In this paper we insist on the way to numerically discretize such equations in order to show both feasibility and computational effort. The mathematical models are validated by numerical experiments.

# Contents

# 1 Introduction and objectives

Information system performance through traditional packet-level simulation is becoming increasingly difficult as today's infrastructures and subsystems grow in scale along many dimensions, especially for telecommunications, e-business, collaborative management, etc. As a need expressed by consulting groups and large electronic business-based companies, fluid simulation has been proposed to cope with the size and complexity of such systems.
The fluid paradigm that abstracts the flow of information data as a fluid flow is born in the $80ths$ ( [AMS82]) and today is widely used in the simulation community and industry. The classical modelling concepts come from stochastic theory and Markov chains, ordinary differential equations (ODEs), pure computer aggregation theory, integrodifferential analytical solutions and so on.

At our knowledge, deterministic partial differential equations (PDEs) have not been proposed for such problems yet, although the concept of fluid modelling is often evoked in large networks problems and the success of averaged PDE models for Computational Fluid Dynamics (CFD) is well-known.

By analogy, we propose a contribution to what could be termed as Information Fluid Dynamics (IFD).
The deterministic PDE modelling actually seems well-adapted because it intrinsically takes into account effects of memory and transport inside each processing block. However, other methods like classical ODE approaches or analytical solutions need to model this memory by some cumulative processes. For example, Reich gives the following analytical solution (see [Rei58], [Ben63]) : suppose a conserving server with constant service rate $\Phi_o$ that is empty at initial time. Let $A(t)$ denote the cumulative offered fluid at time $t$. Then the backloged fluid $s(t)$ is given by

$$s(t) \;=\; \sup_{0\le u\le t}\left\{A(t)-A(u)-\Phi_o\,(t-u)\right\}. \tag{1}$$

Some numerical schemes have been derived from the discretization of (1) (see [YG99]). But they cannot strictly respect the fluid conservation at the discrete point of view because of the lack of flux balance in the expression.
Because of the growing complexity of scheduling and management rules under constraints of quality of service (QoS) and relative cost, it is often necessary to

deal with work-conserving multiplexing systems that distinguish different sources of fluid. Then several authors ([LGK+99]) have derived multifluid extensions of expression (1), but under strong hypotheses of:

- piecewise constant fluid arrival rates that keep the solutions be analytical and allow exact assessments of departure component fluid rate;
- time-constant total departure rate.

In the case of large scale networks, such models produce a "ripple effect" of rate changes to manage (see [KSCK96],[LGK+99]). This is illustrated in figure 1 : as three fluid chunks enter a FIFO ("First In - First Out" policy) queue with two rate changes for each component, the coupling between each fluid served using a fraction of the overall departure rate produce an increasing number a rate changes per fluid to manage downstream the network, here 5, 5 and 2 respectively. As the size an complexity grow, the ripple effect can result in fluid simulations becoming more expensive than their packet-level counterparts. A

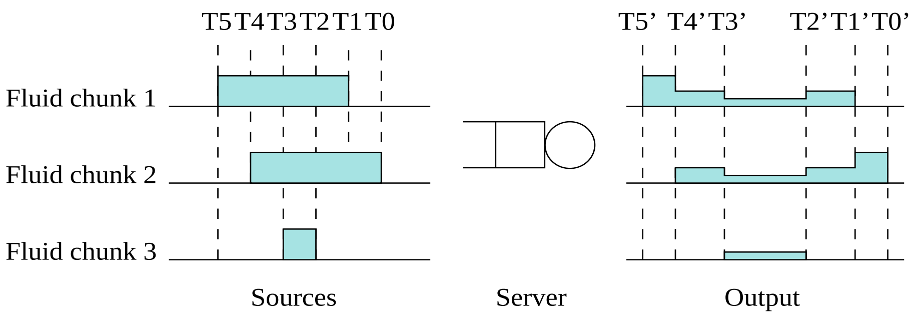


Figure 1: An example of ripple effect, taken out of Liu *et al* [LGK+99]

solution proposed in [LFG+01] to raise the ripple effect is the flow aggregation policy. The idea is to isolate the flow of interest and aggregate the remaining flow into a single flow. But for certain applications aimed at achieving multicriteria decision making support, the actual flow of interest is the complete set of colored component flows.

Another solution proposed by Yan and Gong [YG98] is to discretize equation (1) using time-approximation scheme and a fixed time step. Their time-driven approximation provides both upper bound and lower bound error estimates and

numerical results are very promising. But the extension for multi-class fluid with per flow routing is still in progress.
Other works deal with ordinary differential equation (like [KM98] for particular buffer management and control system).
In this paper, we use classical arguments taken out from Fluid Mechanics to form models of queuing systems. In fact, the flow problem inside a queue can be seen as an incompressible flow inside a domain with a free moving boundary. The length of the domain varies according to the amount of backloged fluid (see figure 2). It is well-known that free boundary problem are quite complex to solve and numerically approximate (combustion, fluid-structure interaction, etc). But, in one space dimension, we can make a change of space variable in order formulate the problem into a fixed domain, let say $]0,1[$. In that case, the difficulty now concentrates on the fluid nature that becomes compressible. But, the reformulated compressible fluid problem is easier to discretize as we will see later.
One of the main advantage of this model is that it is not sensible to the ripple

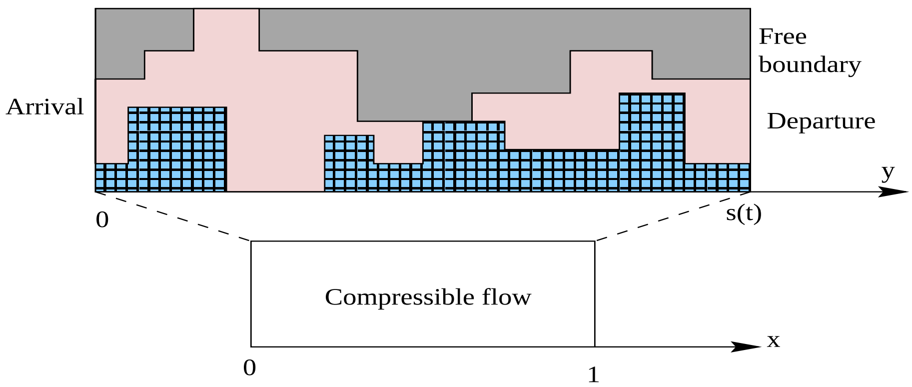


Figure 2: Free boundary incompressible flow versus fixed boundary compressible flow

effect of change of rate. This alternative approach provides a new unconditionally outperforming candidate compared with event-driven algorithms. Moreover, the extension to multi-class fluid is natural and quite easy.

As example, we show here the coupled system of ODEs-PDEs for FIFO model:

$$\frac{ds}{dt} = \Phi_i(t) - \Phi_o(t), \ t \geq 0,$$

$$\frac{\partial \rho_k}{\partial t} + \frac{\partial}{\partial x}\left(\rho_k \frac{x\Phi_o(t) + (1-x)\Phi_i(t)}{s(t)}\right) = 0, \quad x \in ]0,1[, \ t \geq 0,$$

$$(IC) \quad \rho_k(x,0) = \rho_k^0(x) \geq 0 \text{ subject to } \sum_{k=1}^{N} \rho_k^0(x) = s_0 > 0 \ \forall x \in [0,1],$$

$$(IC2) \quad s(0) = s_0,$$

$$(BC) \quad \rho_k(0,t) = \frac{\Phi_{i;k}(t)}{\Phi_i(t)} s(t), \ \forall t \geq 0,$$

where $x$ and $t$ ar both space and time variables, $s(t)$ is the queue size, $\Phi_{i;k}(t)$ is the $k$-th fluid component time-dependent arrival rate, $\Phi_i(t) = \sum_k \Phi_{i;k}(t)$, $\Phi_o(t)$ is the time-dependent departure rate, IC and BC state for initial and boundary conditions, $\rho_k(x,t)$ is the density per unit length of the $k$-th fluid at time $t$ and position $x$. That means that, for any interval $\omega$ in $]0,1[$,

$$\int_\omega \rho_k(x,t)\, dx$$

measures the quantity of backlogged $k$ fluid inside interval $\omega$ at time $t$.
One drawback of a fluid model is that the accuracy of the interested measures is compromised due to the abstraction. In this study, we will not address the accuracy issues of the fluid simulator, but rather focus on its stability, robustness and computational complexity. Nicol *et al* [NGJ99] claim that despite the high level of abstraction of fluid models, the error of estimated measures obtained with fluid simulation is very small compared to the results of packet-level simulation.

The paper is organized as follows:

In section 2, we present a classical framework of models using ODEs. Some numerical schemes will be presented. Those will be useful to understand the way to properly discretize the dynamics systems involving PDEs. In section 3, we present the construction of our fluid models for both multicomponent FIFO and LIFO (Last In - First Out) queues. In section 4, we expose a processor sharing PDE model. In the three following sections, we expose some numerical schemes and prove their accuracy, robustness and stability. Finally, in section 8, we present numerical experiments. They agree with the theoretical results and show that they can support irregular fluid sources and capture the bursty nature of the network traffic.

Notice finally that, although this fluid model has been especially designed for information systems project management needs because of the large amount of events to manage, we claim that it can also be applied for technical communication network architecture dimensioning, or critical production material flux management in conventional industry.

# 2 Dynamic queue modelling using ordinary differential equations

We here adopt a continuous modelling. Let $\Phi_i : \mathbb{R} \to \mathbb{R}$ and $\Phi_o : \mathbb{R} \to \mathbb{R}$ be two given positive bounded piecewise continuous functions that respectively denote the incoming flux and outgoing flux of a queue. The incoming flux is an average on the rate of arriving material on a certain time interval. The outgoing flux corresponds to a certain throughput of processing per time unit. From these two fluxes are derived an effective incoming flux $\Phi_{ei}$ and an effective outgoing flux $\Phi_{eo}$ of general expressions

$$\Phi_{ei} \; = \; \Phi_{ei}(\Phi_i, \Phi_o, x, x_M), \quad \Phi_{eo} \; = \; \Phi_{eo}(\Phi_i, \Phi_o, x, x_M).$$

The relations will be detailed later. The root differential problem that governs the evolution of $x$ of initial size $x_0$ is

$$\begin{cases} \dfrac{dx}{dt} \; = \; \Phi_{ei}(t) \; - \; \Phi_{eo}(t), \quad t > 0, \\ x(0) = x_0. \end{cases} \tag{2}$$

## 2.1 Taking into account queue size bounds

The queue size cannot become negative and in general cannot exceed a maximum size that we will denote $x_M$. So one has to design $\Phi_{ei}$ and $\Phi_{eo}$ in such a way that these two constraints are fulfilled. The physical interpretation of what happens when the bounds are reached leads to the following definition:

$$\Phi_{ei}(t) = \begin{cases} \Phi_o(t) \text{ if } x(t) = x_M \text{ and } \Phi_i(t) - \Phi_o(t) \geq 0, \\ \Phi_i(t) \text{ otherwise,} \end{cases} \tag{3}$$

$$\Phi_{eo}(t) = \begin{cases} \Phi_i(t) \text{ if } x(t) = 0 \text{ and } \Phi_i(t) - \Phi_o(t) \leq 0, \\ \Phi_o(t) \text{ otherwise.} \end{cases} \tag{4}$$

The bound processing can be seen as a flux limiting approach. The whole method respects the conservation of the global "mass" of material into consideration. Note that when the queue is empty, all happens as if the flow instantaneously crossed the queue block with infinite velocity. When the maximum queue size is reached, then the effective incoming flux adapts its value with the outgoing flux to make stable the limit size value $x_M$. In that case, there exists a lost (or redirected) flux $\Phi_l > 0$ equal to $\Phi_i(t) - \Phi_o(t)$. Of course, this loss is not an error of modelling, but an effective loss of information due to congestion.
This model is a continuous translation of the real even-driven phenomenon (see for example [Joh97] for a discrete event description).

