Sharpness
A Tight Condition for Scalability

Augustin Chaintreau

Thomson
augustin.chaintreau@thomson.net

Abstract. A distributed system is scalable if the rate at which it completes its computation and communication tasks does not depend on its size. As an example, the scalability of a peer-to-peer application that transmits data among a large group depends on the topology and the synchronization implemented between the peers. This work describes a model designed to shed light on the conditions that enable scalability. Formally, we model here a collection of tasks, each requiring a random amount of time, which are related by precedence constraints. We assume that the tasks are organized along an euclidean lattice of dimension d. Our main assumption is that the precedence relation between these tasks is invariant by translation along any of these dimensions, so that the evolution of the system follows Uniform Recurrence Equations (UREs). Our main result is that scalability may be shown under two general conditions: (1) a criterion called "sharpness" satisfied by the precedence relation and (2) a condition on the distribution of each task completion time, which only depends on the dimension d. These conditions are shown to be tight. This result offers a universal technique to prove scalability which can be useful to design new systems deployed among an unlimited number of collaborative nodes.

1 Introduction

Scalability is usually regarded as an important if not critical issue for any distributed communication/computation system. However, it is in general difficult to describe formally in mathematical terms. This paper describes a model designed to shed light on the impact of synchronization among an ever-increasing number of nodes participating in a distributed system. The results we provide answer the following question: "Under which general conditions is a distributed system scalable, in the sense that the rate of tasks' completion remains the same independently of the number of participating nodes?" We characterize in particular the impact of three factors: 1- the organization of the local feedback and synchronization mechanisms deployed between the nodes (acknowledgment, etc.), 2- the global topology of the communication (in this paper, an euclidean lattice), and 3- the variations of local delay, or local computation time, which is captured in this model by random times to complete each step. This general problem is primarily motivated today to study the throughput of communication protocols in large scale data networks.

Model. This paper analyzes the process of completion times $\mathbf{T}$ for a collection of tasks which are related together via precedence constraints. As an example of precedence constraint, a customer can be served in a server only once the previous customer has completed its service. As in a queueing system, each task, once it has started, takes a random amount of time to finish, which is called its *weight*. As an example, this random weight can represent either the time needed to exchange a message over a wireless link, a link shared with background data traffic, or the time needed to access some memory or processing unit. We assume that the weights of different tasks are independent random variables. The weight of a given task may occasionally take a large value (for instance, due to local congestion, server load, etc.); this is represented in the model by the weight distribution of this task. Hence, for a given initial condition of the system, the completion time of a task is a random variable that depends on the precedence relation with other tasks, and the (random) weights for all of them.

We focus here on the case where the collection of tasks is infinite and regular. Formally, we assume there exists a finite subset of indexes $\mathcal{H}$, which we call the *pattern*, and a dimension $d \geq 1$ such that the collection of tasks is

$$\left\{ \; a \times h \; \middle| \, a \in \mathbb{Z}^d, \; h \in \mathcal{H} \; \right\} .$$

In other words, one can describe this system as a finite set $\mathcal{H}$ of tasks to be done locally, which is reproduced at every position of a d-dimensional lattice. Moreover, we assume that the precedence constraints that relate all the tasks of the system together are *invariant by translation*. Thus, a given task $(a \times h)$ depends on different tasks, some have the same position a in the lattice and a different index $h' \neq h$ chosen in $\mathcal{H}$, some have different positions in the lattice and any index h' in $\mathcal{H}$. The above assumption guarantees that, after translation, these precedence relations do not depend on the position a.

Main result. In this paper we provide sufficient and necessary conditions for the following property to hold: for any $a \in \mathbb{Z}^d$, and any $h \in \mathcal{H}$ we have

$$\exists M \in \mathbb{R} \quad \text{such that almost surely} \quad \limsup_{m \to \infty} \frac{1}{m} T_{(m \cdot a) \times h} < M \, , \tag{1}$$

where $T_{a \times h}$ denotes the completion time for the task $(a \times h)$, and $(m \cdot a)$ denotes the vector $(m \cdot a_1, \ldots, m \cdot a_d) \in \mathbb{Z}^d$.

A system that satisfies the above property is called *scalable*, as this guarantees that the completion time grows linearly along any direction drawn in the lattice. Equivalently, the (random) set that contains all tasks completed before t grows with t according to a positive linear rate, in any given direction. As an example, if one dimension of $\mathbb{Z}^d$ denotes the sequence number of a packet to be received, this condition guarantees a positive throughput for each node in the system, even if the system itself is infinite.

