A MATHEMATICAL MODEL OF ASYNCHRONOUS DATA FLOW IN PARALLEL COMPUTERS *

RICHARD C. BARNARD †, KAI HUANG ‡, AND CORY HAUCK§

Abstract. We present a simplified model of data flow on processors in a high performance computing framework involving computations necessitating inter-processor communications. From this ordinary differential model, we take its asymptotic limit, resulting in a model which treats the computer as a continuum of processors and data flow as an Eulerian fluid governed by a conservation law. We derive a Hamilton-Jacobi equation associated with this conservation law for which the existence and uniqueness of solutions can be proven. We then present the results of numerical experiments for both discrete and continuum models; these show a qualitative agreement between the two and the effect of variations in the computing environment's processing capabilities on the progress of the modeled computation.

Key words. data flow, high-performance computing, asymptotic approximation, conservation laws, Hamilton-Jacobi equation

AMS subject classifications. 35L65, 93A30, 70H20, 41A60

1. Introduction.

It has been well-established that current and future generations of extreme scale computers have achieved and, for the foreseeable future, are expected to achieve increases in performance via greater levels of parallelism at multiple levels — e.g., within the processors as well as increasing the number of processors and nodes —as opposed to increases in clock speeds, which are expected to remain relatively flat. Additionally, extremely concurrent codes, involving dynamic parallelism and greater degrees of asynchronous parallel executions, are increasingly needed to leverage this large scale parallellism [15, 29].

As machine improvements depend on increasingly complex architectures and as additional constraints on system development and planning (such as power consumption [15]) arise, a need for predictive, quantitative models of computational performance will grow greater. Previously developed modeling tools such as LogP [12, 13] result in easily evaluated models which can prove difficult to extend and modify. Alternatively, PRAM models have been used as abstractions of codes; these however have scalability issues due to the complexity of simulating them [25]. Other modern tools [23, 24] are similarly still limited to fine-grain simulations of at most a few dozen nodes, again due to their computational complexity during simulations.

Core counts are now in the hundreds of thousands and millions on machines in the TOP500 list of supercomputers; node counts consistently are in the thousands [1]. Such numbers mean that fine-grained simulation tools (such as those listed above) are

Funding: This research is sponsored by the Office of Advanced Scientific Computing Research; U.S. Department of Energy. The work was performed at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725.

†Western Washington University, Bellingham, WA (rick.barnard@wwu.edu, https://cse.wwu.edu/mathematics/barnarr3).

‡Michigan State University, East Lansing, MI (huangk18@msu.edu).

§Oak Ridege National Laboratory, Oak Ridge, TN (hauckc@ornl.gov, https://www.csm.ornl.gov/~hfd/).

incapable of describing large-scale phenomena; essentially the simulation tools begin to require computational resources beyond those of the systems they are simulating. Alternative approaches have been proposed to address these issues: miniapp codes can mimic key features of the performance of exascale codes with a much smaller codebase [16]. Aspen, a framework for performance modeling [28, 31], uses a domain specific language which encodes both abstracted features of machines hardware and specific software applications to provide coarse-grained simulations. However, these suffer from the need to develop specialized simulation codes which can be problem dependent, resulting in possibly labor-intensive tools. A workflow modeling apporach, Pegasus, has been developed to model workflows using a graph-theoretic perspective to detect and manage anamolies in the computing environment [14].

We propose developing a macroscopic model of extreme scale computers which views such computing environments in a continuum framework. Such a model has several potential benefits: in addition to being computationally tractable, it will open up the possibility of using the theoretical tools of partial differential equations to understand and control the performance of high-performance computing systems. Specifically, our goal is to derive a fluid-limit model of data flow — which can be described by a partial differential equation — from a simplified deterministic model of data processing and flow in an extreme scale computer with interprocessor communications and asynchronous executions. Fluid models, beyond their obvious utility in physical systems, have been used to model flows in networks, such as vehicular traffic flows [3], supply chains [2], and gas networks [4, 8]. In particular, as discussed in [22] and [2], such fluid models lie at the end of a hierarchy of models which begin with microscopic or discrete models. That is, similar to the derivation of physical fluid laws from many body physics, one may derive continuum-level flow equations from discrete-level models of the dynamics of agent interactions. With such a model, standard numerical simulation tools and analytical methods may be brought to bear for studying large-scale phenomena in extreme-scale computing.

We begin in Section 2 with a microscopic model of a network of processors performing a multi-stage computational task which necessitates inter-processor communications. In Section 3, we derive a formal asymptotic limit of this agent-based model as the scale of the system increases, resulting in an Eulerian fluid flow model. Along with the resulting nonlinear conservation law, we present a related Hamilton-Jacobi equation and establish the existence of solutions in Section 3. In Section 4, we present the results of numerical experiments to show agreement between the microscopic and fluid models and then illustrate the behavior of solutions under heterogeneous computing layouts.

2. The discrete model. In this section, we introduce the microscopic model, which is based on a highly simplified, deterministic, semi-discrete ordinary differential equation (ODE). We imagine the computer as a network of processors $\{\mathcal{P}_i\}_{i=1}^{i^{\max}}$ that are arranged in a one-dimensional, periodic lattice. The computer is assigned a computational job involving a sequence of $k^{\max}$ tasks which are identical in the sense that each one takes the same effort to complete. This computational job is divided by distributing data amongst processors. We denote by $q_{i,k}(t)$ the amount of data in $\mathcal{P}_i$ that sits in stage k at time t.

2.1. Conservation law. The dynamics of $q_{i,k}$ are given by a conservation law of the form

$$\dot{q}_{i,k}(t) = F_{i,k-1}(t) - F_{i,k}(t), \quad k = 1, \dots, k^{\max}, \quad i = 1, \dots i^{\max}, \tag{2.1}$$

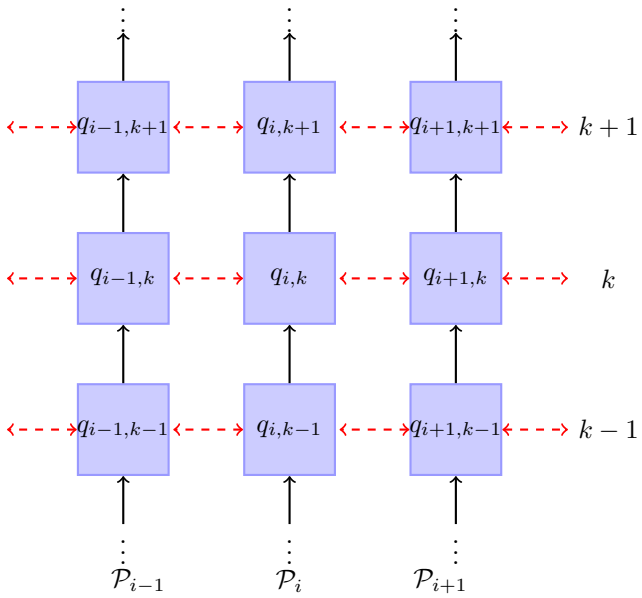


Fig. 1: Schematic of network of processors. Dashed lines denote inter-processor communications

where $F_{i,k}$ ($i = 1, \ldots, i^{\max}$, $k = 1, \ldots, k^{\max}-1$) is the rate of data moving in processor i from stage k to $k+1$, referred to as the *throughput*. At the first stage $k = 1$, $F_{i,0}$ ($i = 1, \ldots, i^{\max}$) is the rate of data being loaded into processor i to be processed, referred to as the *inflow*, and at the final stage, $F_{i,k^{\max}}$ ($i = 1, \ldots, i^{\max}$) is the rate of data completing the final stage of the job, referred to as the *outflow*. Equation (2.1) implies that the data in each processor is neither created or destroyed, only moved in and out of the processor or in between stages; that is,

$$\frac{d}{dt}\left(\sum_{k=1}^{k^{\max}} q_{i,k}\right) = F_{i,0} - F_{i,k^{\max}}. \tag{2.2}$$

A key aspect of the model is that it does not separately track data that moves between processors; instead the effects of communication delays will be incorporated directly into the definition of the throughputs.

A fundamental quantity of interest in the discrete model is $Q_{i,k}(t)$, which is defined as the amount of data at time t that has gone through the first $k-1$ stages of $\mathcal{P}_i$. For each $t \geq 0$,

$$Q_{i,k}(t) = \Big(\sum_{j=k}^{k^{\max}} q_{i,j}(t)\Big) + \int_0^t F_{i,k^{\max}}(s)ds. \tag{2.3}$$

2.2. Processor throttling. We now turn to specifying the form of $F_{i,k}$. In the absence of throttling, each processor $\mathcal{P}_i$ moves data between stages at a rate $a_i \geq 0$,

which we refer to as the *maximum throughput*. For the purposes of the current paper, we assume that a_i is given. In practice, it must be determined from experiments, fine-scale models, or a combination of both. It may also depend on k, although for simplicity, we assume here that it does not. Throttling is said to occur whenever $F_{i,k}(t) < a_i$; this happens for one of two reasons.

1. **Self-throttling:** Given an amount of data $q_{i,k}$ to be processed at stage k in processor $\mathcal{P}_i$, we define the self-throttling function

$$v_1(q_{i,k}; q_*) = \max\left\{0, \min\left\{1, \frac{q_{i,k}}{q_*}\right\}\right\}. \tag{2.4}$$

   Clearly if no data is available to be processed, then $F_{i,k} = 0$. Furthermore, we assume that if the amount of data to be processed drops below a certain threshold $q_* > 0$, then $\mathcal{P}_i$ cannot maintain the throughput a_i and the throughput is reduced.

2. **Neighbor throttling:** As the computational task is not entirely parallel across processors, $\mathcal{P}_i$ requires sufficient information from its neighbors to perform task k at full throughput. The neighbor throttling function v_2 models this dependence. It gives the amount of available data on $\mathcal{P}_i$ at stage k

$$v_2(q_{i,k}, \Delta_{i+1,k}, \Delta_{i-1,k}; \beta) = \min\left\{q_{i,k}, \frac{1}{\beta}\max\{\Delta_{i+1,k}, 0\}, \frac{1}{\beta}\max\{\Delta_{i-1,k}, 0\}\right\}. \tag{2.5}$$

   Here $\Delta_{i\pm1,k}$ denotes the data on the right/left neighbor which is available to be used by $\mathcal{P}_i$ to process $q_{i,k}$. The parameter $\beta \in (0, 1]$ allows for the possibility that computations do not rely in a one-to-one fashion upon the availability of data from neighbors. If $\Delta_{i\pm1,k} = 0$ the processing of data stops due to the absence of a necessary component of the computational task and so $F_{i,k} = 0$. Alternatively, if both $\Delta_{i+1,k}$ and $\Delta_{i-1,k}$ exceed $\beta q_{i,k}$, then $\mathcal{P}_i$ has sufficient data from its neighbors to process $q_{i,k}$ and no throttling occurs. The data from the left/right neighbor which is available for processing at stage k is given by

$$\Delta_{i\pm1,k} = Q_{i\pm1,k} - Q_{i,k+1} = Q_{i\pm1,k} - (Q_{i,k} - q_{i,k}). \tag{2.6}$$

   The data on each neighbor must have completed the same stage for it to be available; additionally, this data is not reused on $\mathcal{P}_i$ for the same stage. This means that the data available to be used from the neighbors can be written as above and so the amount of data available to be processed on $\mathcal{P}_i$ at stage k is given by

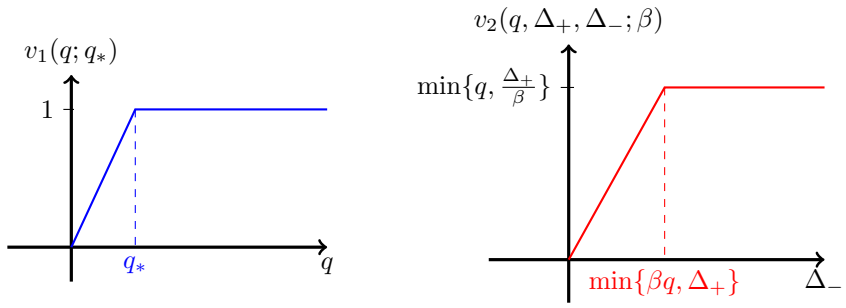
$$v_2\big(q_{i,k}, Q_{i+1,k} - Q_{i,k} + q_{i,k}, Q_{i-1,k} - Q_{i,k} + q_{i,k}; \beta\big). \tag{2.7}$$

The throughput $F_{i,k}$ is a composition of the throttling functions v_1 and v_2:

$$F_{i,k} = a_i v_1\Big(v_2\big(q_{i,k}, Q_{i+1,k} - Q_{i,k} + q_{i,k}, Q_{i-1,k} - Q_{i,k} + q_{i,k}; \beta\big); q_*\Big). \tag{2.8}$$

At first glance, this definition of $F_{i,k}$ appears circular since it depends on $Q_{i,k}$, which in turn depends on $F_{i,k^{\max}}$. However, as a consequence of the conservation law (2.2),

$$\int_0^t F_{i,k^{\max}}(s)ds = \int_0^t F_{i,0}(s)ds + \sum_{j=1}^{k^{\max}} q_{i,j}(0) - \sum_{j=1}^{k^{\max}} q_{i,j}(t) \tag{2.9}$$



(a) $v_1(q;q_*)$ for a given q_* (b) $v_2(q,\Delta_+,\Delta_-;\beta)$ for a given q and $\Delta_+ \geq 0$.