## 2.2 Numerical approximation and system engineering software integration

In what follows, let $\tau^n$ be the time step of discretization. We will denote it $\tau$ in the case of a constant step. For that discussion, we will only deal with the (explicit or implicit) Euler method, but the extension of the comments to more accurate methods like Runge-Kutta (see [Sha91]) would hold and be straightforward. Let us denote by $x^n$ the approximation value of the queue size at time $t^n$, and $\Phi_{ei}^n = \Phi_{ei}(t^n)$, etc.

What is encountered in most of system simulators and softwares for dealing with such a model is an EDO solver coupled with a feedback loop for state conditions. But it presents the drawback not to strictly respect the range of admissible value for $x$. Indeed, because of time discretization for each block system, the feedback operates only after it detects a negative value for $x$ or a value greater than $x_M$. This error can become large if large time steps are used and produce spurious behaviors in the global system.
Of course, the model value could be "clipped" to be exactly equal to 0 or $x_M$,

but the conservation property is then lost. This can be critical in some particular applications.

A solution is to adapt locally the block time step in order to perfectly reach zero or $x_M$. Let us deal with the near zero case. Suppose that at time $t^n$, $\Phi_i(t^n) - \Phi_o(t^n) < 0$ and

$$x^n \ + \ \tau(\Phi_i^n - \Phi_o^n) \ < 0.$$

To preserve the positivity, a local time step $\tau^n < \tau$ namely

$$\tau^n \ = \ -\frac{x^n}{\Phi_i^n - \Phi_o^n}$$

could be used in order to perfectly compute $x^{n+1} = 0$. But this involves a variable time step strategy which is not always easy to configure into a system simulation software with simplified time block event handler.

In what follows we propose a method that automatically preserves the range of the queue size independently to the choice of the time step. Moreover, we will not need to distinguish crude (or potential) flux and effective flux. Let us first deal with positivity preserving. From the explicit Euler scheme

$$x^{n+1} \ = \ x^n \ + \ \tau(\Phi_i^n - \Phi_o^n),$$

is built a semi-implicit Euler scheme with a corrective term

$$x^{n+1} \ = \ x^n \ + \ \tau(\Phi_i^n - \frac{x^{n+1}}{x^n}\,\Phi_o^n).$$

This scheme clearly preserves the positivity of the sequence $(x^n)$ for any $\tau$. Let us inspect the consistency of the scheme. Suppose $x^n > 0$. The consistency error $e^n$ is then given by

$$e^n \ = \ x(t^{n+1}) \ - \ \left(1 + \tau\frac{\Phi_o^n}{x^n}\right)^{-1} (x^n + \tau\Phi_i^n).$$

Because $x'(t^n) = \Phi_i^n - \Phi_o^n$, it is easy to show that for a solution $x(t)$ of class $\mathcal{C}^2$,

$$|e^n| \ = \ \frac{\tau^2}{2}M \ + \ \frac{\tau\frac{\Phi_o^n}{x^n}}{1+\tau\frac{\Phi_o^n}{x^n}}\ \tau(\Phi_i^n - \Phi_o^n), \quad M = \sup_{[t^n,t^{n+1}]} |\Phi_i'(t) - \Phi_o'(t)|.$$

If $\frac{\tau}{x^n} \to +\infty$, that means that if $\tau$ is not sufficiently small when $x^n$ tends to zero, then the error consistency is of order one and leads to a zero order scheme which is not reasonable. If a variable time step is used in order to control the ratio $\frac{\tau}{x^n}$, one can see that this would lead to a method with a time step that tends to zero and a bounded advance in time of limit $t^\star = \inf\{t, x(t) = 0\}$ at convergence. That remark leads us to continue to use a fixed time step and forget the implicit term when conditions of queue loading are present. This leads to the rule

$$\begin{aligned}
&\text{If } \Phi_i^n - \Phi_o^n > 0, \text{ then solve} \\
&\qquad x^{n+1} \;=\; x^n \;+\; \tau(\Phi_i^n - \Phi_o^n), \\
&\text{else solve} \\
&\qquad x^{n+1} \;=\; x^n \;+\; \tau(\Phi_i^n - \frac{x^{n+1}}{x^n}\,\Phi_o^n).
\end{aligned}$$

Note that, when $x^n << \tau\Phi_i^n$ and $x^n << \tau\Phi_o^n$, then the implicit scheme gives

$$\frac{x^{n+1}}{x^n} \;\sim\; \frac{\Phi_i^n}{\Phi_o^n},$$

so that the artificial outgoing flux $\frac{x^{n+1}}{x^n}\Phi_o^n$ is equivalent to $\Phi_i^n$ which is consistent with the expected average outgoing flux in that case.

We can adopt the similar approach for the upper bound $x_M$ and combine the two strategies for a final algorithm:

$$\begin{aligned}
&\text{If } \Phi_i^n - \Phi_o^n > 0, \text{ then solve} \\
&\qquad x^{n+1} \;=\; x^n \;+\; \tau\left(\frac{x_M - x^{n+1}}{x_M - x^n}\,\Phi_i^n - \Phi_o^n\right), \qquad (5) \\
&\text{else solve} \\
&\qquad x^{n+1} \;=\; x^n \;+\; \tau\left(\Phi_i^n - \frac{x^{n+1}}{x^n}\,\Phi_o^n\right). \qquad (6)
\end{aligned}$$

We have finally proved

**Proposition 1** *If $x^0 \in [0, x_M]$, the sequence $(x^n)$ given by (5),(6) is bounded with values in $[0, x_M]$. When $x^n \to 0$,*

$$\Phi_{eo}^n \;=\; \frac{x^{n+1}}{x^n}\,\Phi_o^n \;\sim\; \Phi_i^n;$$

*When $x^n \to x_M$,*

$$\Phi_{ei}^n \;=\; \frac{x_M - x^{n+1}}{x_M - x^n}\,\Phi_i^n \;\sim\; \Phi_o^n;$$

This method is clearly easy to implement and does not require feedback loops. We give here a numerical example: let us choose

$$\Phi_i(t) = \frac{1}{2}(2 + \mathrm{sgn}(\sin(\pi t/2))), \quad \Phi_o(t) = \frac{1}{2}(2 - \mathrm{sgn}(\sin(\pi t/2))).$$

We suppose that the initial queue size is zero and choose $x_M = 1$. The exact solution for the queue size is time piecewise linear, although effective incoming and outgoing fluxes are piecewise constant. Below, we compare exact and numerical solutions for $\tau = 0.1$. Qualitatively, the numerical sequence belongs to $]0,1[$. The discrete error is somewhat important, but this can be enhanced by using second order accurate schemes. Finally, we show the numerical results obtained with $\tau = 0.001$.

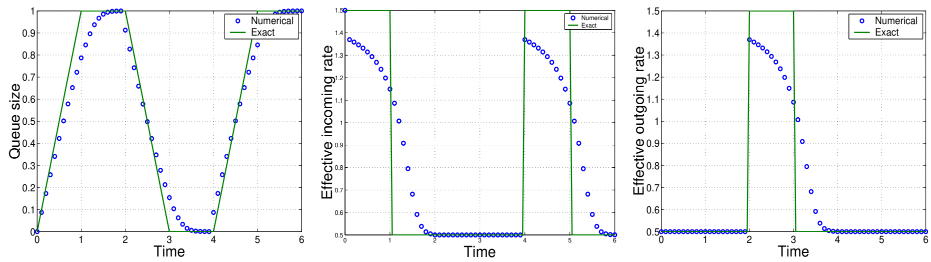


Figure 3: Comparison between numerical solutions and exact solutions for $\tau = 0.1$

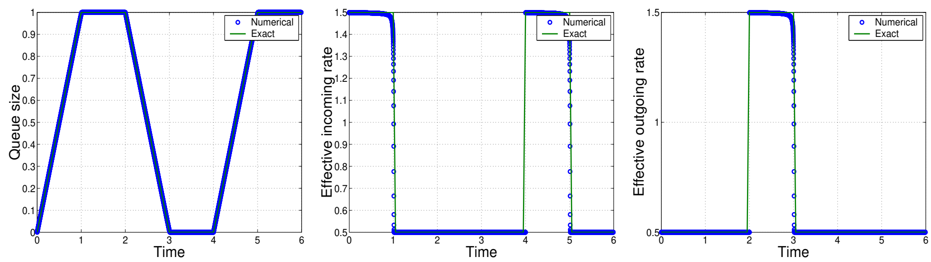


Figure 4: Comparison between numerical solutions and exact solutions for $\tau = 0.001$

**Remark 1** *Because of numerical float number truncation errors, it is observed that the computational sequence* $x^n$ *can attain values* $0$ *or* $x_M$*. To overcome* *division by zero errors, the effective algorithm is indeed*

$$\text{If } \Phi_i^n - \Phi_o^n > 0, \text{ then solve}$$
$$x^{n+1} = x^n + \tau \left( \frac{x_M - x^{n+1}}{x_M - x^n + \epsilon} \Phi_i^n - \Phi_o^n \right),$$
$$\text{else solve}$$
$$x^{n+1} = x^n + \tau \left( \Phi_i^n - \frac{x^{n+1}}{x^n + \epsilon} \Phi_o^n \right).$$

*for any small positive parameter $\epsilon$ larger than the computer accuracy.*

Finally notice that the range requirement for sequences $(x_n)_n$ involves a limiting process on crude fluxes that defines the numerical effective rates. We give the last definition:

**Definition 1** *The respective arrival and departure rates*

$$\Phi_{i;e}^n = \frac{x_M - x^{n+1}}{x_M - x^n + \epsilon} \Phi_i^n, \quad \Phi_{o;e}^n = \frac{x^{n+1}}{x^n + \epsilon} \Phi_o^n *$$