We prove that scalability, as defined above, is characterized only by two conditions: one that deals with the organization of the precedence relation between tasks, which is called *sharpness*, and another condition that deals with the weight

distributions, and which only depends on d. We also prove that these two conditions are tight, and in particular that a non-sharp system in dimension $d = 2$ is never scalable.

Implications. The result above, although it is stated in quite abstract terms, has some important consequences for the design of distributed systems. First, it proves that a large class of systems are scalable although they implement some closed feedback loop among an infinite number of nodes. In particular it shows that distributed reliable systems can be implemented using only *finite* buffers in nodes, while remaining truly scalable. A few examples of this counter-intuitive fact have already been shown in [1,2]. What is new in this paper is that we identify the ingredients of such scalability in a systematic way.

Second, under appropriate moment condition on weights, the scalability of a distributed system is shown to be equivalent to the sharpness condition, which itself corresponds to a finite number of linear inequalities. Proving scalability is therefore greatly simplified, and keeping this condition in mind can even help dimensioning new distributed systems (as happened for instance in [3]).

The assumption that the precedence relation among tasks should be invariant by translation might seem restrictive at first. It is to some extent necessary since designing a general model for irregular systems seems difficult, not to mention finding exact conditions for their scalability. However, we would like to point here that our model allows for patterns $\mathcal{H}$ of arbitrary finite size. It is therefore sufficient to model systems that are regular only at a certain scale. Moreover, it is often the case that irregular systems are included in a larger one that is regular; it is then sufficient to prove scalability by inclusion using stochastic ordering. Last, we have assumed that the systems is organized along an euclidean lattice, although in practice many distributed systems are organized along other hierarchical topologies. The sharpness condition and the scalability result presented here can be extended to this case, although it is far beyond the scope of this paper. The interested reader may found first results on general graphs reported in [4].

Relation with previous work. Scalability has been addressed in the past for distributed communication protocols and congestion control [2,5,3] , stream processing [6] and computational grids [7]. All these works addressed the issue of scale for a distributed system where a form of synchronization is implemented between a large number of nodes. In contrast, our work treats a general case, and identify for the first time a tight condition that characterizes scalability.

Our results extend recent advances in stochastic network theory on infinite tandem of queues with general service time distribution and blocking [1]. Rather than studying a certain feedback mechanism, our results determine which feedback systems make the throughput independent of the system size. In addition, most of the previous results deal only with networks of single server queues. In contrast, our model applies for any pattern $\mathcal{H}$ that contains an arbitrary finite number of tasks. As an example, it can be used to characterize tandem of any finite timed event graph. Such generalization is made possible, as in [1], through

a formulation that reminds last-passage percolation time, which allows to use the powerful framework of subadditive ergodic theory [8]. However, no prior knowledge of last-passage percolation is needed to prove this result.

The systems we consider follow solutions of Uniform Recurrence Equations (UREs). UREs were introduced by Karp et al. in [9], where a general condition for the existence of a solution is presented. UREs have been used in the past to study synchronization in parallel computation, such as the discrete solution of differential equations. Our work points in a different direction: for the first time we study the solutions of general UREs when each step takes a random amount of time, and we characterize exactly when this solution grows linearly as its index grows. This is a stronger property as we prove that the sharpness criterion defined here is strictly stronger than the condition defined in [9] to classify UREs.

The organization of the paper is as follows. Section 2 presents the model sketched above in more details. Examples and relations with Uniform Recurrence Relations are explained. Section 3 defines the *sharpness* condition, and establishes the topological consequence of this criterion on the dependence paths between the tasks. Section 4 contains the main results of the paper. Section 5 concludes the paper with remarks on possible extensions.

NB: As for UREs, the boundary condition defining the system initial condition plays an important role in the analysis. In order to focus on the essential connection between sharpness and scalability, we choose to describe a single boundary case, corresponding to a system "initially empty". The result of this paper can be obtained under more general boundary condition (see [4] Chap.3).

2 Pattern Grid

2.1 Definition

Pattern grids are defined as directed graphs that follow an invariant property. Their vertices are indexed by both a multidimensional integer (*i.e.* the "position" of the task in the d-dimensional lattice $\mathbb{Z}^d$) as well as a local index chosen in a finite set $\mathcal{H}$. Formally, a graph $\mathcal{G}_{\texttt{patt}} = (\mathcal{V}, \mathcal{E})$ is called a *pattern grid*, with dimension d and pattern $\mathcal{H}$, if:

- The set of its vertices is $\mathcal{V} = \mathbb{Z}^d \times \mathcal{H}$.
- The set of its edges $\mathcal{E}$ is invariant by any translation in the lattice $\mathbb{Z}^d$;
  for all a, a', v in $\mathbb{Z}^d$ and h, h' in $\mathcal{H}$, we have
  $(a \times h) \to (a' \times h') \in \mathcal{E}$ if and only if $((a+v) \times h) \to ((a'+v) \times h') \in \mathcal{E}$.