Fig. 2: Throttling functions v_1 and v_2, defined in (2.4) and (2.5), respectively

Thus to complete the model, we need only prescribe initial data $q_{i,k}(0)$ and the inflow $F_{i,0}$. To prescribe the inflow, we specify $q_{i,0}$ and then let $F_{i,0}$ be evaluated according to (2.8).

PROPOSITION 2.1. *The system (2.1) with (i) throughput $F_{i,k}$ defined in (2.8) for $i = 1,\dots,i^{\max}$ and $k = 1,\dots,k^{\max}$; (ii) prescribed initial data $q_{i,k}(0)$ for $i = 1,\dots,i^{\max}$ and $k = 1,\dots,k^{\max}$; and (iii) prescribed inflow data $F_{i,0}$ for $i = 1,\dots,i^{\max}$ and $t \geq 0$ has a unique solution for all $t \geq 0$. Moreover, if $q_{i,k}(0) \geq 0$ for all $i = 1,\dots,i^{\max}$ and $k = 1,\dots,k^{\max}$, then $q_{i,k}(t) \geq 0$ for all $t \geq 0$ and $i = 1,\dots,i^{\max}$ and $k = 1,\dots,k^{\max}$.*

Proof. Since $F_{i,k}$ is globally Lipschitz in its arguments for every i,k, standard ODE theory (see, for example Theorem III.VI of [32]) implies the existence of a unique solution. Moreover, it is clear from (2.8) that $0 \leq F_{i,k} \leq a_i v_1(q_{i,k};q_*)$. Hence according to (2.1),

$$\dot{q}_{i,k}(t) \geq -a_i v_1(q_{i,k};q_*). \tag{2.10}$$

Standard comparison results for ordinary differential equations imply then that $q(t) \geq 0$. (See, for example, Lemma 1.2 of [30] and for comparison, use the zero function, which satisfies (2.10) as an equality.) ∎

3. The continuum model. In this section, we derive a continuous model that is formally accurate in the limit as $i^{\max}$ and $k^{\max}$ tend to infinity. We assume, in taking this limit, that the job performed by the computer is fixed – that is, the total amount of work does not change. For given $i^{\max}, k^{\max}$, we define the quantities:

$$\delta := (k^{\max})^{-1}, \quad \varepsilon := (i^{\max})^{-1}, \quad \eta := \frac{\varepsilon}{\delta}. \tag{3.1}$$

Here, δ is the fraction of the work done in each stage and ε is the average amount of data in a processor. Finally, η is simply the ratio of $k^{\max}$ and $i^{\max}$ which will be of use in the following analysis.

3.1. Formal Derivation. To derive a continuum model, we first express the ODE (2.1) in terms of the following $O(1)$ quantities:

(3.2)
$$r_* := \frac{q_*}{\varepsilon\delta}\,, \quad r_{i,k} := \frac{q_{i,k}}{\varepsilon\delta}\,, \quad R_{i,k} := \frac{1}{\varepsilon}Q_{i,k}\,, \quad D^{\pm}_{i,k} := \pm\frac{R_{i\pm1,k} - R_{i,k}}{\varepsilon}\,, \quad \alpha_i := \frac{a_i}{\varepsilon}\,.$$

In terms of these rescaled quantities, the neighbor throttling function as can be written as

(3.3)
$$\begin{aligned} &v_2\big(q_{i,k}, Q_{i+1,k} - Q_{i,k} + q_{i,k}, Q_{i-1,k} - Q_{i,k} + q_{i,k}; \beta\big) \\ &\quad = \min\left\{\varepsilon\delta r_{i,k}, \frac{1}{\beta}\max\{\varepsilon R_{i+1,k} - \varepsilon R_{i,k} + \varepsilon\delta r_{i,k}, 0\}, \frac{1}{\beta}\max\{\varepsilon R_{i-1,k} - \varepsilon R_{i,k} + \varepsilon\delta r_{i,k}, 0\}\right\} \\ &\quad = \varepsilon\min\left\{\delta r_{i,k}, \frac{1}{\beta}\max\{R_{i+1,k} - R_{i,k} + \delta r_{i,k}, 0\}, \frac{1}{\beta}\max\{R_{i-1,k} - R_{i,k} + \delta r_{i,k}, 0\}\right\} \\ &\quad = \varepsilon\delta\min\left\{r_{i,k}, \frac{1}{\beta}\max\{\eta D^+_{i,k} + r_{i,k}, 0\}, \frac{1}{\beta}\max\{-\eta D^-_{i,k} + r_{i,k}, 0\}\right\}. \end{aligned}$$

Therefore,

(3.4)
$$\begin{aligned} &v_1\Big(v_2(q_{i,k}, v_2(q, Q_{i+1,k} - Q_{i,k} + q_{i,k}, Q_{i-1,k} - Q_{i,k} + q_{i,k})); q_*\Big) \\ &\quad = \min\left\{1, \frac{\min\left\{r_{i,k}, \frac{1}{\beta}\max\{\eta D^+_{i,k} + r_{i,k}, 0\}, \frac{1}{\beta}\max\{-\eta D^-_{i,k} + r_{i,k}, 0\}\right\}}{r_*}\right\}. \end{aligned}$$

With (3.4) in mind, we define the rescaled throttling functions

(3.5a)
$$w_1(r, r_*) = \max\left\{0, \min\left\{1, \frac{r}{r_*}\right\}\right\}.$$

(3.5b)
$$w_2\left(r, D^-, D^+; \eta, \beta\right) = \min\left\{r, \frac{1}{\beta}\max\{\eta D^+ + r, 0\}, \frac{1}{\beta}\max\{\eta D^- + r, 0\}\right\}$$

and the composite function

(3.6)
$$w\left(r, D^-, D^+; r_*, \alpha, \eta, \beta\right) := \alpha w_1\left(w_2(r, D^-, D^+; \eta, \beta); r_*\right).$$

The dynamics in (2.8) can now be re-expressed in terms of the $O(1)$ quantities in (3.2), thereby obtaining a evolution formula for $r_{i,k}$:

(3.7)
$$\dot{r}_{i,k}(t) = \frac{f_{i,k-1}(t) - f_{i,k}(t)}{\delta}$$

for $i = 1, \ldots, i^{\max}$ and $k = 1, \ldots, k^{\max}$, where

(3.8)
$$f_{i,k}(t) = w\left(r_{i,k}(t), -D^-_{i,k}(t), D^+_{i,k}(t); r_*, \alpha_i, \eta, \beta\right)$$

for $i = 1, \ldots, i^{\max}$, $k = 0, \ldots, k^{\max}$, and $r_{i,0}$ is prescribed for $i = 1, \ldots, i^{\max}$

The next step is to interpret (3.7) as a conservative finite-difference formula for a sufficiently smooth function $\rho = \rho(x, y, t)$, defined on $[0, 1) \times [0, 1] \times [0, \infty)$, such that

(3.9)
$$\rho(x_i, z_k, t) = r_{i,k}(t),$$

on grid points

$$x_i = (i - 0.5)\,\varepsilon \quad \text{and} \quad z_k = k\delta, \tag{3.10}$$

for $i = 1, \ldots, i^{\max}$ and $k = 0, \ldots, k^{\max}$. We also let $\alpha = \alpha(x)$ be a continuous function such that $\alpha(x_i) = \alpha_i$.

Let $\psi = \psi(x, z, t)$ be a smooth test function with compact support on $[0, 1) \times [0, 1] \times [0, \infty)$ and set $\psi_{i,k}(t) = \psi(x_i, z_k, t)$. From (3.7),

$$\delta \sum_{k=1}^{k^{\max}} \psi_{i,k}(t)\dot{r}_{i,k}(t) = \sum_{k=0}^{k^{\max}-1} [\psi_{i,k+1}(t) - \psi_{i,k}(t)] f_{i,k}(t) \tag{3.11}$$

$$+ \psi_{i,0}(t) f_{i,0}(t) - \psi_{i,k^{\max}}(t) f_{i,k^{\max}}(t). \tag{3.12}$$

Let the function $\phi = \phi(x, z, t)$ interpolate the fluxes on the grid:

$$\phi(x_i, z_k, t) = f_{i,k}(t), \tag{3.13}$$

for $i = 1, \ldots, i^{\max}, k = 1, \ldots, k^{\max}, t \geq 0$. Then (3.11) can be interpreted formally as the weak formulation (with respect to z) of a conservation law for ρ with flux ϕ:

$$\int_0^1 \psi(x, \xi, t)\partial_t \rho(x, \xi, t) d\xi = \int_0^1 \partial_z \psi(x, \xi, t)\phi(x, \xi, t) d\xi$$

$$+ \psi(x, 0, t)\phi(x, 0, t) - \psi(x, 1, t)\phi(x, 1, t) + O(\delta). \tag{3.14}$$

To derive a closed model from (3.14), we approximate ϕ in terms of ρ. Such an approximation depends on $D^{\pm}_{i,k}$ via the formula for $f_{i,k}$ in (3.8). From the definition of $Q_{i,k}$ in (2.3) and the scalings in (3.2), it follows that

$$R_{i,k}(t) = \delta \sum_{j=k}^{k^{\max}} r_{i,j}(t) + \int_0^t f_{i,k^{\max}}(s) ds \tag{3.15}$$

and, moreover, that for any finite $t > 0$,

$$\pm D^{\pm}_{i,k}(t) = \frac{\delta}{\varepsilon} \sum_{j=k}^{k^{\max}} [r_{i\pm1,j} - r_{i,j}(t)] + \frac{1}{\varepsilon} \int_0^t [f_{i\pm1,k^{\max}}(s) - f_{i,k^{\max}}(s)]\, ds \tag{3.16}$$

$$= \delta \sum_{j=k}^{k^{\max}} \left[\pm \partial_x \rho(x_i, z_j, t) + \frac{\varepsilon}{2} \partial_x^2 \rho(x_i, z_j, t) + O(\varepsilon^2) \right] \tag{3.17}$$

$$+ \int_0^t \left[\pm \partial_x \phi(x_i, 1, s) + \frac{\varepsilon}{2} \partial_x^2 \phi(x_i, 1, t) + O(\varepsilon^2) \right] ds$$

$$= \pm \partial_x P(x_i, z_k, t) + \frac{\varepsilon}{2} \partial_x^2 P(x_i, z_k, t) + O(\varepsilon^2) + O(\delta) \tag{3.18}$$

where P is given by

$$P(x, z, t) = \int_z^1 \rho(x, \xi, t) d\xi + \int_0^t \phi(x, 1, s) ds \tag{3.19}$$