*are called limited fluxes or numerical effective fluxes.*

# 3 Dynamic queue modelling using partial differential equations

## 3.1 Scalar case for FIFO modelling

We are here interested in the queue modelling with the "First In, First Out" (FIFO) rule. We give coordinates to the FIFO queue: suppose that at the entrance the abscissa $y$ is 0 and the position at the end of the queue at time $t$ is is the queue size $y = s(t)$. We have again

$$\frac{ds}{dt} = \Phi_i(t) - \Phi_o(t).$$

As a Lagrangian method, we want to follow the path of a "particle" of material inside the queue. Let $y(t; y_0, t_0)$ be the particle at position $y$ at time $t$ with initial position $y_0$ at time $t_0$. If $y(t; 0, t_0)$ in still inside the queue, then the evolution of its position is governed by the differential equation

$$\frac{d}{dt} y(t; 0, t_0) = \Phi_i(t).$$

With the Lagrangian approach, the queue is seen as a domain with right moving boundary and the material flow as an incompressible oflow.
We decide to pass to an Eulerian approach. We make the variable change

$$x(t) = \frac{y(t)}{s(t)}$$

so that the queue is here seen as a steady interval $[0,1]$. We wish to follow a particle travelling inside $[0,1]$, so we are interested in the particle

$$x(t;0,t_0) = \frac{y(t;0,t_0)}{s(t)}.$$

A derivation gives

$$\frac{d}{dt}x(t;0,t_0) = \frac{1}{s(t)}\left(\frac{d}{dt}y(t;0,t_0) - x(t;0,t_0)\,\frac{d}{dt}s(t;0,t_0)\right)$$

so that

$$\frac{d}{dt}x(t;0,t_0) = \frac{x(t;0,t_0)\Phi_o(t) + (1-x(t;0,t_0))\Phi_i(t)}{s(t)}. \qquad (7)$$

This defines a particle velocity $v(t)$. In order to study the mean flow inside the queue, we naturally define a density of mass $\rho : \mathbb{R} \times \mathbb{R}^+ \to \mathbb{R}^+$ per unit of queue size (unit chosen here as 1). The flow is here seen as a compressible one. First, because $\rho$ is a mass density, it is normal to have

$$\int_0^1 \rho(x,t)\,dx = s(t).$$

We want to derive an equation on $\rho$. Let $\omega = ]\omega_L, \omega_R[$ be an open set of $]0,1[$. The conservation principle inside $\omega$ says that the variation of mass in $\omega$ is equal (up to the sign) to the balance of flux through its boundary, or again

$$\frac{d}{dt}\int_\omega \rho(x,t)\,dx + [\rho(x,t)v(x,t)]_{\omega_L}^{\omega_R} = 0.$$

By making tend to zero the size of $\omega$, this leads to the partial differential equation

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}\left(\rho \frac{x\Phi_o(t) + (1-x)\Phi_i(t)}{\int_0^1 \rho(x,t)\,dx}\right) = 0. \qquad (8)$$

This is a scalar nonlinear conservation law with a nonstandard form because of the integral term. The associated flux function $f$ is defined by

$$f(\rho, x; \Phi_i, \Phi_o) = \rho\frac{x\Phi_o(t) + (1-x)\Phi_i(t)}{\int_0^1 \rho(x,t)\,dx}.$$

It is easy to see that the nature of the equation of hyperbolic. Indeed it has a finite positive velocity of propagation $v(t)$. Because of assumptions on regularity on $\Phi_i$ and $\Phi_o$, it is clear that $f$ is at least Lipschitz continuous. This is a necessary condition of well-posedness for hyperbolic problems. Finally, to have a well-posed problem, we also need to add an upstream boundary condition at $x = 0$. This is naturally

$$f(\rho, 0; \Phi_i, \Phi_o) \;=\; \Phi_i(t) \tag{9}$$

which gives the value

$$\rho(0,t) \;=\; s(t) \;=\; \int_0^1 \rho(x,t)\, dx.$$

In fact, in the scalar case, the solution is trivial being equal to

$$\rho(x,t) \;=\; \tilde{\rho}(t) \;=\; s(t).$$

Indeed, by searching a solution that does not depend on $x$, this gives $\tilde{\rho}(t) = s(t)$ up to a constant, but this constant is indeed zero because of the boundary condition. This remark invites us to choose an initial condition $\rho^0(x)$ which is in fact constant over the interval $]0,1[$ and equal to the initial queue size $s(0)$. By this way, we have proved the existence of a solution.

This result seems quite curious and would say that the PDE approach does not bring anything in the scalar case. But in the multi-species vector case, we are going to see that this approach is powerful and overcomes some problems of numerical integration encountered with ODE approaches.

## 3.2 Multi-species case for FIFO

From now on, we distinguish different "materials" entering inside the queue. The motivation of such a work is that lots of engineering or organization problems encountered today need to manage different sources of production or information inside the same process unit, so that the simulation step for process optimization needs a level of description with "colored" materials. Suppose that we are looking for the evolution of $N$ different species of materials inside a FIFO queue. Of course, some incoming fluxes per species $\Phi_{i;k}(t)$ have to be given. We will denote by $\Phi_i(t)$ the total incoming flux, *i.e.*

$$\Phi_i(t) \;=\; \sum_{k=1}^{N} \Phi_{i;k}(t). \tag{10}$$

We also suppose that the total outgoing flux $\Phi_o(t)$ (which corresponds to a capacity of processing) is known. It is supposed that the speed of processing does not differ from the different species. Let us denote by $\rho_k$ the density per size unit of the $k$th species and $\rho = \sum_k \rho_k$ the total density. By the same approach than before, we could show that an Eulerian approach leads to the following system of conservation laws

$$\frac{\partial \rho_k}{\partial t} \;+\; \frac{\partial}{\partial x} f_k(\rho_l, x; \Phi_{i;l}, \Phi_o) \;=\; 0, \quad k = 1...N, \tag{11}$$

with component flux

$$f_k(\rho_l, x; \Phi_{i;l}, \Phi_o) \;=\; \rho_k \frac{x\Phi_o(t) + (1-x)\Phi_i(t)}{\sum_{k=1}^{N} \int_0^1 \rho_k\,(x,t)\,dx}. \tag{12}$$

The above equations are coupled by the bias of the integral mass term

$$s(t) \;=\; \sum_{k=1}^{N} \int_0^1 \rho_k\,(x,t)\,dx.$$

This is a degenerate hyperbolic system with a unique linearly degenerate multiple field of positive velocity

$$\frac{x\Phi_o(t) + (1-x)\Phi_i(t)}{\sum_{k=1}^{N} \int_0^1 \rho_k\,(x,t)\,dx}.$$

What is particular here is that the propagation velocity function is not bounded on the state space and tends to infinity when $s(t)$ tends to zero. To this system are associated some upstream boundary conditions

$$f_k(\rho_l, 0; \Phi_{i;l}, \Phi_o) \;=\; \Phi_{i;k}(t) \tag{13}$$

which express that the mass fractions

$$Y_k \;=\; \frac{\rho_k}{\rho}$$

at interface $x = 0$ are equal to a fraction of the incoming flux:

$$Y_k(0,t) \;=\; \frac{\Phi_{i;k}(t)}{\Phi_i(t)}. \tag{14}$$

Although the structure of the global solution $U = (\rho_1, \rho_2, ..., \rho_N)$ is no more trivial, we can again enounce that the total density $\rho$ is a function of the time alone provided that the total density at initial time is also constant. Let $\Omega = \{U = (\rho_1, \rho_2, ..., \rho_N),\ \rho_k \geq 0, \sum_{k=1}^N \rho_k > 0\}$ be the state space for the multi-species queue modelling. We state

**Proposition 2** *Let us consider the solution $U(x,t)$ of the problem (11),(14) with initial conditions $U(x,0) = U^0(x) \in BV(]0,1[)$, $U^0(x) \in \Omega\, \forall x \in ]0,1[$ such that*

$$\rho(x,0) = \sum_{k=1}^{N} \rho_k^0(x) \;=\; s(0). \tag{15}$$

*Then*

$$\rho(x,t) \;=\; \sum_{k=1}^{N} \rho_k(x,t) \;=\; s(t) \tag{16}$$

*where $s(t)$ denotes the queue size at time $t$. The component outgoing flux $\Phi_{o;k}(t)$ is given by*

$$\Phi_{o;k}(t) \;=\; Y_k(1,t)\ \Phi_o(t). \tag{17}$$

This compatibility property (16) will have to be preserved at the discrete level for numerical stability and consistency purposes.

**Proof.** By summing the consecutive $N$ equations of conservation, we fall down into the conservation law of the total density (8). We have already proved that this equation has a trivial solution $\tilde{\rho}$ that does not depend on $t$ if its initial data are constant. More, by integrating the equation on $]0,1[$, one finds

$$\frac{d\tilde{\rho}}{dt} \;=\; \Phi_i(t) - \Phi_o(t) \;=\; s(t).$$

That closes the proof.

**Comments 1** *The solutions of such a system provide both the size of the queue and the fractions of species at any position of the queue during their travel.*

**Remark 2** *At first sight, it would seem strange to transform an ordinary differential problem into a more complex partial differential one. In fact, in the differential case, one needs to memorize in time the profiles of incoming and outgoing fluxes and integrate it to build a particle travel. In the partial differential case, this memorizing is materialized by the field of the $\rho_k$'s.*

### 3.3 LIFO modelling

We now deal with the "Last In First Out" rule. We directly write the problem in the multi-component case. Suppose that the abscissa of the entrance of the FIFO queue is $y = s(t)$ at time $t$ and $y = 0$ is its bottom. It the particle $y(t; s(t_0), t_0)$ is still in the queue, then the evolution equation is now

$$\frac{d}{dt} y(t; s(t_0), t_0) \;=\; 0;$$

indeed, the particle do not move, only the right boundary of the queue moves ! By the change of variable $x(t) = y(t)/s(t)$, we find the evolution equation

$$\frac{d}{dt} x(t; s(t_0), t_0) \;=\; \frac{x(\Phi_o(t) - \Phi_i(t))}{s(t)},$$

where $\Phi_i(t)$ is defined like in (10). Then, by a conservation principle, we find the following system of conservations laws

$$\frac{\partial \rho_k}{\partial t} \;+\; \frac{\partial}{\partial x} g_k(\rho_l, x; \Phi_{i;l}, \Phi_o) \;=\; 0, \quad k = 1...N, \tag{18}$$

with component flux

$$g_k(\rho_l, x; \Phi_{i;l}, \Phi_o) \;=\; \rho_k \frac{x(\Phi_o(t) - \Phi_i(t))}{\sum_{k=1}^N \int_0^1 \rho_k\,(x,t)\,dx}. \tag{19}$$

Note here that the boundary conditions are different from the FIFO case. Indeed, at $x = 0$, the flux is zero and no boundary conditions have to be imposed. This expresses a "wall" boundary condition at the bottom of the queue. At $x = 1$, it is either an upstream or downstream boundary condition according to the sign of $\Phi_i(t) - \Phi_o(t)$. More precisely,

- If $\Phi_i(t) - \Phi_o(t) < 0$, then the queue size decreases and no condition has to be imposed (downstream condition).