In this work, we consider only the case of locally finite graphs (*i.e.* the number of edges that leave any vertex is finite). The invariant property implies that the degrees of vertices in this graph are uniformly bounded. We denote by H the cardinal of $\mathcal{H}$. When $H = 1$, the index h plays no particular role and the pattern grid follows a lattice.

To illustrate this definition, fix all coordinates in the left index (for instance, all of them null), consider the set of all the associated vertices (*e.g.* $(0,\ldots,0)\times\mathcal{H}$) and all the directed edges starting from any of those vertices. This defines a finite collection of local tasks to complete and relations between them and a few others "neighboring" tasks. The pattern grid is what is obtained when one reproduces this finite object at every site of an euclidean lattice.

2.2 Examples

Due to space constraints, we can only describe a few illustrating examples of the definitions above.

Infinite tandem of queues. The simplest case of a pattern grid with dimension 2 is an infinite line of single server queues (indexed by k) in tandem, serving customers (indexed by m). It works as follows: when a customer has completed its service in server k, he enters immediately the (infinite) buffer of server $(k+1)$, where it is scheduled according to a first-come-first-served discipline.

Let us consider all the tasks of the type "service of customer m in server k", which are naturally indexed by $(m,k)\in\mathbb{Z}^2$. The relation between them is essentially described by two precedence rules:

- $(m,k)\to(m,k-1)$ (*i.e.* the service of m in k cannot start unless its service in server $(k-1)$ is completed).
- $(m,k)\to(m-1,k)$ (*i.e.* the service of m in k cannot start unless the previous customer has completed its service in k).

Hence this system is well described by a pattern grid with dimension 2, a pattern containing a single element, and the above edges.

Infinite tandem of queues with blocking. Let us now assume as in [1] that each server implements two queues (input and output), both with a finite size $B_{\mathtt{IN}}$ and $B_{\mathtt{OUT}}$, such that buffers overflow are avoided by the appropriate blocking of service. It is not hard to see that such systems can be modeled by a pattern grid with dimension 2 and pattern $\mathcal{H}=\{i,o\}$. It contains the following edges:

- $(m,k)\times i\to(m,k-1)\times o$ (*i.e.* serving a customer in k requires that he was forwarded from the previous server $(k-1)$.)
- $(m,k)\times o\to(m,k)\times i$ (*i.e.* a customer cannot enter the output buffer before he has been served by this server.)
- $(m,k)\times i\to(m-1,k)\times i$ (*i.e.* serving a customer requires that the previous customer has been served.)
- $(m,k)\times i\to(m-B_{\mathtt{OUT}},k)\times o$ (*i.e.* avoid $B_{\mathtt{OUT}}$ overflow.)
- $(m,k)\times o\to(m-B_{\mathtt{IN}},k+1)\times i$ (*i.e.* avoid $B_{\mathtt{IN}}$ overflow.)

Fig.1 describes a portion of the graph defined in the two above examples. Other mechanisms of feedback fit in the same model. As an example, one may model TCP connections, with varying windows, organized in tandem and that implement back-pressure blocking [2].

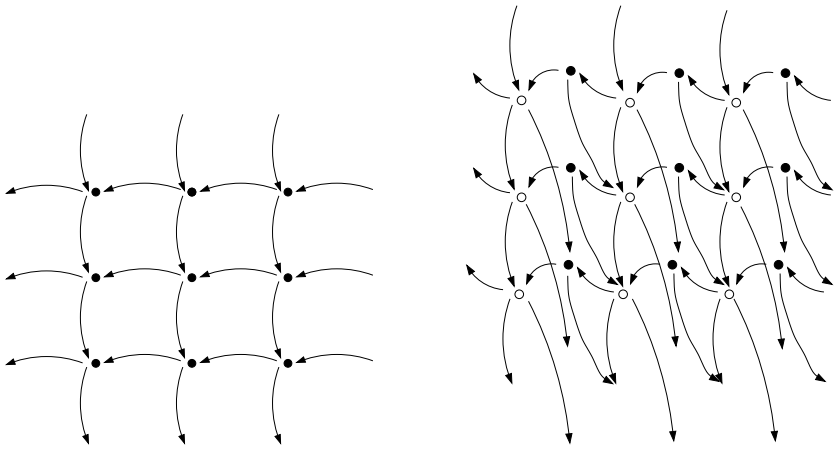

Fig. 1. Two examples of pattern grids: infinite tandem of single server queue (left), the same with input-output blocking (right) (i is represented by a white dot, o by a black dot, we set $B_{\mathtt{IN}} = 1$, $B_{\mathtt{IN}} = 2$).