Motivated by the above calculation, we approximate ϕ by one of two flux functions:

(3.20a) $$\Phi^{(0)}\left(\rho, \partial_x P; r_*, \alpha, \eta, \beta\right) = w\left(\rho, -\partial_x P, \partial_x P; r_*, \alpha, \eta, \beta\right)$$

(3.20b)
$$\Phi^{(1)}\left(\rho, \partial_x P, \partial_x^2 P; r_*, \alpha, \eta, \beta\right) = w\left(\rho, -\partial_x P + \frac{\varepsilon}{2}\partial_x^2 P, \partial_x P + \frac{\varepsilon}{2}\partial_x^2 P; r_*, \alpha, \eta, \beta\right).$$

Using (3.18) and the Lipschitz continuity of w with respect to $D^{\pm}$, we conclude that

(3.21)
$$\begin{aligned} &w(r_{i,k}(t), D^-_{i,k}(t), D^+_{i,k}(t); r_*, \alpha_i, \eta, \beta) \\ &\quad = \Phi^{(1)}(\rho(x_i, z_k, t), \partial_x P(x_i, z_k, t), \partial_x^2 P(x_i, z_k, t); r_*, a, \eta, \beta) + O(\varepsilon^2) + O(\delta). \\ &\quad = \Phi^{(0)}(\rho(x_i, z_k, t), \partial_x P(x_i, z_k, t); r_*, a, \eta, \beta) + O(\varepsilon) + O(\delta). \end{aligned}$$

Thus for $0 \ll \varepsilon, \delta \ll 1$, with $\eta \in (0, \infty)$ fixed, (3.14) is formally consistent with the continuum model

(3.22a) $$\partial_t \rho + \partial_z \Phi^{(\ell)}(\rho, \partial_x P, \partial_x^2 P; r_*, a, \eta, \beta) = 0, \qquad (x, z, t) \in \mathbb{T}^1 \times (0,1) \times (0, \infty),$$

(3.22b) $$\rho(x, 0, t) = \rho_{\text{bc}}(x, t), \qquad (x, t) \in \mathbb{T}^1 \times (0, \infty),$$

(3.22c) $$\rho(x, z, 0) = \rho_0(x, z), \qquad (x, z) \in \mathbb{T}^1 \times (0, 1)$$

where

(3.23a) $$P(x, z, t) = \int_z^1 \rho(x, \xi, t) d\xi + \int_0^t \phi^{(\ell)}(x, 1, s) ds,$$

(3.23b) $$\phi^{(\ell)}(x, z, t) = \Phi^{(\ell)}\left(\rho(x, z, t), \partial_x P(x, z, t), \partial_x^2 P(x, z, t); r_*, \alpha, \eta, \beta\right),$$

and $\Phi^{(\ell)}$, $\ell \in \{0, 1\}$, is given in (3.20). For the sake of compactness, we have slightly abused notation in (3.22a), as the definition of $\Phi^{(0)}$ is independent of $\partial_x^2 P$. Additionally, we have identified $[0, 1)$ with the one-dimensional torus $\mathbb{T}^1$ in order to reflect the periodic layout of the processors.

As in the discrete case, it may appear that the model in (3.22) is circular due to the definition of P in (3.23a). However, as with F in (2.9), $\Phi^{(\ell)}$ can be unwrapped, this time using the conservation law (3.22a); that is

(3.24) $$\int_0^t \phi^{(\ell)}(x, 1, s) ds = \int_0^t \phi(x, 0, s) ds + \int_0^1 \rho_0(x, \xi) d\xi - \int_0^1 \rho(x, \xi, t) d\xi$$

Thus the continuum model is complete once initial condition ρ_0 and inflow condition $\phi_{\text{bc}} := \phi(\cdot, 0, \cdot)$ are specified. In practice, ρ_{bc} is prescribed and then ϕ_{bc} is evaluated using (3.23b) and (3.20).

We use the flux function $\Phi^{(0)}$ for all of the numerical simulations in Section 4. This function is a piecewise constant that can be expressed in the following form:

(3.25)
$$\Phi^{(0)}(r, D; r_*, \alpha, \beta) = \begin{cases} \alpha & (r, D) \in \Omega_1, \\ \dfrac{\alpha r}{r_*} & (r, D) \in \Omega_2, \\ \dfrac{\alpha(r + \eta D)}{\beta r_*} & (r, D) \in \Omega_3, \\ \dfrac{\alpha(r - \eta D)}{\beta r_*} & (r, D) \in \Omega_4, \end{cases}$$

where the subdomains Ω_i are depicted in Figure 3.

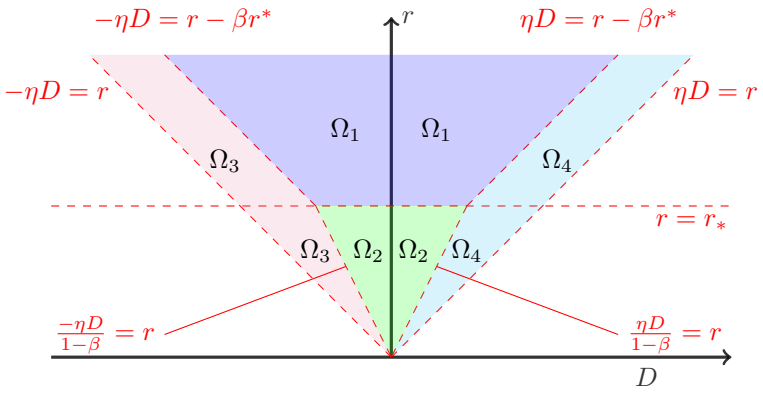


Fig. 3: Flux $\Phi^{(0)}$ defined in (3.25)

3.2. A Hamilton-Jacobi formulation. To our knowledge, there are no immediate conclusions available from the literature regarding the existence or uniqueness of solutions to (3.22). However, we may consider instead a related Hamilton-Jacobi equation which in turn opens up the extensive theory of viscosity solutions. For background, we refer to [6, 11, 20]. We are interested primarily in obtaining, for regular inputs α, $\phi_{\rm bc}$, and ρ_0, the existence and uniqueness of P. It is possible that for more general inputs such results are available in the extensive viscosity solution literature (e.g. [5, 10, 21]).

Integrating (3.22a) with respect to z gives

$$\partial_t \int_z^1 \rho(x,\xi,t)d\xi + \Phi^{(\ell)}(x,1,t) - \Phi^{(\ell)}(x,z,t) = 0 \tag{3.26}$$

Meanwhile, differentiating (3.23a) gives

$$\partial_t P(x,z,t) = \partial_t \int_z^1 \rho(x,\xi,t)d\xi + \Phi^{(\ell)}(x,1,t) \tag{3.27}$$

Combining (3.26) and (3.27) and using the fact that $\rho = -\partial_z P$ gives a closed Hamilton-Jacobi equation for P with initial and boundary conditions that are derived by applying the definition of P in (3.23a) to (3.22c) and (3.22b), respectively. The complete model is, for some $T > 0$,

$$\partial_t P - \Phi^{(\ell)}(-\partial_z P, \partial_x P, \partial_x^2 P; r_*, \alpha, \eta, \beta) = 0, \quad (x,z,t) \in \mathbb{T}^1 \times (0,1) \times (0,T), \tag{3.28a}$$

$$P(x,0,t) - \int_0^1 \rho_0(x,\xi)d\xi - \int_0^t \phi_{\rm bc}(x,s)ds = 0, \qquad (x,t) \in \mathbb{T}^1 \times (0,T), \tag{3.28b}$$

$$P(x,z,0) - \int_z^1 \rho_0(x,\xi)d\xi = 0, \qquad (x,z) \in \mathbb{T}^1 \times (0,1), \tag{3.28c}$$

where (3.28b) is derived by integrating (3.22b) over $z \in (0,1)$ and applying (3.24).

THEOREM 3.1. *Assume that α and ρ_0 are (i) non-negative, (ii) uniformly Lipschitz in their arguments, and (iii) periodic in x (that is, $\alpha(0) = \alpha(1)$ and $\rho_{\rm bc}(0,t) =$*

$\rho_{\text{bc}}(1,t)$ for all $t \in [0,T]$). Further, assume that there is an M where $\int_0^T \phi_{\text{bc}}(x,s)ds \leq M$ for all $x \in \mathbb{T}^1$. Then there exists a unique, continuous, viscosity solution (in the sense of [11]) to (3.28).

Proof. We show that Lemma A.6 applies by first modifying the domain in (3.28). We extend α, ρ_0, ρ_{bc}, and ϕ_{bc} as functions of x from $\mathbb{T}^1$ to all of $\mathbb{R}$ by tiling; for simplicity, in the remainder of the proof we still refer to these extensions by the same name. The assumption that $\alpha(0) = \alpha(1)$ means that the extended version of α is uniformly Lipschitz on $\mathbb{R}$. We then consider (3.28) defined on $\Omega := \mathbb{R} \times (0,1) \times (0,T)$, and to more closely align with the results in the appendix, let

$$H^{(\ell)}(x,z,t,P,\nabla P,\nabla^2 P) = -\Phi^{(\ell)}(-\partial_z P, \partial_x P, \partial_x^2 P; r_*, \alpha(x), \eta, \beta) \tag{3.29}$$

for $\ell \in \{0,1\}$, where $\nabla = (\partial_x, \partial_z)$. By the hypothesis on α, both $H^{(0)}$ and $H^{(1)}$ are uniformly Lipschitz on all of $\Omega \times \mathbb{R} \times \mathbb{R}^2 \times \mathcal{S}^2$, where $\mathcal{S}^n$ is the space of all $n \times n$ symmetric matrices. Moreover, $H^{(\ell)}$ is nonnegative, bounded by α, and independent of the argument P. This means that it immediately satisfies Hypotheses 1 and 3-8 of Lemma A.6. Thus the only condition of Lemma A.6 left to be verified is Hypothesis 2, which is the degenerate ellipticity condition on $H^{(\ell)}$. Verifying this condition can be done in a sequence of simple steps, starting with the definitions of w_1 and w_2.

$$\begin{aligned}
&w_1 \text{ is non-decreasing WRT } r \text{ and } w_2 \text{ is non-decreasing WRT } D^+, D^- && \text{(see (3.5))}\\
&\quad \Longrightarrow w \text{ is non-decreasing WRT } D^+, D^- && \text{(see (3.6))}\\
&\quad \Longrightarrow \Phi^{(\ell)} \text{ is non-decreasing WRT } \partial_x^2 P && \text{(see (3.20))}\\
&\quad \Longrightarrow H^{(\ell)} \text{ is degenerate elliptic} && \text{(see (3.29))}
\end{aligned}$$

Finally, to invoke Theorem A.8, we must establish the existence of subsolutions and supersolutions as defined in (A.2) and (A.3), respectively. This is done by the usual construction found in, for instance, [20, Section 2.3.2.1]. Let

$$P^{\pm}(x,z,t) := \int_z^1 \rho_0(x,\xi)d\xi \pm M \pm t \cdot \sup_x \alpha(x). \tag{3.30}$$

Clearly $\partial_t P^{\pm} = \pm \sup \alpha(x)$ and since $|\Phi_\ell| \leq |\alpha(x)|$, it follows that

$$\pm\left[\partial_t P^{\pm} - \Phi^{(\ell)}(-\partial_z P^{\pm}, \partial_x P^{\pm}, \partial_x^2 P^{\pm}; r_*, \alpha, \eta, \beta)\right] \geq 0. \tag{3.31}$$

Thus $P^{\pm}$ satisfy the interior conditions in (A.3a) and (A.2a), respectively. Next write (3.28b) and (3.28c) in the form $h(t,x,P,\nabla P) = 0$. Then it is straight-forward to verify that $\pm h(t,x,P^{\pm},\nabla P^{\pm}) \geq 0$. Hence $P^{\pm}$ satifies the parabolic boundary conditions in (A.3a) and (A.2a), respectively. Therefore P^- is a subsolution and P^+ is a supersolution for (3.28). This completes the proof. ∎

Remark 3.2. In general, results regarding the regularity of solutions to (3.28) using $\Phi^{(0)}$ (no such results are known by the authors for $\Phi^{(1)}$) require additional smoothness of (and possibly convexification of) $\Phi^{(0)}$ as well as other technical conditions (see [26], [7], [9]). Therefore obtaining the existence of a L^1 function ρ solving (3.22) (in some generalized sense) via the existence of P solving (3.28) is still an open problem.