- If $\Phi_i(t) - \Phi_o(t) > 0$, then the queue size increases and one has to impose a condition. Remark that a fraction of the incoming flux equal to $\frac{\Phi_o(t)}{\Phi_i(t)}$ is directly redirected outside the queue because of the LIFO process, so that the component outgoing flux is again equal to

$$\Phi_{o;k}(t) \;=\; \frac{\Phi_{i;k}(t)}{\Phi_i(t)}\,\Phi_o(t).$$

Then, the residual incoming flux serves for boundary condition. We impose:

$$g_k(\rho_l, 1; \Phi_{i;l}, \Phi_o) = \Phi_{i;k}(t) - \frac{\Phi_{i;k}(t)}{\Phi_i(t)}\,\Phi_o(t) \text{ if } \Phi_i(t) - \Phi_o(t) > 0. \tag{20}$$

Then we find the analogous result

**Proposition 3** *Let us consider the solution $U(x,t)$ of the problem (18),(20) with initial conditions $U(x,0) = U^0(x) \in BV(]0,1[)$, $U^0(x) \in \Omega\,\forall x \in ]0,1[$ such that*

$$\rho(x,0) = \sum_{k=1}^{N} \rho_k^0(x) \;=\; s(0). \tag{21}$$

*Then*

$$\rho(x,t) \;=\; \sum_{k=1}^{N} \rho_k(x,t) \;=\; s(t) \tag{22}$$

*where $s(t)$ denotes the queue size at time $t$. The component outgoing flux $\Phi_{o;k}(t)$ is given by*

$$\Phi_{o;k}(t) \;=\; \begin{cases} \dfrac{\Phi_{i;k}(t)}{\Phi_i(t)}\,\Phi_o(t) \;\; \textit{if } \Phi_i(t) - \Phi_o(t) > 0, \\ \Phi_{i;k}(t) \;+\; Y_k(1,t)\,(\Phi_o(t) - \Phi_i(t)) \quad \textit{otherwise.} \end{cases} \tag{23}$$

## 4 Model of processor sharing

Suppose that a process is able to completely treat a request of a task within a delay time $\theta$. Then the crude rate of this process is

$$\Phi_o \;=\; \frac{1}{\theta}.$$

Now suppose that, instead of adding a queue to the system, the policy is to perform all the tasks at the same time by splitting up uniformly the capacity of processing. That means that, if at a time $t$, the number of tasks to process is $N$, then the effective delay time is now $N\,\theta$.

Moreover, because some requests only begin to be processed while other ones are almost completely treated, that involves a certain transport process along the value chain. So this leads us to consider again some partial differential transport equations for modelling.

### 4.1 Scalar case model

We adopt a continuous approach. For generality purposes, it can supposed that the delay time $\theta$ is a time function. Suppose that the (virtual) length of the value chain is normalized to 1. Suppose also that, at time $t$, the number of request during processing is $s(t)$. We define again a density of mass $\rho(x,t)$, *i.e.* for $\omega = ]\omega_L, \omega_R[ \in ]0,1[$,

$$\int_\omega \rho(x,t)\,dx$$

is the number of requests in progress between $\omega_L$ and $\omega_R$. In particular,

$$\int_0^1 \rho(x,t)\,dx \;=\; s(t).$$

As said before, because of resource sharing the effective rate at time $t$ is

$$\Phi_e(t) \;=\; \frac{\Phi_o(t)}{\int_0^1 \rho(x,t)\,dx}.$$

A mass balance sheet in the interval $\omega$ gives

$$\frac{d}{dt}\int_\omega \rho(x,t)\,dx \;+\; \left[\frac{\rho(\omega_R,t)\Phi_o(t)}{\int_0^1 \rho(x,t)\,dx} - \frac{\rho(\omega_L,t)\phi_o(t)}{\int_0^1 \rho(x,t)\,dx}\right]_{\omega_L}^{\omega_R} \;=\; 0.$$

By making tend the size of $\omega$ to zero, this leads to then partial differential equation

$$\frac{\partial \rho}{\partial t} \;+\; \frac{\partial}{\partial x}\left(\rho\,\frac{\Phi_o(t)}{\int_0^1 \rho(x,t)\,dx}\right) \;=\; 0. \tag{24}$$

This is again a nonlinear transport equation of hyperbolic nature with a nonlocal term. For well-posedness are added to this equation an initial condition

$$\rho(x,0) \;=\; \rho^0(x) \tag{25}$$

for any given function $\rho^0 \in BV(]0,1[$ and an upstream boundary condition

$$\rho(0,t)\,\frac{\Phi_o(t)}{\int_0^1 \rho(x,t)\,dx} \;=\; \Phi_i(t). \tag{26}$$

By integrating equation (24) over $]0,1[$, we find the differential equation for $s(t)$:

$$\frac{d}{dt}s(t) \;=\; \Phi_i(t) \;-\; \frac{\rho(1,t)\Phi_o(t)}{s(t)}, \tag{27}$$

which shows the effective outgoing rate

$$\Phi_{eo}(t) \;=\; \frac{\rho(1,t)\Phi_o(t)}{s(t)}.$$

depending on the "internal state" $\rho$. Now for this problem, the solution cannot be a function of the time alone, even in the scalar case.
We are looking for solutions $\rho(x,t)$ such that $\int_0^1 \rho(x,t)\,dx > 0$.

## 4.2 Vector multicomponent case model

Suppose now that we want to distinguish different sources of request or tasks for simulation accuracy purposes. We keep the notation of the FIFO multicomponent model. The extension to the multicomponent case is then straightforward: the system of equations is

$$\frac{\partial \rho_k}{\partial t} + \frac{\partial}{\partial x}\left(\rho_k \frac{\Phi_o(t)}{\sum_{k=1}^n \int_0^1 \rho_k(x,t)\,dx}\right) \;=\; 0, \tag{28}$$

with boundary conditions

$$\rho_k(0,t)\,\frac{\Phi_o(t)}{\sum_{k=1}^n \int_0^1 \rho_k(x,t)\,dx} \;=\; \Phi_{i;k}(t).$$

The total mass

$$s(t) \;=\; \sum_{k=1}^N \int_0^1 \rho_k(x,t)\,dx$$

inside the system at time $t$ satisfies the differential equation

$$\frac{d}{dt}s(t) \;=\; \Phi_i(t) \;-\; \sum_{k=1}^N \rho_k(1,t)\,\frac{\Phi_o(t)}{s(t)}.$$

It is obtained by summation on $k$ and integration over $]0,1[$. The components of the effective outgoing rate vector are

$$\Phi_{eo;k}(t) \;=\; \rho_k(1,t)\,\frac{\Phi_o(t)}{s(t)}. \tag{29}$$

To summarize, the complete dynamic problem reads

$$\frac{\partial \rho_k}{\partial t} + \frac{\Phi_0(t)}{s(t)}\,\frac{\partial \rho_k}{\partial x} = 0, \quad x \in ]0,1[, \quad t \geq 0, \tag{30}$$

$$\frac{d}{dt}s(t) = \Phi_i(t) - \sum_{k=1}^{N} \rho_k(1,t)\,\frac{\Phi_o(t)}{s(t)}, \tag{31}$$

$$(BC) \quad \rho_k(0,t) = \frac{\Phi_{i;k}(t)}{\Phi_o(t)}\,s(t), \tag{32}$$

$$(IC) \quad \rho_k(x,0) = \rho_k^0(x), \; s(0) = \sum_{k=1}^{N} \int_0^1 \rho_k^0(x)\,dx, \tag{33}$$

with assumptions $\Phi_{i;k}, \Phi_o \in \mathcal{C}^0 \cap L^\infty(0,+\infty)$, $\rho_k^0 \in BV(0,1)$, $\rho_k^0 \geq 0$ and $s(0) > 0$. Notice that we could writes this system only with $\rho_k$ as unknowns forgetting $s(t)$. But the given system seems easier to discretize, replacing a presumed numerical integration of the non local term in (28) by an ODE solver for $s$.

### 4.3 Qualitative properties

System (30)-(33) as stable solutions in some cases. Indeed, suppose for $N = 1$ that

$$\Phi_i(t) = \rho(1,t)\,\frac{\Phi_o(t)}{s(t)} \forall t.$$

This situation can arise for example when the sharing operating system is supposed not to treat more than a certain bound of requests. When this limit is attained, when the incoming rate limited to the outgoing rate in order to maintain the number of requests equal to the upper bound. The total mass $s$ is constant, so that from (31) and (32), one gets

$$\rho(0,t) = \rho(1,t).$$

We have found stable bounded periodic solutions. In this case, the system behaves like a delay block with a delay time $\theta(t)$ at time $t$ that satisfies

$$\int_{t-\theta(t)}^{t} \frac{\Phi_o(y)}{s}\,dy = 1.$$

In particular, this delay time become constant if $\Phi_o$ is constant, equal to

$$\theta = \frac{s}{\Phi_o}.$$

It is proportional to the total load mass inside the system. For $s = 1$, that means if only one request is in progress, then we retrieve of course $\theta = {\Phi_0}^{-1}$. For $N \geq 2$, under the same hypotheses, the total density profile stays periodic, showing again stable solutions, but the field of mass fractions $Y(x,t)$ of course can vary.

# 5 Numerical discretization for the FIFO equation

## 5.1 Building an explicit scheme

Because of the hyperbolic nature of the system of equations, it is natural to use upwind difference scheme for the approximation of the solutions of (11),(12),(13). The numerical analysis of such schemes is nowadays quite well-understood (see [GR91] for example). As a starting point, let us present the first order accurate explicit scheme.

Suppose that we use a constant mesh step $h = \frac{1}{M}$ for a given integer $M$. Let $\tau^n$ denote variable time step at time $t^n$. We define $M$ mesh points $x_j = (j - \frac{1}{2})\, h$, $j = 1..M$ on the interval $[0,1]$. We will often need to use midpoints $x_{j+\frac{1}{2}} = j\, h$, $j = 0, M$. Finally, let us denote $\Phi^n_{i;k} = \Phi_{i;k}(t^n)$, $\Phi^n_i = \Phi_i(t^n)$ and $\Phi^n_o = \Phi_o(t^n)$. We are going to compute piecewise constant approximations functions $(\rho_k)^{\tau,h}$ with constant values $\rho^n_{k;j}$ on the cell $I_j = ]x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}[\times[t^n, t^{n+1}[$. The variables $\rho^n_{k;j}$ are also approximations of the value of the exact solutions $\rho_k$ at points $(x_j, t^n)$. Still for notations, we define the vector $U^n_j = (\rho^n_{1;j}, \rho^n_{2;j}, ..., \rho^n_{N;j})$. Suppose that the sequence $(U^n_j)_{j=0,M}$ is known. We want to build a numerical scheme that gives the sequence $(U^{n+1}_j)_j$ at time $t^{n+1} = t^n + \tau^n$.

For the initial data approximation, we decide to use the conservative projection

$$\rho^n_{k;j} \;=\; \frac{1}{h}\int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} \rho^0_k(x)\, dx. \tag{34}$$

Suppose that $\sum_k \rho^0_k(x) = s(0)\ \forall x$. Then the above average process preserves the property at the discrete level:

$$\rho^0_j \;=\; \sum_{k=1}^{N} \rho^0_{k;j} \;=\; s^0\ \forall j \in \{0, ..., M\}.$$

Consequently, the compatibility condition (22) of proposition 2 is at least respected at the initial time.