2.3 Evolution equation of a pattern grid

We now define how the relations between tasks, represented by edges, describe the evolution of the system: the edge $(a \times h) \to (a' \times h')$ in $\mathcal{E}$ represents that the task $(a \times h)$ cannot start unless $(a' \times h')$ has been completed. Vertex $(a' \times h')$ is then called an *immediate predecessor* of $(a \times h)$.

We associate with each vertex $(a \times h)$ of the pattern grid a *weight* denoted by $\mathtt{W}(a \times h)$, representing the time needed to complete this task, once it has started. For instance in the two examples shown above, this weight is the service time of customer m in server k. It is generally random; sometimes it can be taken equal to zero (like for $(m,k) \times o$ in the second example above, if the delay between two servers is neglected).

We consider the process of completion time for every task

$$\mathbf{T} = \left\{ \; T_{a \times h} \in \mathbb{R} \cup \{-\infty\} \;\; \middle| \; a \in \mathbb{Z}^d, \; h \in \mathcal{H} \; \right\} .$$

Assuming that a task begins as soon as all its immediate predecessors have been completed, we have for all a in $\mathbb{Z}^d$ and h in $\mathcal{H}$:

$$T_{a \times h} = \mathtt{W}(a \times h) + \max \left\{ T_{a' \times h'} \middle| \, (a \times h) \to (a' \times h') \in \mathcal{E} \right\} . \tag{2}$$

As an example, for the infinite tandem of queues described above it becomes Lindley's equation:

$$T_{(m,k)} = \mathtt{W}(m,k) + \max \left(T_{(m,k-1)}, T_{(m-1,k)} \right) .$$

Relation with UREs. It is easy to see that another way to characterize the set $\mathcal{E}$ of edges in a pattern grid is via a collection of *dependence sets*: a finite collection $(\Delta_{h,h'})_{h,h'\in\mathcal{H}}$ of subsets of $\mathbb{Z}^d$ indexed by $\mathcal{H}^2$, such that

$$(a \times h) \to (a' \times h') \in \mathcal{E} \text{ if and only if } (a' - a) \in \Delta_{h,h'} . \tag{3}$$

Note that, as the graph is supposed locally finite, all these subsets are necessarily finite. Following this definition, Eq.(2) may be rewritten as

$$T_{a\times h} = \mathtt{W}(a \times h) + \max\left\{ T_{(a+r)\times h'} \middle| r \in \Delta_{h,h'},\ h' \in \mathcal{H} \right\} . \tag{4}$$

which defines a set of Uniform Recurrence Equations (UREs), already introduced in [9]. These systems of equations have been studied as they characterize dependencies between computation tasks in a parallel computation. The article by Karp et al. is motivated by the numerical resolution of discrete version of classical differential equations.

Boundary condition. Let $\mathtt{beg}$ be chosen in $\mathcal{H}$. We assume that the system starts to complete the task $(0 \times \mathtt{beg})$ at time $t = 0$, and that the system is "initially empty" (*i.e.* all tasks $(a \times h)$ such that a has at least one negative coordinate are supposed to be initially complete). In other words, we introduce $\mathcal{G}^{[0]}_{\mathtt{patt}}$ the pattern grid where the weight of any task $(a \times h)$ is replaced by $-\infty$ whenever a has at least one negative coordinate. The process $\mathbf{T}$ is then a solution of the following system of equations:

$$\begin{cases} T_{0\times\mathtt{beg}} = \mathtt{W}(0 \times \mathtt{beg}) , \\ T_{a\times h} = \mathtt{W}(a \times h) + \max\limits_{(a\times h)\to(a'\times h')\in\mathcal{E}} T_{a'\times h'} \ , \text{ for } (a \times h) \neq (0 \times \mathtt{beg}). \end{cases} \tag{5}$$

One can immediately check that the following defines a solution of Eq.(5):

$$\forall a \in \mathbb{Z}^d, h \in \mathcal{H} , \quad T_{a\times h} = \sup\left\{ \mathtt{W}(\pi) \ \middle| \ \begin{array}{c} \pi \text{ a path in } \mathcal{G}^{[0]}_{\mathtt{patt}} \\ \pi : a \times h \rightsquigarrow 0 \times \mathtt{beg} \end{array} \right\} , \tag{6}$$

where a *path* π is defined following the natural definition of paths in directed graph, and its weight $\mathtt{W}(\pi)$ is the sum of the weights of all its vertices. The supremum is taken over all possible paths, where we include paths that contain a single vertex and no edge.