3.3. Higher Dimensional Models. Both the discrete and continuum models above can be readily extended to systems of processors arranged in an n-dimensional

periodic lattice. Assuming that processors only communicate with their nearest neighbors (i.e., no diagonal communication), the n-dimensional analog of the system formed by (2.1), (2.3), and (2.8) is:

$$\frac{dq_{\mathbf{i},k}}{dt} = F_{\mathbf{i},k-1} - F_{\mathbf{i},k}, \qquad Q_{\mathbf{i},k}(t) = \left(\sum_{j=k}^{k^{\max}} q_{\mathbf{i},j}(t)\right) + \int_0^t F_{\mathbf{i},k^{\max}}(s)ds. \tag{3.32}$$

$$F_{\mathbf{i},k} = a_{\mathbf{i}}\, v_1\left(\min_{1\le d\le n}\left\{v_2\left(q_{\mathbf{i},k}, Q_{\mathbf{i}-\mathbf{e}_d,k} - Q_{\mathbf{i},k} + q_{\mathbf{i},k}, Q_{\mathbf{i}+\mathbf{e}_d,k} - Q_{\mathbf{i},k} + q_{\mathbf{i},k};\beta\right)\right\}; q_*\right) \tag{3.33}$$

where $\mathbf{i} = (i_1, \ldots, i_n)$ is a multi-index and $(\mathbf{e}_d)_i = \delta_{d,i}$. As in the one-dimensional case, v_2 provides the amount of available data to process, after accounting for the throttling from neighbors over a given axis. The multidimensional discrete model then takes the minimum over all possible axes in order to determines what is available to be processed to the next stage. As in the one-dimensional case, self-throttling is computing use v_1 based on the amount of data available for processing.

If $i_d^{\max}$ denotes the number of processors along the d direction, we let $\varepsilon_d = (i_d^{\max})^{-1}$; the definition of δ is unchanged. For notational convenience, we set $V := \prod_{d=1}^n \varepsilon_d$. We define $w_{2,d}$ by replacing η in the definition of w_2 with $\eta_d := \varepsilon_d/\delta$. Then we define the quantities analogous to those in (3.2)

$$r_* := \frac{q_*}{\delta V}, \quad \hat{\alpha}_{\mathbf{i}} := \frac{\alpha_{\mathbf{i}}}{V}, \quad r_{\mathbf{i},k} := \frac{q_{\mathbf{i},k}}{\delta V}, \quad R_{\mathbf{i},k} := \frac{1}{V}Q_{\mathbf{i},k}, \quad D^{d,\pm}_{\mathbf{i},k} := \pm\frac{R_{\mathbf{i}\pm\mathbf{e}_d,k} - R_{i,k}}{\varepsilon_d}. \tag{3.34}$$

Continuing as in Section 3.1, we define

$$x_{\mathbf{i}} = \Big(\varepsilon_1(i_1+0.5), \ldots, \varepsilon_n(i_n+0.5)\Big), \qquad z_k = k\delta \tag{3.35}$$

and the smooth density function $\rho(x,z,t)$ defined on $[0,1)^n \times (0,1) \times [0,\infty)$ for which $\rho(x_{\mathbf{i}}, z_k, t) = r_{\mathbf{i},k}(t)$. Arguments analogous to those used to obtain (3.16) and (3.18) give us that

$$\pm\, D^{d,\pm}_{\mathbf{i},k} \approx \pm\partial_{x_d} P(x_{\mathbf{i}}, z_k, t) + \frac{\varepsilon_d}{2}\partial^2_{x_d} P(x_{\mathbf{i}}, z_k, t). \tag{3.36}$$

We assume that all of the $i_d^{\max}$ are of the same order, so that the order of accuracy of the above approximation is consistent across all dimensions.

As advection in the z-direction is unchanged, we have the continuum model

$$\partial_t \rho + \partial_z \Phi^{(\ell)}(\rho, \nabla_x P, \nabla_x^2 P; r_*, \alpha, \vec{\eta}, \beta) = 0, \quad (x,z,t) \in \mathbb{T}^n \times (0,1) \times (0,\infty), \tag{3.37a}$$

$$\rho(x,0,t) = \rho_{\mathrm{bc}}(x,t), \quad (x,t) \in \mathbb{T}^n \times (0,\infty), \tag{3.37b}$$

$$\rho(x,z,0) = \rho_0(x,z), \quad (x,z) \in \mathbb{T}^n \times (0,1), \tag{3.37c}$$

where $\mathbb{T}^n$ denotes the n-dimensional torus parameterized by $[0,1)^n$, P and $\phi^{(\ell)}$ are defined as in (3.23a) and (3.23b), respectively, and the form of $\Phi^{(\ell)}$, $\ell \in \{0,1\}$ is a

slightly generalized version of (3.20).

(3.38a)
$$\Phi^{(0)}(\rho, \nabla_x P, \nabla_x^2 P; r_*, \alpha, \vec{\eta}, \beta) = \alpha\, w_1\Big(\min_{1\le d\le n} \Big(\big[w_2(\rho, \partial_{x_d} P, -\partial_{x_d} P; \eta_d, \beta) \big] \Big); r_* \Big)$$

(3.38b)
$$\Phi^{(1)}(\rho, \nabla_x P, \nabla_x^2 P; r_*, \alpha, \vec{\eta}, \beta) =$$

(3.38c)
$$\alpha\, w_1\Big(\min_{1\le d\le n} \Big(\big[w_2(\rho, \partial_{x_d} P + \frac{\varepsilon_d}{2}\partial_{x_d}^2 P, -\partial_{x_d} P + \frac{\varepsilon_d}{2}\partial_{x_d}^2 P; \eta_d, \beta) \big] \Big); r_* \Big)$$

For notational convenience we have defined $\Phi^{(\ell)}$ using the full tensor $\nabla_x^2 P$; however, we note that the flux functions do not depend on mixed second derivatives. The procedure used in Section 3.2 to obtain a Hamilton-Jacobi equation for P can be repeated here; the only changes are (i) the multi-dimensional version of $\Phi^{(\ell)}$ in (3.20) and (ii) the domain of the x variable. Verifying that these newly-defined flux functions satisfy the conditions of Theorem A.6 is essentially the same as before. Existence and uniqueness of viscosity solutions P then follow.

4. Numerical Simulations. In this section, we perform numerical simulations of the one dimensional processor system in order to (i) test the ability of the macroscopic model to approximate the discrete model when ε and δ are small and (ii) explore how model parameters affect the model output. All simulations are based on the flux $\Phi^{(0)}$, although results with $\Phi^{(1)}$ demonstrate similar. Problem data is specified in terms continuum model of continuum models quantities. These quantities are translated back to discrete model quantities in order to implement ODE simulations.

4.1. ODE Implementation. The explicit two-step Adams-Bashforth (Section III of [18]) is used to simulate the discrete model formed by (2.1), (2.3), and (2.8). Given η, values $i^{\max}$ and $k^{\max}$ are chosen so that $k^{\max}/i^{\max} = \eta$ (cf. (3.1)). We then compute a solution to the discrete model as follows. Using (3.2) and (3.9), we convert $r_*, a, \rho_0, \rho_{bc}$ to their discrete counterparts:

$$q_* = \varepsilon\delta r_*, \qquad q_{i,k}(0) = \varepsilon\delta\rho_0(x_i, z_k), \qquad q_{i,0}(t) = \rho_{\mathrm{bc}}(x,t), \quad a_i = \varepsilon\alpha(x_i). \tag{4.1}$$

This discrete model data is used to set the time step:

$$\Delta t = \frac{q_*}{2\big(\max_i a_i\big)\sqrt{i^{\max}k^{\max}}}. \tag{4.2}$$

The outflow at $F_{i,k^{\max}}$ is tracked and accumulated over time in order to compute $Q_{i,k}$ from (2.3). At the final time T, the result of the explicit time stepping is converted back, via the formula in (3.2), i.e., $r_{i,k}(T) = (\varepsilon\delta)^{-1} q_{i,k}(T)$. In order to compare this against solutions to the continuum model (see below), we use these point-wise values to generate a piecewise constant function r over the cells $C_{i,k} = (x_i - .5\varepsilon, x_i + .5\varepsilon) \times (z_k, z_k + \delta)$:

$$r(x,z) = \sum_{i,k} r_{i,k}\chi_{C_{i,k}}(x,z). \tag{4.3}$$

4.2. Hamilton-Jacobi Implementation. The Hamilton Jacobi equation (3.28) is solved numerically using a fifth-order WENO interpolation in x and z and the optimal third-order SSP Runge-Kutta method for time integration. Details of these

algorithms can be found in Sections 3.2 and 6, respectively, of [27]. Once a numerical solution for P is computed, we again use WENO interpolation to approximate ρ via the relation $\rho(x,z,t) = -\partial_z P(x,z,t)$.

To condense the notation, let $\sigma = \partial_x P$, $\tau = \partial_z P$ and $\upsilon = \partial_{xx} P$. Then for fixed r_*, α, η, β, and ℓ, let $H(\sigma,\tau,\upsilon) = -\Phi^{(\ell)}(-\tau,\sigma,\upsilon;r_*,a,\eta,\beta)$. The numerical solution for P is computed on a grid $\{x_n, z_m\}$ where

$$x_n = n\Delta x, \qquad n = 1,2,\dots,N, \qquad \Delta x = N^{-1}, \tag{4.4}$$

$$z_m = m\Delta z, \qquad m = 1,2,\dots,M, \qquad \Delta z = M^{-1}. \tag{4.5}$$

The semi-discrete method for the grid function $P_{n,m}(t) \approx P(x_n, z_m, t)$ is

$$\frac{d}{dt}P_{n,m}(t) = -\hat{H}(\sigma^-_{n,m}, \sigma^+_{n,m}, \tau^-_{n,m}, \tau^+_{n,m}; \upsilon_{n,m}), \tag{4.6}$$

where the numerical approximations $\sigma^\pm_{n,m} \approx \sigma(x^\pm_n, z_m)$ and $\tau^\pm_{n,m} \approx \tau(x_n, z^\pm_m)$ are obtained via WENO interpolation and $\upsilon_{n,m} \approx \upsilon(x_n, z_m)$ is computed by central difference. The numerical flux function $\hat{H}$, based on the global Lax-Friedrichs flux:

$$\hat{H}(\sigma^-,\sigma^+,\tau^-,\tau^+;\upsilon) = H\left(\frac{\sigma^- + \sigma^+}{2}, \frac{\tau^- + \tau^+}{2}, \upsilon\right) - \frac{1}{2}\lambda^x(\sigma^+ - \sigma^-) - \frac{1}{2}\lambda^z(\tau^+ - \tau^-), \tag{4.7}$$

where

$$\lambda^x = \max_{\sigma,\tau}|H_\sigma| = \frac{\alpha\eta}{\beta r_*}, \quad \lambda^z = \max_{\sigma,\tau}|H_\tau| = \frac{\alpha}{\beta r_*}. \tag{4.8}$$

The time step for the SSP integrator is given by

$$\Delta t\left(\frac{\lambda^x}{\Delta_x} + \frac{\lambda^z}{\Delta z}\right) \le 0.6. \tag{4.9}$$

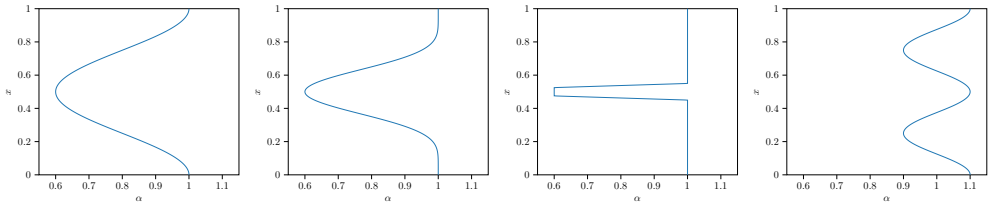
4.3. Experiments. We perform a sequence of exploratory experiments below, modifying the parameters η and β, as well as the throughput function α. In all cases, α, ρ_0, and $\rho_{\rm bc}$ are periodic with respect to x and the parameter $r_* = 1$. Results are presented as two-dimensional color maps or line-outs in the z direction. In all figures, the horizontal axis corresponds to the z-axis. Profiles of α for each experiment are depicted in Figure 4.