We deal with conservative schemes of the form

$$U_j^{n+1} \ = \ U_j^n \ - \ \frac{\tau^n}{h} \ \left(\Phi(U_j^n, U_{j+1}^n) - \Phi(U_{j-1}^n, U_j^n)\right) \tag{35}$$

where the numerical flux $\Phi(U,V)$ is a Lipschitz continuous numerical flux consistent with the real flux $F = (f_k)_k$, *i.e.* $\Phi(U,U) = F(U)$. This flux has to be upwind for stability purposes. To build a simple upwind flux, we are going to use solutions of local approximate Riemann problems at cell interfaces $x_{j+\frac{1}{2}}$. Consider the following Riemann problem in the cell $]x_j, x_{j+1}[\times[t^n, t^{n+1}[$:

$$\begin{cases} \frac{\partial \rho_k}{\partial t} \ + \ \frac{\partial}{\partial x}\tilde{f}_k(\rho_k, x_{x_{j+\frac{1}{2}}}, \Phi_{i,l}^n, \Phi_o^n, s^n) \ = \ 0, \ \forall k = 1, N, \\ \rho_k(x, t^n) \ = \ \rho_{j;k}^n \, 1_{x < x_{j+\frac{1}{2}}} \ + \ \rho_{j+1;k}^n \, 1_{x > x_{j+\frac{1}{2}}}, \end{cases} \tag{36}$$

where

$$\tilde{f}_k(\rho_k, x_{x_{j+\frac{1}{2}}}, \Phi_{i,l}^n, \Phi_o^n, s^n) \ = \ \rho_k \ \frac{x_{j+\frac{1}{2}}\Phi_o^n + (1 - x_{j+\frac{1}{2}})\Phi_i^n}{s^n}, \ k = 1..N. \tag{37}$$

Note that we have added a tilde symbol on $f_k$ to express the certain approximation $s^n$ of the nonlocal term of the component flux. For the moment, remember only that $s^n$ is some approximation of the queue size $s(t^n)$. Its expression will be given later.

Remark that these equations are no more than local linear transport equations with common transport velocity, so that the solutions of approximate Riemann problems are trivial and have a closed form.

Provided that $s^n$ always stays positive during the time iterations, we always deal with positive propagation velocity. That leads us to propose as numerical upwind flux

$$\Phi_k(U_j^n, U_{j+1}^n) \ = \ \rho_{k;j}^n \ \frac{x_{j+\frac{1}{2}}\Phi_o^n + (1 - x_{j+\frac{1}{2}})\Phi_i^n}{s^n} \quad \forall k. \tag{38}$$

For both well-posedness and stability of the scheme, we need to add upstream boundary conditions. According to the previous continuous analysis, at the left of the cell $I_1$ is imposed vector the flux

$$\Phi_L \ = \ (\Phi_{i;k})_k \, .$$

Comparing this flux with (38), all happens as if the numerical flux (38) were imposed with a left "ghost cell" of state $U_0$ equal to

$$\rho_{k;0}^n \ = \ \frac{\Phi_{i;k}^n}{\Phi_i^n} \, s^n$$

such that

$$\Phi(U_0, U_1) \; = \; \Phi_L \tag{39}$$

We can know give our result of stability for the resulting difference scheme:

**Proposition 4** *Suppose that the initial data $U_j^0$ are computed by (34) such that $U_j^0 \in \Omega$ for all $j$, and*

$$\sum_{k=1}^{N} \rho_{j;k}^0 \; = \; s^0 \quad \forall j.$$

*Let us denote $s^n$ the quantity*

$$s^n \; = \; \sum_{k=1}^{N} \rho_{k;1}^n. \tag{40}$$

*Then, under the Courant-Friedrichs-Levy condition on time step*

$$\frac{\tau^n}{h} \frac{\max(\Phi_i^n, \Phi_o^n)}{s^n} \; < 1, \tag{41}$$

*(known as CFL condition), the numerical scheme (35),(38),(39) has the following properties:*

1. *$s^n > 0$ and*
$$\sum_{k=1}^{N} \rho_{k;j}^n \; = \; s^n \quad \forall j \in \{1, .., M\}, \quad \forall n \in \mathbb{N}; \tag{42}$$
2. *it preserves the positivity of the $\rho_k$'s:*
$$\rho_{k;j}^n > 0 \quad \forall j \in \{1, .., M\}, \quad \forall n \in \mathbb{N}; \tag{43}$$
3. *it is first order accurate in both space and time.*

Before performing the proof, let us make the following remarks.

**Remark 3** 1. *The property (42) is a discrete version of the compatibility condition (22). It seems to be essential in order to preserve a good consistency with the nonlocal mass term.*

2. *The CFL condition (41) can become very restrictive on the time step size in the case where outward flux conditions lead to a queue size that tends to zero. In fact, it is easy to prove that $\tau^n$ also tends to zero. This is not reasonable. That is why an implicit version of the scheme is used in the next subsection.*

**Proof.** We want to prove the property for all $n$

$$\mathcal{P}^n \;=\; \{\sum_k \rho^n_{k;j} = s^n \;\forall j,\; \rho^n_j > 0 \;\forall j,k\}.$$

Of course, $\mathcal{P}^0$ is true because the discrete initial condition is supposed to belong to the state space and satisfy the compatibility condition. Suppose now that $\mathcal{P}^n$. We first note that

$$\begin{aligned}\rho^{n+1}_{k;j} \;&=\; \left(1 - \frac{\tau^n}{h}\frac{x_{j+\frac{1}{2}}\Phi^n_o + (1-x_{j+\frac{1}{2}})\Phi^n_i}{s^n}\right)\rho^n_{k;j} \\ &\quad + \frac{\tau^n}{h}\frac{x_{j-\frac{1}{2}}\Phi^n_o + (1-x_{j-\frac{1}{2}})\Phi^n_i}{s^n}\,\rho^n_{k;j-1}.\end{aligned}$$

Under the CFL condition (41), all the term are strictly positive, which give $\rho^{n+1}_{k;j}$ for all $j$ and $k$. Secondly, by summing all the balance equations all $\rho^{n+1}_{k;j}$, and noting that $x_{j+\frac{1}{2}} = x_{j-\frac{1}{2}} + h$, this leads to

$$\sum_k \rho^{n+1}_{k;j} \;=\; s^n + \tau^n\,(\Phi^n_i - \Phi^n_o)\,.$$

We exactly retrieve the explicit Euler discretization of the equation of evolution of the queue size. In particular, it does not depend on indice $j$.

## 5.2 Building an implicit scheme

To raise the problem of vanishing time steps in near zero queue size conditions, we now study an implicit version of the scheme. We decide not to implicit the terms $\Phi_i$ and $\Phi_o$ for practical reasons : in a block component system code, a global implicit system is quite more difficult to implement. We only want to implicit the "queue block" in order to build a black box with only explicit flux parameters as interfaces.

The numerical scheme has to be designed in such a way that it provides:

- Unconditional stability;
- Positivity;
- Accuracy : in the first section, we have seen that the design for positivity preserving can however produce a loss of accuracy, especially near zero queue size conditions;

- Conservation form : we consider schemes of the form
$$U_j^{n+1} = U_j^n - \frac{\tau^n}{h}\left(\Phi^{n,n+1}_{j+\frac{1}{2}} - \Phi^{n,n+1}_{j-\frac{1}{2}}\right)$$
with
$$\Phi^{n,n+1}_{j+\frac{1}{2}} = \Phi(U_j^n, U_j^{n+1}, U_{j+1}^n, U_{j+1}^{n+1};\ \Phi^n_{i;k}, \Phi^n_o);$$
- Compatibility condition of constant total density.

All these constraints have lead us to a particular scheme which is defined below. It is based on the previous construction in the case of the implicit ordinary differential Euler scheme of the first section.

**Definition 2** *We now define the mass fractions as*
$$Y^n_{k;j} = \frac{\rho^n_{k;j}}{s^n}.$$
*Suppose that*
$$s^0 = s(0) > 0,$$
$$Y^0_{k;j} > 0 \ \ \forall k, \ \ \sum_k \rho^0_{k;j} = s^0$$
*are given initial data. Let us consider the following implicit scheme:*

*If* $(\Phi^n_i - \Phi^n_o > 0)$, *then solve*

$$s^{n+1} = s^n + \tau^n(\Phi^n_i - \Phi^n_o), \tag{44}$$
$$\left(s^{n+1} + \frac{\tau^n}{h}\left(x_{j+\frac{1}{2}}\Phi^n_o + (1-x_{j+\frac{1}{2}})\Phi^n_i\right)\right) Y^{n+1}_{k;j} - \frac{\tau^n}{h}(x_{j-\frac{1}{2}}\Phi^n_o + (1-x_{j-\frac{1}{2}})\Phi^n_i)\, Y^{n+1}_{k;j-1} = s^n\, Y^n_{k;j} \ \forall j, k \tag{45}$$
$$\textit{with } Y^{n+1}_{k;0} = \frac{\Phi^n_{i;k}}{\Phi^n_i}, \tag{46}$$

*else* $(\Phi^n_i - \Phi^n_o \leq 0)$ *solve*

$$s^{n+1} = s^n + \tau^n(\Phi^n_i - \frac{s^{n+1}}{s^n}\Phi^n_o), \tag{47}$$
$$\left(s^{n+1} + \frac{\tau^n}{h}\frac{s^{n+1}}{s^n}x_{j+\frac{1}{2}}\Phi^n_o + \frac{\tau^n}{h}(1-x_{j+\frac{1}{2}})\Phi^n_i\right) Y^{n+1}_{k;j} - \frac{\tau^n}{h}\left(\frac{s^{n+1}}{s^n}x_{j-\frac{1}{2}}\Phi^n_o + (1-x_{j-\frac{1}{2}})\Phi^n_i\right) Y^{n+1}_{k;j-1} = s^n\, Y^n_{k;j} \ \forall j, k, \tag{48}$$
$$\textit{with } Y^{n+1}_{k;0} = \frac{\Phi^n_{i;k}}{\Phi^n_i}. \tag{49}$$

At each time step, we need to solve a linear system with a two-diagonal lower triangular matrix as iteration matrix. One of our main results is the following one:

**Proposition 5**

*The conservative scheme (44)-(49) is unconditionally $L^\infty$-stable. More precisely,*

$$0 \le Y_{k;j}^n \le 1 \quad \forall\, k, j, n, \tag{50}$$

$$\sum_{k=1}^{N} Y_{k;j}^n = 1 \quad \forall\, j, n, \tag{51}$$

$$s^n > 0, \quad |s^n - s^0| \le \max_{m\in\{1,\ldots,n\}} \tau^m \; n \sup_{t\in[0,t^n]} |\Phi_i(t) - \Phi_o(t)|. \tag{52}$$

*It is first order accurate in both space and time. The numerical effective outgoing rate is then given by the following expression:*

*If* $(\Phi_i^n - \Phi_o^n > 0)$, *then*

$$\Phi_{eo;k}^n = Y_{k;M}^{n+1}\,\Phi_o^n, \tag{53}$$

*else* $(\Phi_i^n - \Phi_o^n \le 0)$

$$\Phi_{eo;k}^n = Y_{k;M}^{n+1}\,\frac{s^{n+1}}{s^n}\Phi_o^n. \tag{54}$$

**Proof.** Let us consider the case $\Phi_i^n - \Phi_o^n > 0$. The numerical flux is here

$$\Phi_{j+\frac{1}{2}}^{n,n+1} = Y_{k;j}^{n+1}\,(x_{j+\frac{1}{2}}\,\Phi_o^n + (1 - x_{j+\frac{1}{2}})\Phi_i^n)$$

so that the scheme writes

$$\begin{aligned} s^{n+1}\,Y_{k;j}^{n+1} &= s^n\,Y_{k;j}^n \\ &- \frac{\tau^n}{h}\left[Y_{k;j}^{n+1}\,(x_{j+\frac{1}{2}}\,\Phi_o^n + (1 - x_{j+\frac{1}{2}})\Phi_i^n) - Y_{k;j-1}^{n+1}\,(x_{j-\frac{1}{2}}\,\Phi_o^n + (1 - x_{j-\frac{1}{2}})\Phi_i^n)\right]. \end{aligned}$$