Note that whenever a contains a negative coordinate, the supremum is equal to $-\infty$. When the precedence relation between tasks is acyclic (*i.e.* when the system has no deadlock, see the next section) one can show by induction that this solution is unique for every task $(a \times h)$ for which a path exists in the above supremum.

One may rephrase Eq.(6) as "The completion time of task $(a \times h)$ is the maximum sum of weights along a dependence path leading from $(a \times h)$ back to the origin task $(0\times\mathtt{beg})$." By an analogy with models from statistical physics, this may be called the *last-passage percolation time* in $(a \times h)$. It is important to note that, as in percolation model, these variables exhibit super-additive property, such that one can benefit from the subadditive ergodic theorem which generalizes the law of large number [8].

3 Sharpness

In this section, we characterize the properties of the paths in the pattern grid with a single condition: the sharpness criterion. We first prove under this condition that the combinatorial properties of these paths follow the connected subsets of a lattice (also called *lattice animals* [10]). When this condition is not verified, we show in dimension 2 that the combinatorial properties of these paths are radically different.

3.1 Definitions

Dependence graph, simple cycle For any pattern grid, we define the associated *dependence graph* as the following directed multi-graph, where all edges are labeled with a vector in $\mathbb{Z}^d$:

$$\begin{cases} \text{Its set of vertices is } \mathcal{H}. \\ \text{Its set of edges is } \{h \to h' \text{ with label } r \mid r \in \Delta_{h,h'},\ h, h' \in \mathcal{H}\}\ . \end{cases}$$

Note that according to this definition, $h \to h'$ is an edge of the dependence graph with label r if and only if for all $a \in \mathbb{Z}^d$, $(a \times h) \to ((a + r) \times h')$ is in $\mathcal{E}$. We represent the dependence graph associated with the two examples of 2.2 in Figure 2.

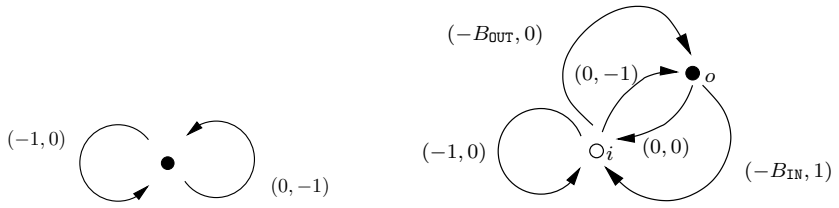

Fig. 2. Two examples of dependence graphs: for an infinite tandem (left), same with input and output blocking (right).

A *path* in the dependence graph follows the natural definition of a path in a graph. Its *size* is given by the number of vertices it contain (the same vertex can be included multiple times); its *associated vector* is the sum of the label for the edges that it contains.

A path drawn in the dependence graph which begins and ends in the same vertex h of $\mathcal{H}$ is called a *cycle*. It is a *simple cycle* if it does not contain any other cycle. In other words, all vertices visited by this cycle are distinct except the first and last one, which are necessarily the same. As a consequence, a simple cycle contains at most $H + 1$ vertices (including multiplicity), and the collection of simple cycles is finite.

Sharpness condition We denote by $\mathcal{C}$ the set of all vectors associated with a simple cycle in the dependence graph. For two vectors u and v in $\mathbb{Z}^d$, $\texttt{<}u, v\texttt{>}$ denotes their scalar product, $\texttt{<}u, v\texttt{>} = \sum_{i=1}^{d}(u_i \cdot v_i)$.