EXAMPLE 1 (Agreement between models). *The purpose of this example is to demonstrate that the macroscopic model approximates the microscopic model when ε and δ are sufficiently small. We set $\beta = 1$ and consider $\eta \in \{0.2, 1, 5\}$. The initial condition, boundary condition, and processor speed are given by*

$$\rho_0(x,z) = 1.5\left(\sin(2\pi z)\right)^6 \chi_{[0,0.5]}(z), \quad \rho_{\rm bc}(x,t) = 0, \quad \alpha(x) = 1 - 0.4(\sin(\pi x))^2, \tag{4.10}$$

respectively. Both models are simulated up to a final time $t = 0.5$.

For this example, the Hamilton-Jacobi simulation is performed with a 1000×1000 mesh and a time step chosen according to (4.9) in order to generate a highly resolved numerical solution of the macroscopic model. For the microscopic model, we use $i^{\max} = 1000$ and $k^{\max} = 200$ when $\eta = 0.2$, $i^{\max} = k^{\max} = 500$ when $\eta = 1$, and $i^{\max} = 200$ and $k^{\max} = 1000$ when $\eta = 5$. These solutions to the microscopic model are then used to obtain the piecewise-constant function r on the 1000×1000 mesh from the Hamilton-Jacobi simulation.



(a) Example 1 (b) Examples 2-3 (c) Example 4 (d) Example 5

Fig. 4: Profiles of the processor speed α used in the numerical experiments. The non-standard orientation of the graphs is set to match the axes in the numerical results that follow.

Numerical results for $\eta = 0.2$, $\eta = 1.0$, and $\eta = 5.0$ are shown in Figure 5, Figure 6, and Figure 7, respectively. While the results demonstrate general qualitative agreement between the models, discrepancies develop over time, especially for smaller values of η; see Figures 5i and 5l. For the worst case scenario ($\eta = 0.2$), we increase the size of the discrete model by a factor of 2.5 (giving $i^{\max} = 2500$ and $k^{\max} = 500$), at which point the discrepancy between models decreases noticeably; see Figure 8.

For the remaining examples, the Hamilton-Jacobi simulations are performed on a coarser mesh of 100×100.

EXAMPLE 2 (Variations in η). *In this example, we examine the effect of η on solutions to the macroscopic model while $\beta = 1.0$ is fixed. The initial condition, boundary condition, and processor speed are given by*

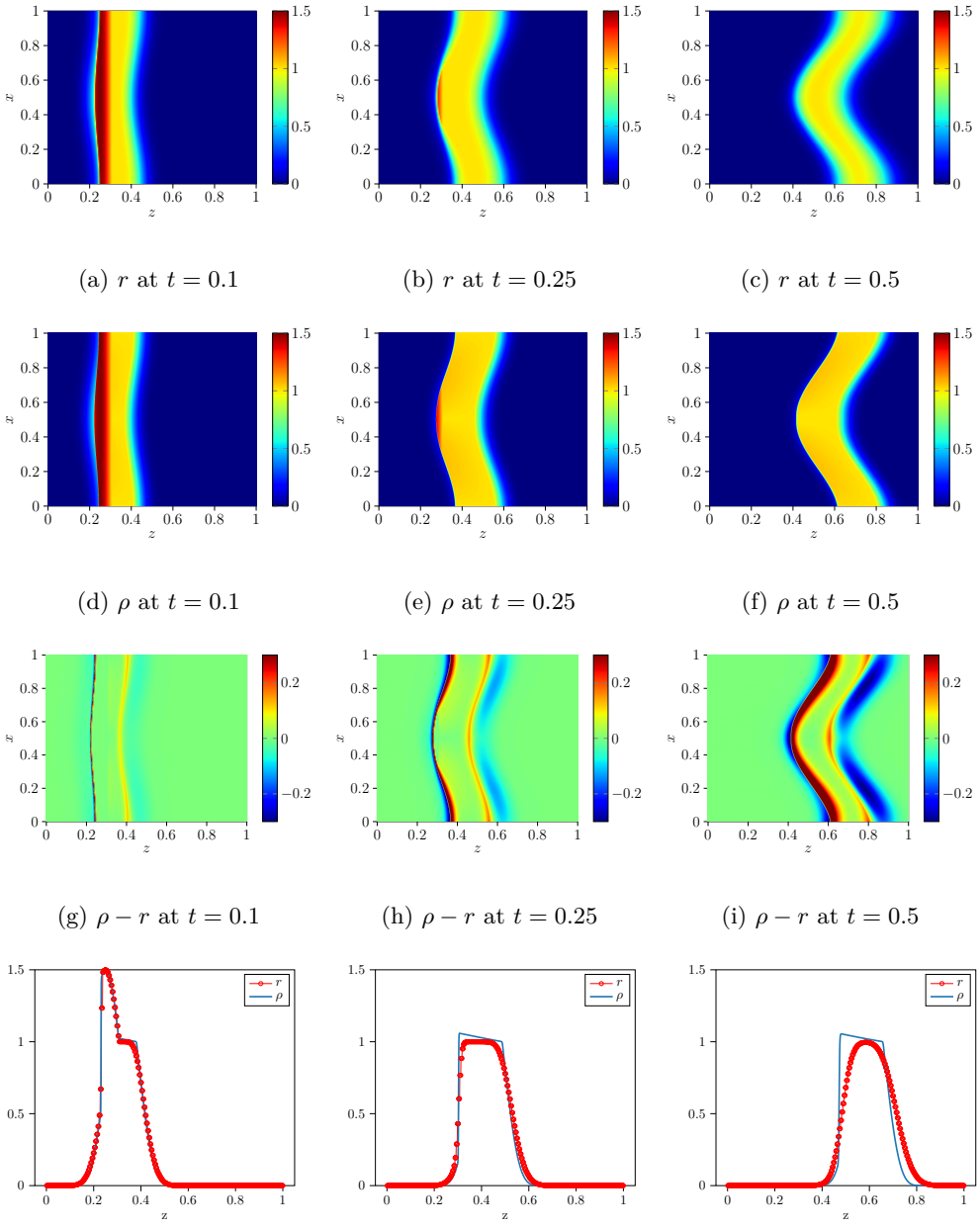
$$\rho_0(x,z) = 1.5\chi_{z\leq 0.2}(x,z) \quad \rho_{\rm bc}(x,t) = 0, \quad \alpha(x) = 1 - 0.4(\sin(\pi x))^6, \tag{4.11}$$

respectively. It is expected that the slower processor speed around $x = 0.5$ will slow down neighboring processors due to neighbor-based throttling, encoded in the definition of w_2 in (3.5b). Moreover, the effect should become more global in x as η increases, since larger values of η correspond to a larger number of stages per processor. Indeed as the stages increase, interactions between neighbors begin to have a cumulative global effect. This trend can be observed by comparing results across the first three rows of Figure 9 and in the line-outs in the final row.

EXAMPLE 3 (Variations in β). *In this example, we examine the effect of β on solutions to the macroscopic model, while holding $\eta = 1.0$ fixed. The initial condition, boundary condition, and processor speed are again given by (4.11).*

Based on the definition of the function w_2 in (3.5b), the expectation is that smaller values of β will lead to reduced throttling effects. Such behavior is confirmed by the numerical results in Figure 10.

EXAMPLE 4 (Highly localized slowdown). *In this example, we explore the effects of a highly localized slowdown in processor speed when $\eta = \beta = 1$. The initial and boundary conditions are given in (4.11), while the processor speed is given by*



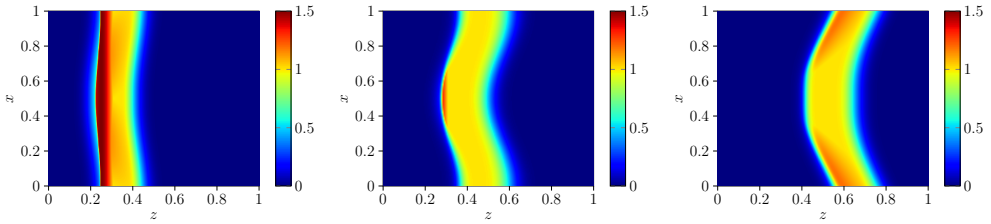
(a) r at $t = 0.1$ (b) r at $t = 0.25$ (c) r at $t = 0.5$

(d) ρ at $t = 0.1$ (e) ρ at $t = 0.25$ (f) ρ at $t = 0.5$

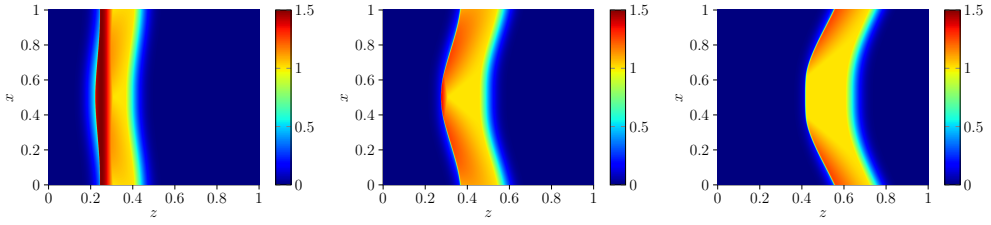
(g) $\rho - r$ at $t = 0.1$ (h) $\rho - r$ at $t = 0.25$ (i) $\rho - r$ at $t = 0.5$

(j) ρ,r at $(x, t) = (0.3, 0.1)$ (k) ρ,r at $(x, t) = (0.3, 0.25)$ (l) ρ,r at $(x, t) = (0.3, 0.5)$

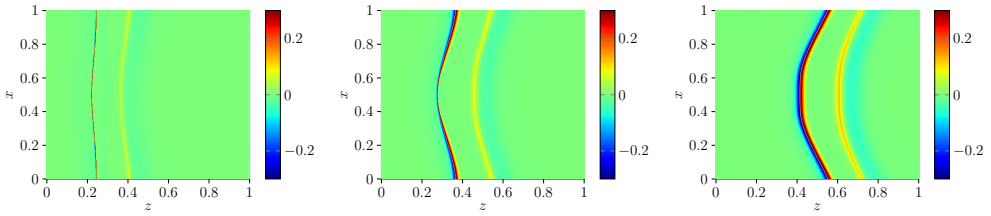
Fig. 5: Discrete solution r and continuum solution ρ when $\eta = 0.2$. From left to right, columns correspond to solutions at $t = 0.1$, $t = 0.25$, and $t = 0.5$ Discrete solution is computed with $(i^{\max}, k^{\max}) = (1000, 200)$. Continuum solution is computed on a $10^3 \times 10^3$ mesh.