The upstream boundary condition ($j = 0$) at $x_{\frac{1}{2}} = 0$ involves the compatibility condition

$$\Phi_{\frac{1}{2}}^{n,n+1} = Y_{k;0}^{n+1}\,\Phi_i^n = \Phi_{i;k}^n$$

which gives the expected value for $Y_{k;0}^{n+1}$ with $Y_{k;0}^{n+1} \ge 0$. For any value of $\tau^n$, it is easy to prove by recurrence that $Y_{k;j}^{n+1} \ge 0$. Let us now prove that $\sum_k Y_{k;j}^{n+1} = 1$, which will also prove that $Y_{k;j}^{n+1} \leq 1$ for all $k$. For that, we define the property

$$\mathcal{P}_m^n \;=\; \{\sum_{k=1}^{N} Y_{k;j}^p = 1,\ \forall j \leq m,\ \forall p \leq n\}.$$

From the assumptions on initial data, $\mathcal{P}_M^0$ is true. Suppose now that $\mathcal{P}_j^n$ is true. Let us show that $\mathcal{P}_1^{n+1}$ is true. Because of the upstream conditions, and using (44), the scheme writes at $j=1$

$$\left(s^n + \frac{\tau^n}{h}\,\Phi_i^n\right) Y_{1;k}^{n+1} - \frac{\tau^n}{h}\,\Phi_{i;k}^n \;=\; s^n Y_{k;1}^n.$$

Summing up all the equations over $k$ gives

$$\left(s^n + \frac{\tau^n}{h}\,\Phi_i^n\right) \sum_k Y_{1;k}^{n+1} \;=\; s^n \;+\; \frac{\tau^n}{h}\,\Phi_i^n$$

which proves the assertion. Suppose next that $\mathcal{P}_j^{n+1}$, $j \geq 1$ is true. We want to prove $\mathcal{P}_{j+1}^{n+1}$. Again, writing the scheme for the cell $I_{j+1}$, summing up over the $k$s, and remarking that $s^{n+1} + \tau^n(\Phi_o^n - \Phi_i^n) = s^n$ gives the result.

In the case $(\Phi_i^n - \Phi_o^n \leq 0)$, the numerical flux is

$$\Phi_{j+\frac{1}{2}}^{n,n+1} \;=\; Y_{k;j}^{n+1}\left(\frac{s^{n+1}}{s^n}\,x_{j+\frac{1}{2}}\,\Phi_o^n \;+\; (1 - x_{j+\frac{1}{2}})\,\Phi_i^n\right).$$

The boundary condition imposes at $x_{\frac{1}{2}} = 0$

$$\Phi_{\frac{1}{2}}^{n,n+1} \;=\; Y_{k;0}^{n+1}\,\Phi_i^n \;=\; \Phi_{i;k}^n.$$

The proof follows the same line of argument as above. Finally, it is easy to obtain estimate (52). This shows that the queue size does not explode at the discrete point of view provided that the interface fluxes are bounded.
That ends the proof.

**Remark 4** *Because we have $N$ variables in the system, we only need to solve $N$ equations. That means that, in practice, we only consider $(N-1)$ equations on mass fractions. The last one, let say $Y_{N;j}^{n+1}$ is directly obtained by means of the operation*

$$Y_{N;j}^{n+1} \;=\; 1 \;-\; \sum_{k=1}^{N-1} Y_{k;j}^{n+1}.$$

*That involves to select a so-called "slave component", but it has the advantage not to cumulate float number rounding errors. This class of numerical scheme has close connections with those defined for Computational Fluid Dynamics in the case of multicomponent reactive flows.*

# 6 Implicit scheme for the multicomponent shared resource problem

The numerical discretization of system (30)-(33) follows same arguments as before. First, the ODE solver on $s(t)$ needs to respect positivity. We propose to use

$$s^{n+1} \;=\; s^n \;+\; \tau^n \left( \Phi_i^n \;-\; \rho_M^n \, \frac{\Phi_0^n}{{s^n}^2} \; s^{n+1} \right). \tag{55}$$

This implicit scheme is well-posed and does not need fixed point algorithm. Secondly, the propagation velocity equal to

$$\frac{\Phi_o(t)}{s(t)}$$

tends to infinity when the total load mass $s(t)$ tends to zero. If we use an explicit upwind scheme, the stability criterion given by the Courant number becomes too restrictive for time advance. We then need to use an implicit scheme. Because here we deal with a classical free transport equation, we can use the classical upwind implicit scheme, namely,

$$\rho_{k;1}^{n+1} \;\; = \;\; \rho_{k;1}^n - \frac{\tau^n}{h} \left( \frac{\Phi_o^n}{s^{n+1}} \, \rho_{k;1}^{n+1} \; - \; \Phi_{i;k}^n \right), \tag{56}$$

$$\rho_{k;j}^{n+1} \;\; = \;\; \rho_{k;j}^n - \frac{\tau^n}{h} \, \frac{\Phi_o^n}{s^{n+1}} \left( \rho_{k;j}^{n+1} - \rho_{k;j-1}^{n+1} \right), \quad j = 2...M. \tag{57}$$

Then we have the result

**Proposition 6** *The numerical scheme (55), (56), (57) is upwind, unconditionally $L^\infty$-stable. More it preserves positivity of the total load mass s:*

$$s^n \;>\; 0 \;\; \forall n \in \mathbb{N}.$$

# 7 Numerical experiments

## 7.1 Multicomponent FIFO

Let us consider a FIFO queue fed by two fluid sources indexed 1 and 2. Suppose the initial queue size is 100. The two initial mass fractions are constant equal to 0.5. Suppose a constant outgoing rate $\Phi_o = 1000$ (say packets per second for the dimension). We choose a piecewise constant time function for $\Phi_i(t)$:

$$\Phi_i(t) \;=\; \max\left(0,\ 1000 + 1000\left(\mathrm{sgn}(\sin(10\pi t)) - 0.4\right)\right).$$

This will lead to a piecewise linear queue size. For space discretization, we choose $h = \frac{1}{40}$. For time discretization, we choose $\tau = 10^{-3}$. This corresponds to the response time, but we could of course choose greater time steps. Finally, for upstream boundary conditions, we impose

$$Y_1(0,t) \;=\; 0.4 + 0.3 * \mathrm{sgn}(\sin(20\pi t)),\ Y_2(0,t) = 1 - Y_1(0,t).$$

The numerical results are presented below. In figure 5 is drawn the history of queue size, note that $s^n$ never goes down to zero. In figure 6 are presented mass fraction profiles at different times. In particular, the scheme does not degenerate for near zero queue size conditions. Finally, in figure 7 are plotted contour levels of the mass fraction $Y_1$ in the space-time domain. At the beginning of the run-time, we see a propagation of fronts at finite propagation velocity. Notice also the smearing of mass fractions fronts due to the large amount of numerical dissipation for this first order scheme. Other simulation have shown us that this effect of smearing becomes very important when large time steps are used. This invites us to develop more accurate numerical schemes of order 2 in both space and time. This will be the object of a future paper.

Secondly, one can notice between time $t = 0.6$ and $t = 1.5$ some “vertical” contour levels due to the fact that $s^n$ is sometimes very close to zero, so that the propagation velocities are almost infinite. But the numerical scheme does not breakdown. As in the scalar EDO case, due to float number truncation errors, we need to add a parameter $\epsilon$ as mentioned before in order not to create divide by zero errors.

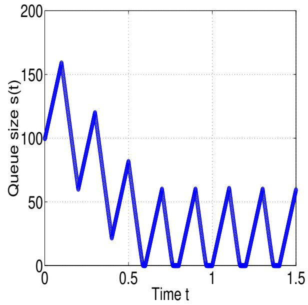


Figure 5: History of FIFO queue size. The numerical scheme preserves the positivity of the queue size.

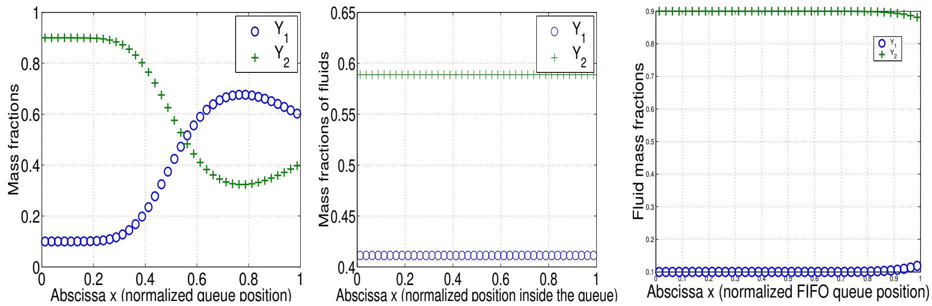


Figure 6: Mass fraction of fluids for respectively $t = 0.1$ $sec$ ($s = 158.4$), $t = 1$ $sec$ ($s = 10^{-14}$) and $t = 1.5$ $sec$ ($s = 59.4$). At the discrete level, the sum of mass fractions gives 1.

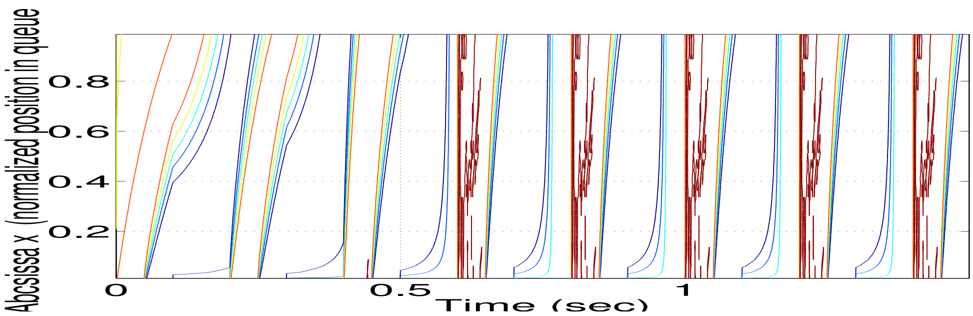


Figure 7: Isolines of mass fraction $Y_1$ in the space-time domain. This shows the transport effect, the smearing of discontinuities due to numerical dissipation and large propagation velocities for near zero queue size conditions

## 7.2 Multicomponent processor sharing

We here use the same expressions for incoming rate and upstream boundary conditions than those used for the multicomponent FIFO test case. For initial conditions, the mass fractions are again constant, equal to 0.5 within the space domain. Let us recall that the space variable $x \in [0, 1]$ gives a key indicator on the rate of progress : $x = 0$ means that the sent request begins to be processed; $x = 1$ means the end of processing for the request particle. For this test case, we use 50 a uniform grid made of 50 discretization points and a CFL number equal to 0.4. The numerical results are interesting. They show a chaotic behavior of the solutions for near bottleneck conditions. Even if the incoming source signal is periodic, this feature is lost into the field after a certain critical time. Secondly, this can be interpreted by the fact that the system that becomes saturated has characteristic times of propagation that becomes smaller that that characteristic time of periodicity for the source signal.
Although this has not been exposed in this paper, we here use a quasi second order accurate in space scheme for the results. Numerical experiment show that this is necessary to get sufficiently accurate results for variables of interest. The first order scheme would largely smear discontinuities and chaos to compute a too strongly smooth averaged field.

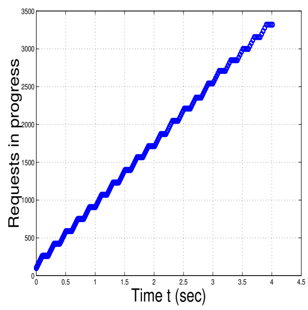


Figure 8: History of number of requests in progress. This shows the creation of a bottleneck due to the constantly increasing number of requests in progress

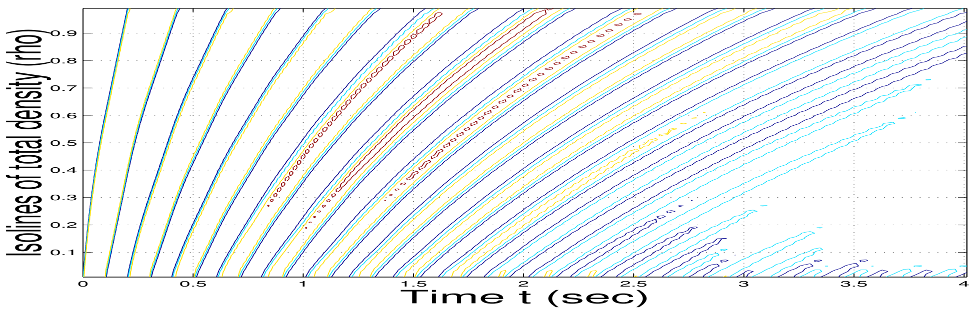


Figure 9: Level contour of total density in the space-time domain. Isolines give an idea of the "information trajectories" and show the slowing-down propagation velocity inside the system. Again, this shows the creation of a bottleneck

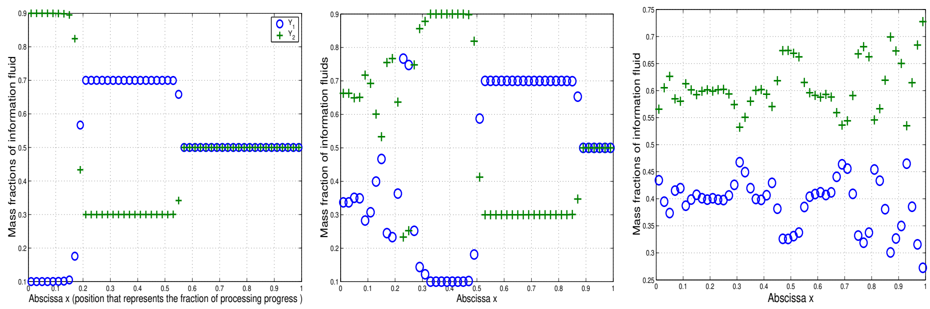


Figure 10: Mass fraction of fluids for respectively $t = 0.0871$ *sec* , $t = 0.1699$ *sec* and $t = 4.0120$ *sec*. Solutions rapidly develops chaotic behaviors. The piecewise constant feature of the source signal is lost after a certain time. This is due to the slowing down of the system and the resulting inadequacies between characteristic times of evolution. In this example, the ripple effect would be critical using fluid approaches like those exposed in [LGK$^+$99].

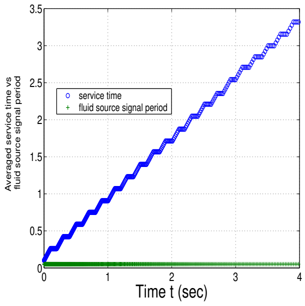


Figure 11: Average service time versus fluid source signal period (equal to 0.05 sec). Chaos explained by the ratio of characteristic times. The average service time grows up from value 0.1 sec to value 3.3 sec.

## 7.3 Three component feedback system

### 7.3.1 Presentation of the device and parameters

We are now interested in simulating a more complex three-fluid system with three independent FIFO queues followed by a processor sharing system intended to treat three fluids simultaneously. Each queue number $k$ has constant crude departure rate equal to $\Phi_o = 1000$ and respective maximum queue size equal to $s_{k;max} = 100$.

The processor sharing system has a crude constant processing rate equal to $\Phi_o$. It is supposed that there exists an upper bound $s_{sh;max}$ for the maximum number of requests into progress at the same time. This bound limits the bottleneck effect discussed in the previous example. An external scalar source fluid $\Phi_i(t)$ is connected to the input port number 1. We here choose for this case

$$\Phi_i(t) \;=\; \max\left(0,\; 1000 + 1000\ \mathrm{sgn}(\sin(2\pi t)) - 0.4\right).$$

We then introduce some feedback process in the system: let us connect the output port number 1 with the input port number 2 and the output port number 2 with the input port number 3.

The "device" is resumed at figure 12. It coarsely models a server that is aimed at fulfilling three successive independent tasks in the value chain of information.

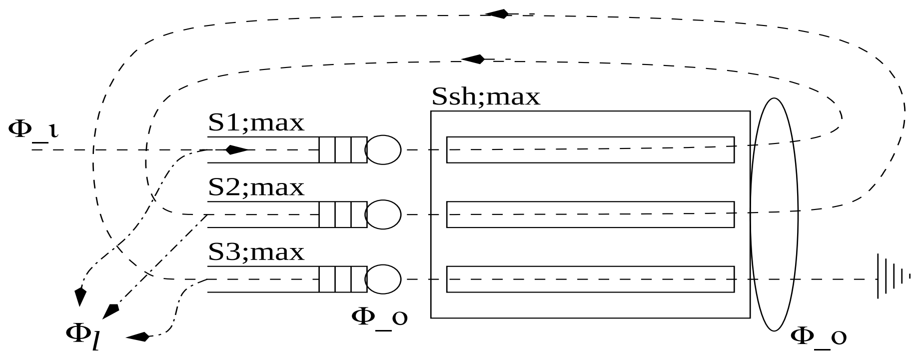


Figure 12: Three component feedback system made of three FIFO queues and a three component processor sharing system at rate $\Phi_o$. The external source rate is $\Phi_i$, Possibly, some flux $\Phi_l$ can be lost upstream the queues because of the limited sizes.

### 7.3.2 Flux limiting process

For the sake of simplicity, suppose first that the interaction between the block components is null. If for a queue $k$, the maximum size $s_{k;max}$ is reached, then the effective incoming rate is limited to the departure rate $\Phi_o$ as exposed in section 2. In this case, there exists some "lost" (or redirected) flux $\Phi_{l;k}(t)$ at each port number $k$ at time $t$. We will denote by $\Phi_l(t)$ the sum of partial lost fluxes $\Phi_{l;k}(t)$.
Inversely, if the queue size attains zero, then the effective queue departure rate $\Phi_{oe;k(t)}$ is limited to the arrival one $\Phi_{i;k}(t)$. For the processor sharing system, if the upper bound is reached, then that involves an overall effective arrival rate $\Phi_{ie;sh}$ that is limited to $\rho(1,t)\Phi_o/s(t)$.

Linking now the block components together, we see can there possible exists a discontinuity of rate between the queues and the shared processing system. With no additional flux limitation, this should involve a violation of the conservation principle. To overcome this problem, we have to make additional work on the interface flux. As an illustration, let us suppose that the queue 1 is full, the queue 2 is empty, the queue 3 is full and the processor sharing system has attained its maximum limit of request under processing and stays saturated. In that case,

$\Phi_{ie;sh}$ is imposed, equal to

$$\Phi_{ie;sh}(t) = \frac{\rho(1,t)\Phi_o}{s(t)}$$

and defines the total interface rate. Notice that this boundary conditions depends on downstream conditions, which is not usual in Computational Fluid Dynamics. Secondly, the queue component departure rates have to adapt themselves to the value of $\Phi_{ie;sh}$ for rate continuity conditions. Without interaction, we should have $\Phi_{o;1} = \Phi_o$, $\Phi_{o;2}(t) = \Phi_{i;2}(t)$ (limited flux) and $\Phi_{o;3} = \Phi_o$. Then the effective queue outgoing rates become,

$$\Phi_{oe;k}(t) = \frac{\Phi_o}{\sum_{m=1}^{3} \Phi_{o;m}(t)} \Phi_{ie;sh}(t) \quad \text{for } k = 1, 2,$$

$$\Phi_{oe;2}(t) = \frac{\Phi_{i;2}(t)}{\sum_{m=1}^{3} \Phi_{o;m}(t)} \Phi_{ie;sh}(t)$$

This is resumed on figure 13.

Notice that queue departure rates now depend on time, which makes hard for

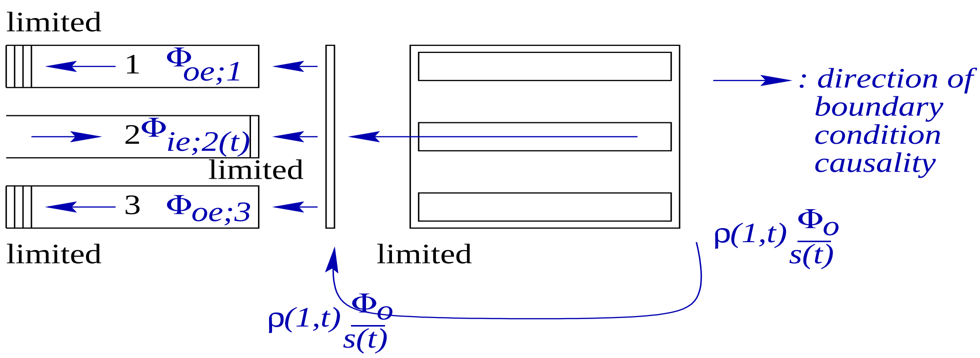


Figure 13: Flux limiting process due to limited range on queue sizes and maximum number of requests under processing. The arrows give an idea of the causality of boundary conditions and limitation processes.

analytical solutions (that suppose piecewise constant output rates) to deal with such conditions. About range preserving, we see first that $\Phi_{oe;2} < \Phi_{i;2(t)}$ under the given assumptions. Consequently, the size of queue 2 cannot become negative. About queues 1 and 3, because outgoing rates are limited and that $\Phi_{oe;k} < \Phi_o$, we need stronger upstream flux limitations on arrival rates for range preserving using updated outgoing rates. This draws the chain of boundary condition causality, as explained on figure 13.

Of course, this limiting process has to be translated and adapted at the discrete level. Because this topic is not at the aim of this paper and is quite complicated to present rapidly, we prefer not to expose it here. But we claim that the discrete problem can be well-posed, and the feasibility is here proved by the following results. The complete analysis of such upstream-downstream boundary conditions will be the subject of a future paper.