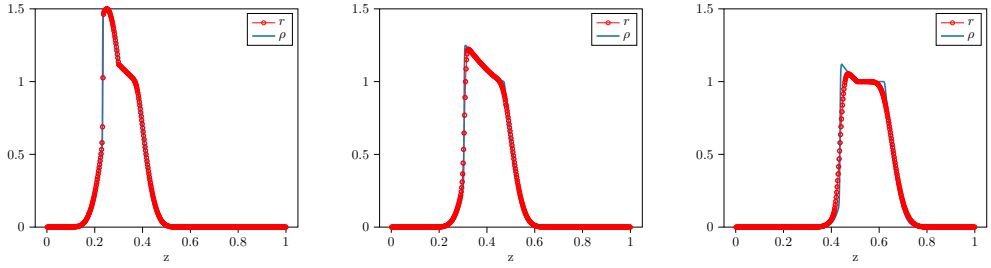
(a) r at $t = 0.1$ (b) r at $t = 0.25$ (c) r at $t = 0.5$



(d) ρ at $t = 0.1$ (e) ρ at $t = 0.25$ (f) ρ at $t = 0.5$

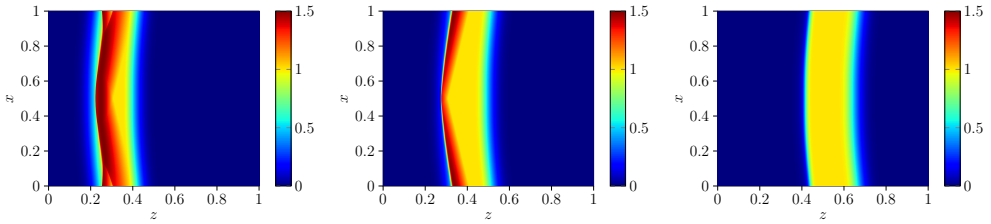


(g) $\rho - r$ at $t = 0.1$ (h) $\rho - r$ at $t = 0.25$ (i) $\rho - r$ at $t = 0.5$

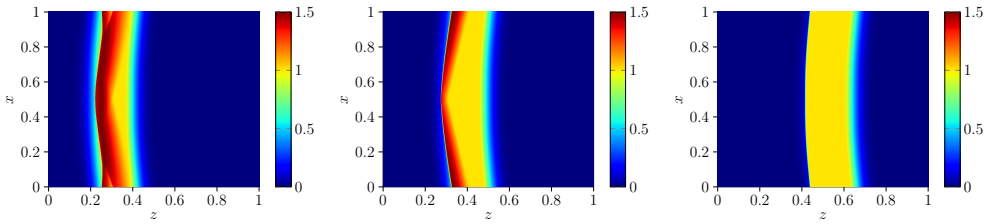


(j) ρ,r at $(x, t) = (0.3, 0.1)$ (k) ρ,r at $(x, t) = (0.3, 0.25)$ (l) ρ,r at $(x, t) = (0.3, 0.5)$

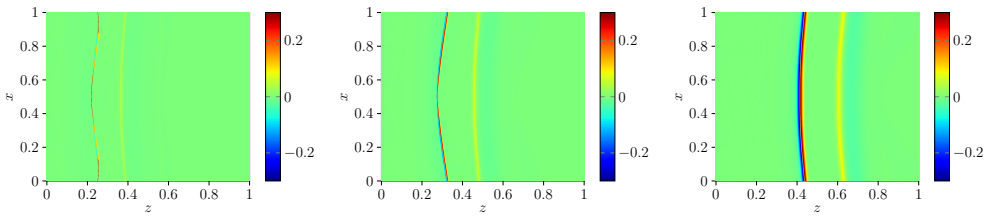
Fig. 6: Discrete r and continuum ρ solutions when $\eta = 1$ case. From left to right, column correspond to solutions at $t = 0.1$, $t = 0.25$, and $t = 0.5$ Discrete solution is computed with $(i^{\max}, k^{\max}) = (500, 500)$. Continuum solution is computed on a $10^3 \times 10^3$ mesh.



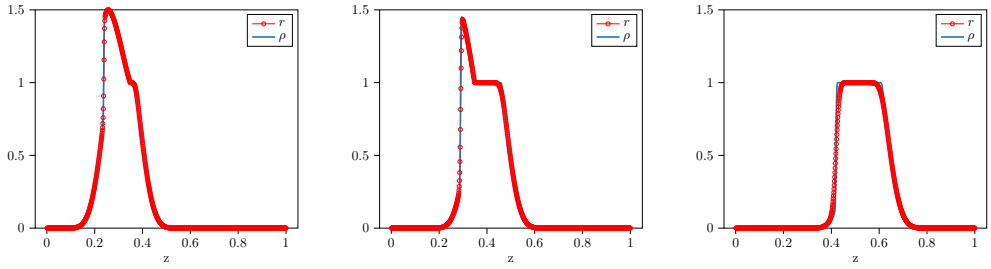
(a) r at $t = 0.1$ (b) r at $t = 0.25$ (c) r at $t = 0.5$



(d) ρ at $t = 0.1$ (e) ρ at $t = 0.25$ (f) ρ at $t = 0.5$

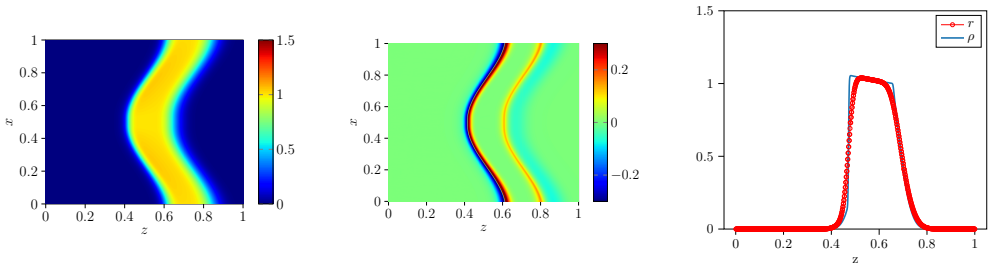


(g) $\rho - r$ at $t = 0.1$ (h) $\rho - r$ at $t = 0.25$ (i) $(\rho - r)$ at $t = 0.50$



(j) ρ,r at $(x, t) = (0.3, 0.1)$ (k) ρ,r at $(x, t) = (0.3, 0.25)$ (l) ρ,r at $(x, t) = (0.3, 0.5)$

Fig. 7: Discrete r and continuum ρ solutions when $\eta = 5$ case. From left to right, column correspond to solutions at $t = 0.1$, $t = 0.25$, and $t = 0.5$ Discrete solution is computed with $(i^{\max}, k^{\max}) = (200, 1000)$. Continuum solution is computed on a $10^3 \times 10^3$ mesh.



(a) r at $t = 0.5$ (b) $\rho - r$ at $t = 0.5$ (c) ρ,r at $(x, t) = (0.3, 0.5)$

Fig. 8: Comparison of the discrete model with $(i^{\max}, k^{\max}) = (2500, 500)$ and the continuum model for $\eta = 0.2$ at time $t = 0.5$. As expected, the discrete model shows better agreement with the continuum model than the previous version with only $(i^{\max}, k^{\max}) = (1000, 200)$ processors and stages; cf. Figure 5

$\alpha(x) = 1 - 0.4c(x)$, *where*

$$c(x) = \begin{cases} 0 & |x - .5| > .05 \\ 40x - 18 & x \in [.45, .475] \\ -40x + 22 & x \in [.525, .55] \\ 1 & |x - .5| < .025 \end{cases}. \tag{4.12}$$

In particular, $\alpha \neq 1$ *only on the interval* $(0.45, 0.55)$. *Simulation results from this example are shown in Figure 11. At early times, slower processors in the center of the* x *domain prohibit neighboring processors from moving data to later stages of the calculation (i.e. along the* z*-direction). The result is a buildup of data in the neighboring processors. As time progresses, the build-up of data spreads as throttled processors near the initial slowdown around* $x = 0.5$ *begin to effect neighbors further away. Eventually these buildups dissipate as the slower processors begin catch up with their throttled neighbors.*

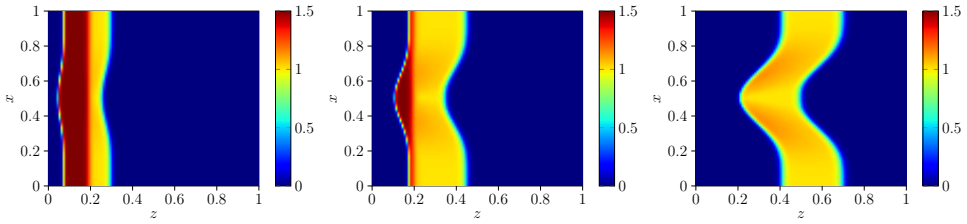
EXAMPLE 5 (Long-term behavior). *In previous examples, we have observed that under some conditions, solutions eventually resemble a traveling profile of the form*

$$\rho_*(x, z, t) = \chi_{[\zeta_0(x), \zeta_1(x)]}(z - st), \tag{4.13}$$

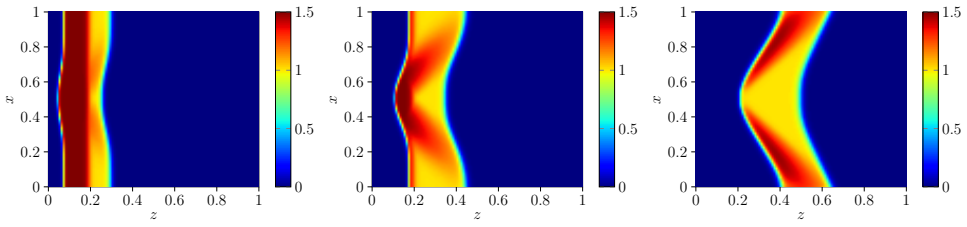
where s *is a positive constant and the profiles* ξ_0 *and* ξ_1 *are constant in time and satisfy* $\zeta_1(x) < \zeta_1(x)$ *for all* $x \in [0, 1)$. *Our intuition is that for a wide range of conditions, traveling profiles are of this type will arise after sufficiently long times, if the* z *domain is extended to* $(0, \infty)$. *Moreover the shape of* ζ_1 *and* ζ_2 *is closely related to the initial data and the shape of* α. [1] *Rather than make a precise conjecture at this point, we instead provide an example which further demonstrates our intuition. Initial and boundary conditions are given in (4.11). Because the domain in* z *is limited, we introduce relatively small variations in* α*, which allow the system to settle faster:*

$$\alpha(x) = 1 + 0.1\cos(4\pi x). \tag{4.14}$$

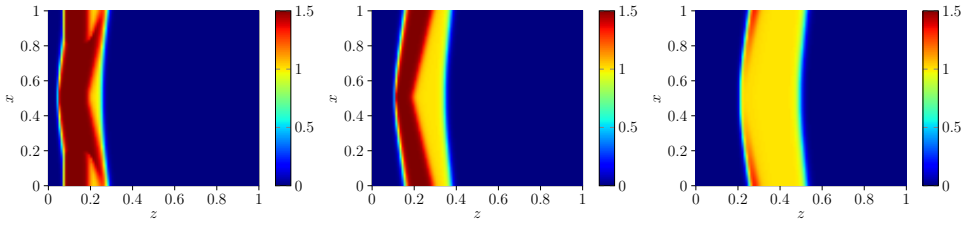
[1] A more systematic study of such profiles in special case can be found in [19].



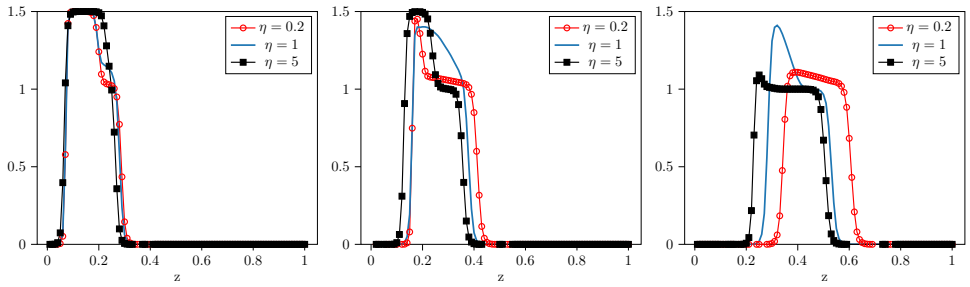
(a) $\rho(x, z, 0.1)$ when $\eta = 0.2$ (b) $\rho(x, z, 0.25)$ when $\eta = 0.2$ (c) $\rho(x, z, 0.5)$ when $\eta = 0.2$



(d) $\rho(x, z, 0.1)$ when $\eta = 1.0$ (e) $\rho(x, z, 0.25)$ when $\eta = 1.0$ (f) $\rho(x, z, 0.5)$ when $\eta = 1.0$

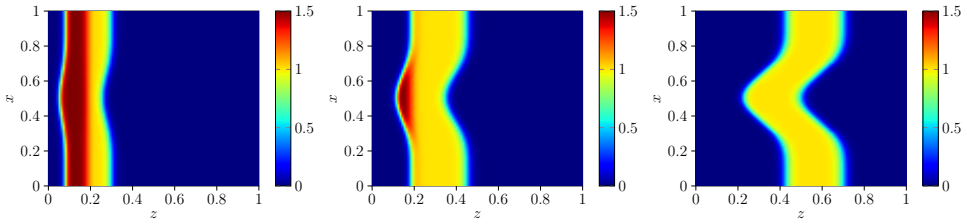


(g) $\rho(x, z, 0.1)$ when $\eta = 5.0$ (h) $\rho(x, z, 0.25)$ when $\eta = 5.0$ (i) $\rho(x, z, 0.5)$ when $\eta = 5.0$

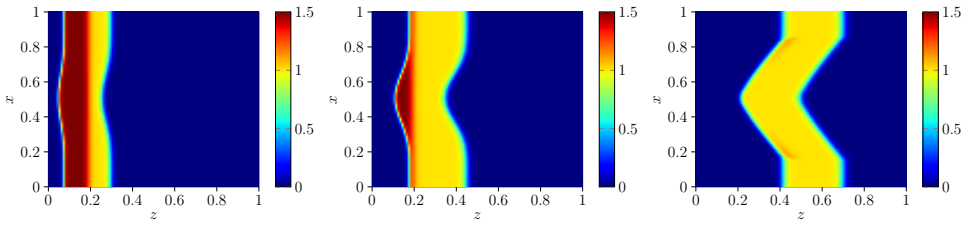


(j) $\rho(0.3, z, 0.1)$ (k) $\rho(0.3, z, 0.25)$ (l) $\rho(0.3, z, 0.5)$

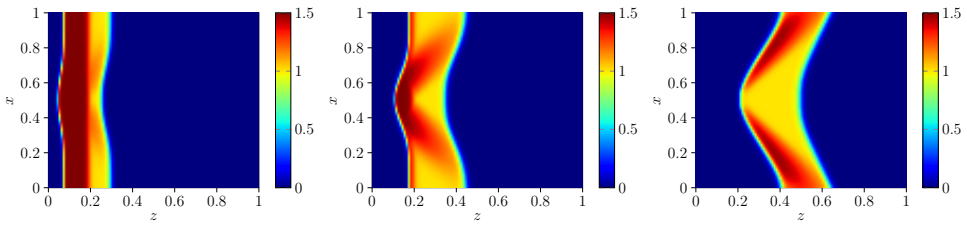
Fig. 9: The effects on ρ due to variations in η. As η increases the throttling effect of a local slowdown spreads more quickly, and data is not processed as quickly.



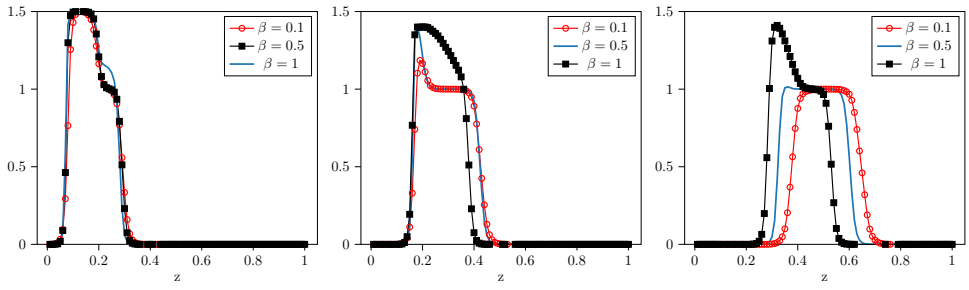
(a) $\rho(x, z, 0.1)$ with $\beta = 0.1$ (b) $\rho(x, z, 0.25)$ with $\beta = 0.1$ (c) $\rho(x, z, 0.5)$ with $\beta = 0.1$



(d) $\rho(x, z, 0.1)$ with $\beta = 0.5$ (e) $\rho(x, z, 0.25)$ with $\beta = 0.5$ (f) $\rho(x, z, 0.5)$ with $\beta = 0.5$

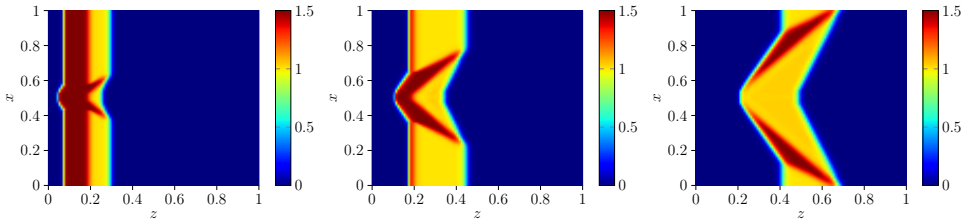


(g) $\rho(x, z, 0.1)$ with $\beta = 1$ (h) $\rho(x, z, 0.25)$ with $\beta = 1$ (i) $\rho(x, z, 0.5)$ with $\beta = 1$



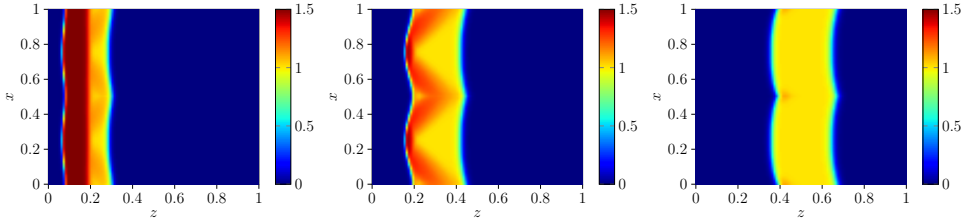
(j) $\rho(0.3, z, 0.1)$ (k) $\rho(0.3, z, 0.25)$ (l) $\rho(0.3, z, 0.5)$

Fig. 10: Plots of the solution ρ from Example 3 for different values of β. Larger values of β lead to more throttling.



(a) $\rho(x, z, 0.1)$ (b) $\rho(x, z, 0.25)$ (c) $\rho(x, z, 0.5)$

Fig. 11: The effect of a highly localized slowdown on ρ



(a) $\rho(x, z, 0.1)$ (b) $\rho(x, z, 0.25)$ (c) $\rho(x, z, 0.5)$

Fig. 12: The effect of small variation in processor speed on ρ. After sufficiently time, a profile emerges with the periodicity of α.

Simulation results for this example are presented in Figure 12. When $t = 0.5$, the solution has nearly settled to a profile of the form (4.13), with cusps that appear where the waves caused by throttling meet, at $x = 0.5$ and at the periodic boundary. In particular the solution has the periodicity of α.

5. Conclusion. We have presented a simple discrete model of a network of processors in a high performance computing environment where the computational throughput depends on the on the availability of data from neighboring processors. This discrete, microscopic-level model has been then used to derive a continuum-level model which treats computational progress as an Eulerian fluid flow. Currently, the existence and uniqueness of solutions to the partial differential equation in this fluid model is open. However, a Hamilton-Jacobi model is available for which we can establish the existence and uniqueness of continuous viscosity solutions; the solution for the governing equation corresponds to the total amount of data that has been processed through a particular stage in the computation. Numerical experiments have shown that this continuum model can capture the asymptotic behavior of the discrete model. Additionally, we have used these experiments to give an initial understanding of solutions' dependence on parameters associated with the parallelism of the modeled computation as well as the effects heterogeneities in processing capacity.

In future work, we intend to explore control strategies for α that can alleviate bottlenecks caused by local slowdowns in the processor speed. We will also extend

the model to allow for more complicated interactions, including stochastic effects, and explore strategies for optimal communication. Finally, we hope to tune the parameters of the model with data taken from processor components of a real supercomputer and then compare predictions of the macroscopic model with the real global behavior of the supercomputer.

Appendix A. Hamilton-Jacobi Theory.

We recall a few standard definitions from the theory of nonlinear second-order Hamilton-Jacobi equations as in, for instance, [6, 11]:

DEFINITION A.1 (Degenerately elliptic function). *Let $F : \mathbb{R}^n \times \mathbb{R}^n \times \mathcal{S}^n \to \mathbb{R}$ be given, where $\mathcal{S}^n$ is the set of symmetric $n \times n$ matrices. Then we say F is degenerately elliptic if $F(x, r, p, X) \leq F(x, r, p, Y)$ whenever $Y \leq X$.*

DEFINITION A.2 (Modulus function). *We call a function $\sigma : [0, \infty) \to [0, \infty)$ a modulus function if $\sigma(0) = 0$ and it is nondecreasing.*

In contrast to the notational convention in Section 3, we follow in this appendix the convention of the viscosity literature and refer to time- and space-dependent functions as $u(t, x)$.

DEFINITION A.3 (Parabolic boundary). *If $U = (0, T] \times D$ where $D \subset \mathbb{R}^n$ and $T \geq 0$, then $\partial_P U := \{0\} \times D \cup [0, T] \times \partial D$ is called the parabolic boundary of U.*

DEFINITION A.4 (Semicontinuous envelope). *The upper (respectively, lower) semicontinuous envelopes of a function $u : V \to [-\infty, \infty]$ are*

$$u^*(x) = \lim_{r \downarrow 0} \sup\{u(y) : y \in V, |y - x| \leq r\},$$

$$u_*(x) = \lim_{r \downarrow 0} \inf\{u(y) : y \in V, |y - x| \leq r\}.$$

They are, respectively, the smallest upper semicontinuous function greater than u and the largest lower semicontinuous function less than u.

DEFINITION A.5 (Viscosity solutions). *Let $f : U \times \mathbb{R} \times \mathbb{R}^n \times \mathcal{S}^n \to \mathbb{R}$ be given. An upper (resp. lower) semicontinuous function u is a viscosity subsolution (resp. supersolution) of*

$$u_t + f(t, x, u, \nabla_x u, \nabla_x^2 u) = 0, \qquad (t, x) \in U, \tag{A.1a}$$

$$h(t, x, u, \nabla_x u) = 0, \qquad (t, x) \in \partial_P U, \tag{A.1b}$$

on $(0, T] \times D$ if at every $(t, x) \in (0, T] \times D$, when $u - \psi$ is locally maximized (resp. minimized) at (t, x) and ψ is $C^2((0, T] \times D)$ we have

$$\psi_t(t, x) + f(t, x, u, \nabla_x \psi(t, x), \nabla_x^2 \psi(t, x)) \leq 0, \quad (t, x) \in U \tag{A.2a}$$

$$\min\{\psi_t(t, x) + f(t, x, u, \nabla_x \psi(t, x), \nabla_x^2 \psi(t, x)), \\ h(t, x, u(t, x), \nabla_x \psi(t, x))\} \leq 0, \quad (t, x) \in \partial_P U \tag{A.2b}$$

(respectively,

$$\psi_t(t, x) + f(t, x, u, \nabla_x \psi(t, x), \nabla_x^2 \psi(t, x)) \geq 0, \quad (t, x) \in U \tag{A.3a}$$

$$\max\{\psi_t(t, x) + f(t, x, u, \nabla_x \psi(t, x), \nabla_x^2 \psi(t, x)), \\ h(t, x, u(t, x), \nabla_x \psi(t, x))\} \geq 0, \quad (t, x) \in \partial_P U. \tag{A.3b}$$

A function u is a viscosity solution of (A.1) if its upper semicontinuous envelope is a viscosity subsolution and its lower semicontinuous envelope is a viscosity supersolution.

Next, we recall the following general comparison theorem, which is Theorem 4.1 of [17].