### 7.3.3 Results report and interpretation

For the discretization of the processor sharing system, we use a 50 points grid discretization and an explicit second order accurate in space version of the scheme. The time dependent time step is calibrated on a CFL condition for stability. We use $CFL = 0.4$. We make a simulation between time 0 and $T = 6$sec.

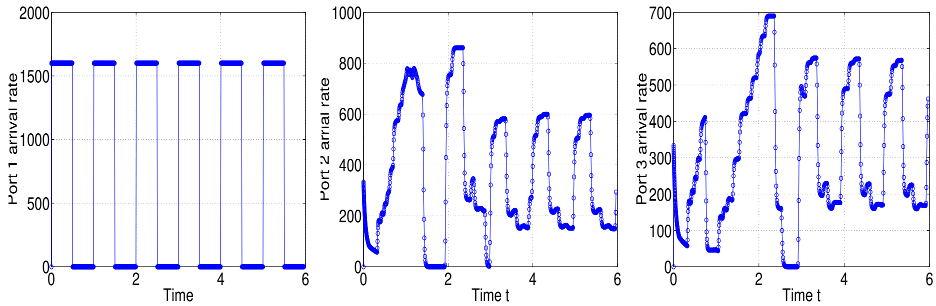


Figure 14: Fluid component arrival rate upstream the queues. The first one is the external rate. The last ones come from feedback loops from sharing system output ports number 1 and 2.

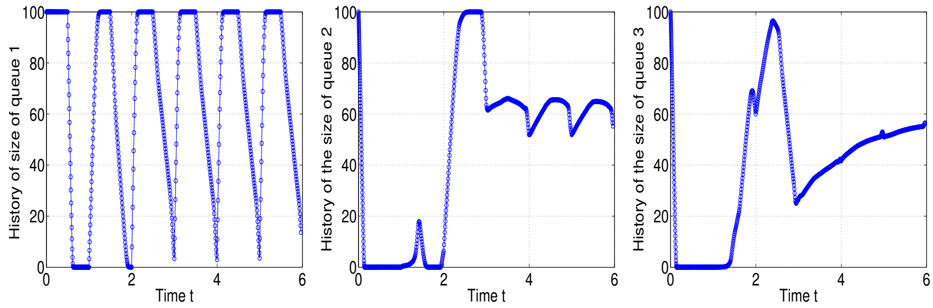


Figure 15: Time history of queue sizes for each queue. The size ranges are well-respected at the discrete point of view.

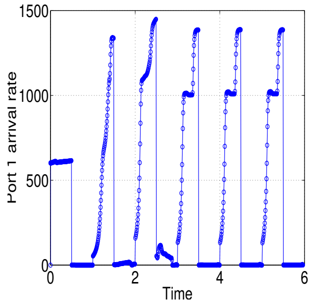


Figure 16: History of total lost rate upstream the queues

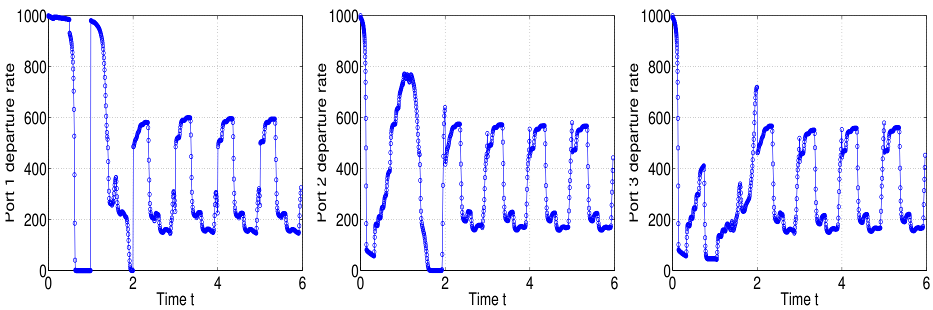


Figure 17: Time history of departure rates that enter into the shared processor system. The rates depend on both upstream and downstream conditions. The interface flux is built as explained above.

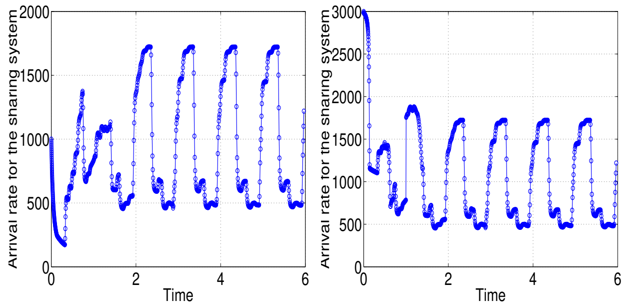


Figure 18: Total arrival and departure rates for the shared processing system. Notice that although the crude rate of the processor is 1000 units/sec, the processor sharing policy allows greater departure rate. But of course one can see that the average rate turns around the crude one.

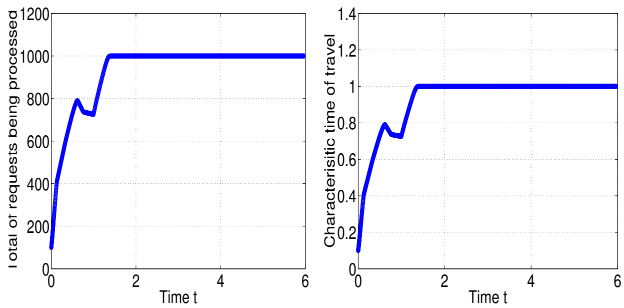


Figure 19: Time history of actual requests under processing and corresponding characteristic time of travel. Because of the policy of limited maximum requests being processed instantaneously, the characteristic time of system travelling is bounded by 1000/1000=1 (the service time would be $10^{-3}$sec for an alone request).

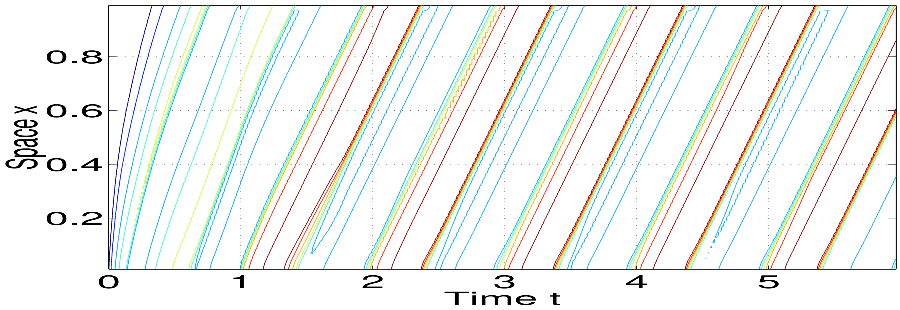


Figure 20: Contour levels of total density at final time. The trajectories become straight become of the upper bound on the number of requests in progress (saturated case).

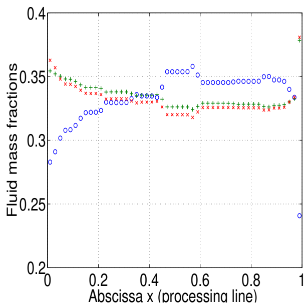


Figure 21: Fluid mass fractions on the "processing line" at final time.

# 8 Concluding remarks and future works

In this paper, we have proposed a family of fluid models aimed at simulating large scale communication networks. Fluid models are designed in order to outperform event and packet-based simulators at the computational time point of view. One of the very special features is that the models lead to partial differential equations. At our knowledge, this is the first time that PDE are used for computer networks simulation and queuing theory. Another feature is that the algorithm complexity does not depend on the so-called ripple effect, which make our models very good candidates for simulating large scale networks. Finally, the extension of multi class fluid sources is natural and allows to consider several fluids of interest for QoS and multi-criteria optimization. That will be the object of a future paper.

The main goal of this paper has been to show the feasibility of such models. For that, we have only here modelled "classical" nodes like FIFO, LIFO, multiplexer or processor sharing. But, we claim that more complex buffer devices (like those exposed in [DKS90],[AG93],[KM98]) can be modelled using similar arguments. This will also be the object of future papers.

# 9 Acknowledgements

This work is performed under the contract agreements between Ecole Normale Supérieure de Cachan, université Cergy-Pontoise and the company Simflux (loi sur l'innovation). I thank Professor J.M. Ghidaglia for having supported this work and for his fruitful comments. I also thank Professor S. Fdida for having provided me recent references on this research theme.

# References

[AG93] K.P. Abhay and R.G. Gallager. A generalized processor sharing approach to flow control in integrated service network : the single node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.

[AMS82] D. Anick, D. Mitra, and M. M. Sondhi. Stochastic theory of a data-handling system with multiple sources. *Bell System Technical Journal*, 61(8):1871–1894, October 1982.

[Ben63] V.E. Beněs. *General Stochastic Processes in the Theory of Queues.* Addison-Wesley, 1963.

[DKS90] A. Demers, S. Keshav, and S. Shenkar. Analysis and simulation of a fair queueing algorithm. *Internet Res. and Exper.*, 1, 1990.

[GR91] E. Godlewski and P.A. Raviart. *Hyperbolic systems of conservation laws.* Mathématiques et Applications. Ellipses, Paris, 1991.

[Joh97] Homayoun Yousefi'zadeh John. A fair neural-based dynamic buffer management scheme for improving packet loss performance in a class of queuing systems with self-similar characteristics. *http://citeseer.nj.nec.com/297149.html*, 1997.

[KM98] K. Kumaran and D. Mitra. Performance and fluid simulations of a novel shared buffer management system. In *Proceedings of IEEE INFOCOM 98*, pages 1449–1461, San Francisco, March 1998.

[KSCK96] G. Kesidis, A. Singh, D. Cheung, and W.W. Kwok. Feasibility of fluid event-driven simulation for ATM networks. In *Proceedings of IEEE Globecom'96*, Londres, November 1996.

[LFG+01] B. Liu, D. Figueiredo, Y. Guo, J. Kurose, D. Towsley, and W Gong. A study of networks simulation efficiency: Fluid simulation vs. packet-level simulation. In *Proceedings of IEEE INFOCOM 2001*, April 2001.

[LGK+99] B. Liu, Y. Guo, J. Kurose, D. Towsley, and W. Gong. Fluid simulation of large scale networks: issues and tradeoffs. In *Proceedings of the*

*International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'99)*, volume IV, pages 2136–2142, Las Vegas, 1999.

[NGJ99] D. Nicol, M. Goldsby, and M. Johnson. Fluid-based simulation of communication networks using ssf. In *Proceedings 1999 European Simulation Symposium*, Erlangen-Nuremberg, Germany, October 1999.

[Rei58] E. Reich. On the integrodifferential equations of takacs. *Annals of Mathematical Statistics*, 29:563–570, 1958.

[Sha91] L.F. Shampine. *Numerical Solutions of Ordinary Differential Equations.* Chapman & Hall, New York, 1991.

[YG98] A. Yan and W. Gong. Time-driven fluid simulation for high-speed networks with flow-based routing. In *Proceedings of the Applied Telecommunications Symposium, Boston, MA*, volume 45, pages 153–158, 1998.

[YG99] A. Yan and W. Gong. Time-driven fluid simulation for high-speed networks. *IEEE Transactions on Information Theory*, 45(5):1588–1599, 1999.