LEMMA A.6 (Generalized comparison principle). *Consider the system (A.1) on $U = (0,T) \times \Omega$ where $\Omega \subseteq \mathbb{R}^n$ is a possibly unbounded domain and $T > 0$. Assume that f satisfies the following assumptions:*

1. *f is continuous on $U \times \mathbb{R} \times (\mathbb{R}^n \setminus \{0\}) \times \mathcal{S}^n$.*
2. *f is degenerately elliptic.*
3. *$-\infty < f_*(t,x,r,0,O) = f^*(t,x,r,0,O) < \infty$ for all $(t,x,r) \in U \times \mathbb{R}$, where O is the zero matrix.*
4. *For every $R > 0$, we have*

(A.4)
$$\sup\{|f(t,x,r,p,X)| : |p|, |X| \leq R, (t,x,r,p,S) \in U \times \mathbb{R} \times (\mathbb{R}^n \setminus \{0\}) \times \mathcal{S}^n\}$$

is finite.

5. *For every $H > 0$, there is a constant c_0 such that $r \mapsto f(t,x,r,p,X) + c_0 r$ is nondecreasing for all $(t,x,r,p,X) \in U \times \mathbb{R} \times (\mathbb{R}^n \setminus \{0\}) \times \mathcal{S}^n$ with $|r| \leq H$.*
6. *For every $R > \rho > 0$ there is a modulus function σ_{R_ρ} such that*

$$|f(t,x,r,p,X) - f(t,x,r,q,Y)| \leq \sigma_{R_\rho}(|p-q| + |X-Y|) \tag{A.5}$$

for $(t,x,r) \in U \times \mathbb{R}$, $\rho \leq |p|$, $|q| \leq R$, and $|X|, |Y| \leq R$.

7. *There is a constant $\rho_0 > 0$ and a modulus function σ_1 such that*

$$f^*(t,x,r,p,X) - f^*(t,x,r,0,O) \leq \sigma_1(|p| + |X|) \tag{A.6}$$
$$f_*(t,x,r,p,X) - f_*(t,x,r,0,O) \geq -\sigma_1(|p| + |X|) \tag{A.7}$$

for $(t,x,r) \in U \times \mathbb{R}$ and $|p|, |X| \leq \rho_0$.

8. *There is a modulus function σ_2 such that*

$$|f(t,x,r,p,X) - f(t,y,r,p,X)| \leq \sigma_2(|x-y|(|p|+1)) \tag{A.8}$$

for any $y \in \Omega$ and $(t,x,r,p,X) \in U \times \mathbb{R} \times (\mathbb{R}^n \setminus \{0\}) \times \mathcal{S}^n$.

Then if u^- and u^+ are viscosity subsolutions and supersolutions of (A.1), respectively, such that for some $K > 0$ independent of $t, x, y \in (0,T] \times \Omega \times \Omega$:

- *$u^-(t,x) \leq K(|x|+1)$ and $u^+(t,x) \geq -K(|x|+1)$;*
- *$(u^-)^*(t,x) - (u^+)_*(t,y) \leq m_T(|x-y|)$ on $\partial_p\big((0,T] \times (\Omega \times \Omega)\big)$;*
- *$(u^-)^*(t,x) - (u^+)_*(t,y) \leq K(|x-y|+1)$ on $\partial_p\big((0,T] \times (\Omega \times \Omega)\big)$.*

Then there is a modulus function σ such that

$$(u^-)^*(t,x) - (u^+)_*(t,y) \leq \sigma(|x-y|). \tag{A.9}$$

This generalized comparison principle, coupled with an argument which uses the framework given in [11], known as Perron's method, gives the existence and uniqueness of a viscosity solution to (3.28). Specifically, we note the parabolic version of this framework uses a result like the following, which is Lemma 2.3.15 from [20]

LEMMA A.7 (Perron process for parabolic equations). *Consider (A.1) where f is degenerate elliptic and continuous. Assume that u^+ and u^- are viscosity supersolutions and subsolutions, respectively. Then there exists a viscosity solution u such that $u^- \leq u \leq u^+$.*

The results above are summarized in the following theorem.

THEOREM A.8 (Unique viscosity solution). *Suppose that f satisfies the conditions of Lemma A.6 and Lemma A.7 and that there exists a viscosity supersolution u^+ and a viscosity subsolution u^- to (A.1). Then there exists a unique continuous viscosity solution to (A.1).*

Proof. According to Lemma A.7, there exists a viscosity solution u to (A.1). To show uniqueness and continuity, let v be another viscosity solution. By definition, u^* and v^* are subsolutions and u_* and v_* are supersolutions. Then (A.9), combined with the properties of envelopes imply that

$$u^* = (u^*)^* \leq (v_*)_* = v_* \leq v^* = (v^*)^* \leq (u_*)_* = u_* \leq u^*. \tag{A.10}$$

Thus $u = v$ and $u^* = u_*$ so that u is continuous. ∎

Acknowledgments. C.D.H and R.C.B. would like to thank Michael Herty for many helpful discussions.

REFERENCES

[1] *Top500 list.* https://www.top500.org/lists/2019/06/, June 2019.
[2] D. ARMBRUSTER, D. MARTHALER, AND C. RINGHOFER, *Kinetic and fluid model hierarchies for supply chains*, Multiscale Modeling & Simulation, 2 (2003), pp. 43–61.
[3] A. AW AND M. RASCLE, *Resurrection of "second order" models of traffic flow*, SIAM Journal on Applied Mathematics, 60 (2000), pp. 916–938.
[4] M. K. BANDA, M. HERTY, AND A. KLAR, *Gas flow in pipeline networks*, Networks and Heterogeneous Media, 1 (2006), pp. 41–56.
[5] G. BARLES, *Fully non-linear neumann type boundary conditions for second-order elliptic and parabolic equations*, Journal of Differential Equations, 106 (1993), pp. 90 – 106.
[6] G. BARLES, *An introduction to the theory of viscosity solutions for first-order hamilton–jacobi equations and applications*, in Hamilton-Jacobi Equations: Approximations, Numerical Analysis and APplications, P. Loreti and N. A. Tchou, eds., Springer Berlin Heidelberg, 2013.
[7] S. BIANCHINI AND D. TONON, *SBV regularity for Hamilton–Jacobi equations with Hamiltonian depending on (t, x)*, SIAM Journal on Mathematical Analysis, 44 (2012), pp. 2179–2203.
[8] J. BROUWER, I. GASSER, AND M. HERTY, *Gas pipeline models revisited: Model hierarchies, nonisothermal models, and simulations of networks*, Multiscale Modeling & Simulation, 9 (2011), pp. 601–623.
[9] P. CANNARSA AND H. FRANKOWSKA, *From pointwise to local regularity for solutions of Hamilton–Jacobi equations*, Calculus of Variations and Partial Differential Equations, 49 (2014), pp. 1061–1074.
[10] B. COCKBURN, *Continuous dependence and error estimation for viscosity methods*, Acta Numerica, 12 (2003), pp. 127–180.
[11] M. G. CRANDALL, H. ISHII, AND P.-L. LIONS, *User's guide to viscosity solutions of second order partial differential equations*, Bulletin of the American Mathematical Society, 27 (1992), pp. 1–67.
[12] D. CULLER, R. KARP, D. PATTERSON, A. SAHAY, K. E. SCHAUSER, E. SANTOS, R. SUBRAMONIAN, AND T. VON EICKEN, *Logp: Towards a realistic model of parallel computation*, SIGPLAN Not., 28 (1993), pp. 1–12.
[13] D. E. CULLER, R. M. KARP, D. PATTERSON, A. SAHAY, E. E. SANTOS, K. E. SCHAUSER, R. SUBRAMONIAN, AND T. VON EICKEN, *Logp: A practical model of parallel computation*, Commun. ACM, 39 (1996), pp. 78–85.
[14] E. DEELMAN, K. VAHI, G. JUVE, M. RYNGE, S. CALLAGHAN, P. J. MAECHLING, R. MAYANI, W. CHEN, R. F. DA SILVA, M. LIVNY, AND K. WENGER, *Pegasus, a workflow management system for science automation*, Future Generation Computer Systems, 46 (2015), pp. 17 – 35, doi:https://doi.org/10.1016/j.future.2014.10.008, http://www.sciencedirect.com/science/article/pii/S0167739X14002015.
[15] J. DONGARRA, J. HITTINGER, J. BELL, L. CHACÓN, R. FALGOUT, M. HEROUX, P. HOVLAND, E. NG, C. WEBSTER, AND S. WILD, *Applied mathematics research for exascale computing*, tech. report, U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research Program, 2014.

[16] S. Dosanjh, R. Barrett, D. Doerfler, S. Hammond, K. Hemmert, M. Heroux, P. Lin, K. Pedretti, A. Rodrigues, T. Trucano, and J. Luitjens, *Exascale design space exploration and co-design*, Future Generation Computer Systems, 30 (2014), pp. 46 – 58.
[17] Y. Giga, S. Goto, H. Ishii, and M.-H. Sato, *Comparison principle and convexity preserving properties for singular degenerate parabolic equations on unbounded domains*, Indiana Univ. Math. J., 40 (1991), pp. 443–470.
[18] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems*, vol. 8 of Springer Series in Computational Mathematics, Springer-Verlag Berlin Heidelberg, 2010.
[19] C. Hauck, M. Herty, and G. Visconti, *Qualitative properties of mathematical model for data flow*. submitted.
[20] C. Imbert and L. Silvestre, *An introduction to fully nonlinear parabolic equations*, in An Introduction to the Kähler-Ricci Flow, S. Boucksom, P. Eyssidieux, and V. Guedj, eds., vol. 2086, Springer International Publishing, 2013, pp. 7–88.
[21] E. R. Jakobsen and K. H. Karlsen, *Continuous dependence estimates for viscosity solutions of fully nonlinear degenerate parabolic equations*, Journal of Differential Equations, 183 (2002), pp. 497 – 525.
[22] A. Klar and R. Wegener, *A hierarchy of models for multilane vehicular traffic. I. Modeling*, SIAM J. Appl. Math., 59 (1999), pp. 983–1001 (electronic).
[23] J. M. Kunkel, *Simulating parallel programs on application and system level*, Computer Science - Research and Development, 28 (2013), pp. 167–174.
[24] A. Nunez, J. Fernandez, R. Filguiera, F. Garcia, and J. Carretero, *Simcan: A flexible, scalable and expandable simulation platform for modelling and simulating distributed architectures and applications*, Simulation Modelling Practice and Theory, 20 (2012), pp. 12–32.
[25] A. Pietracaprina and G. Pucci, *The complexity of deterministic pram simulation on distributed memory machines*, Theory of Computing Systems, 30 (1997), pp. 231–247, doi:10.1007/BF02679461, http://dx.doi.org/10.1007/BF02679461.
[26] L. Rifford, *On viscosity solutions of certain Hamilton–Jacobi equations: Regularity results and generalized sard's theorems*, Communications in Partial Differential Equations, 33 (2008), pp. 517–559.
[27] C.-W. Shu, *High order numerical methods for time dependent hamilton-jacobi equations*, in Mathematics and computation in imaging science and information processing, World Scientific, 2007, pp. 47–91.
[28] K. L. Spafford and J. S. Vetter, *Aspen: A domain specific language for performance modeling*, in Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC '12, Los Alamitos, CA, USA, 2012, IEEE Computer Society Press, pp. 84:1–84:11.
[29] R. Stevens and A. White, *Architectures and technology for extreme scale computing*, in ASCR Scientific Grand Challenges Workshop Series, 2009.
[30] G. Teschl, *Ordinary Differential Equations and Dynamical Systems*, vol. 140 of Graduate Studies in Mathematics, American Mathematical Society, 2012.
[31] J. S. Vetter and J. S. Meredith, *Synthetic program analysis with aspen*, in Proceedings of the 3rd International Conference on Exascale Applications and Software, Edinburgh, Scotland, UK, 04/2015 2015, University of Edinburgh, University of Edinburgh.
[32] W. Walter, *Ordinary Differential Equations*, vol. 182 of Graduate Texts in Mathematics, Springer-Verlag New York, 1998.