Condition 1 *The following conditions are equivalent*

(i) *There exists* $s \in \mathbb{Z}^d$, *such that* $\forall r \in \mathcal{C}$, $\texttt{<}r, s\texttt{>} < 0$.
(ii) *There exists* $s \in \mathbb{Z}^d$, *such that* $\forall r \in \mathcal{C}$, $\texttt{<}r, s\texttt{>} \leq -1$.
(iii) *There exists a hyperplane of* $\mathbb{R}^d$ *such that all vectors in* $\mathcal{C}$ *are contained in an open half-space defined by this hyperplane.*

A pattern grid is then called sharp. A vector s *satisfying* (ii) *is called a sharp vector.*

Condition (iii) may be seen as a rewriting of (i) in geometric terms, (i) implies (ii) since the family $\mathcal{C}$ contains a finite number of vectors. In practice, to determine whether a sharp vector exists, one has to extract all the simple cycles in the dependence graph, and then to solve a finite system of linear inequalities.

As an example $s = (1, 1)$ is a sharp vector for the infinite tandem in Section 2.2. For the second example, $s = (2, 1)$ is a sharp vector since we obviously assume $B_{\texttt{IN}} + B_{\texttt{OUT}} \geq 1$.

Geometric interpretation. Let us define the *cone* $\texttt{Cone}(\mathcal{C})$ containing all linear combinations of elements in $\mathcal{C}$ with non-negative coefficients. Assuming that the pattern grid is sharp, this cone intersects one hyperplane only in 0 and is otherwise contained in one of the open half space defined by this hyperplane. In other words, the angle of this cone should be acute. By analogy, the pattern grid is then called "sharp".

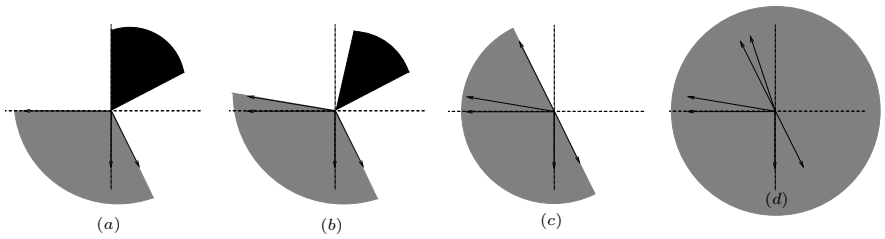

Fig. 3. Geometric representation: (a) and (b) represent families that admit a sharp vector, (c) and (d) families that do not admit such a vector.

Some examples are shown in Figure 3 for the case of dimension 2. Different families $\mathcal{C}$ have been represented, containing from 3 to 6 vectors. The cone generated by positive linear combination of this family is shown in gray. We have shown in black the directions that can define a sharp vector, for the cases (a)

and (b), where such a vector may be found. Case (c) shows an example of family containing opposite vectors, making it impossible to find a sharp vector. In the case (d), the cone created by positive linear combination of vectors is the whole space $\mathbb{R}^2$, such that, again, no sharp vector may be found. We prove in §3.3, that these cases depict all possible situations for dimension 2.

Relation with deadlock avoidance. A loop in the graph defining the pattern grid corresponds to a deadlock of the system, since it denotes that a task indirectly depends on its own completion to start. Karp et. al proved necessary and sufficient conditions to avoid such deadlock, and showed that they characterize system of Uniform Recurrence Equations where an explicit solution can be constructed [9]. Note that a deadlock corresponds to a cycle drawn in the dependence graph whose associated vector is null. In other words, the system has no deadlock if and only if $0 \notin \mathcal{C}$.

One may immediately observe that sharpness implies deadlock avoidance (*e.g.* as a direct consequence of (i)). It is however less obvious that the sharpness condition is indeed strictly stronger than deadlock avoidance (see Section 4.2).

3.2 why is a sharp vector useful ?

The main consequence of sharpness is that one can define a direction in the grid so that dependence paths between tasks remain close to that direction. It comes from the following fact: a path in the dependence graph is more or less a concatenation of a large number of cycles. We can then limit the size of a dependence path based only on its direction in the grid.

Lemma 1. *Suppose that a pattern grid admits a sharp vector s. There exist two constants B, C such that, for any path $\pi : (a \times h) \rightsquigarrow (a' \times h')$*

$$|\pi| \leq B + C \cdot \ \texttt{<}a - a', s\texttt{>} \ .$$

Proof. Let us introduce the *residue* of a pattern grid, for a sharp vector s.

$$\texttt{Res} = \max \left\{ (\ \texttt{<}r, s\texttt{>}\ )^+ | \ r \ \text{ assoc. with } \pi \text{ and } |\pi| \leq H \ \right\} .$$

It is a finite maximum by definition, because the dependence graph contains a finite number of vertices and edges. Note that for the case where the motif set $\mathcal{H}$ contains a single element, this residue is null, because every path is a cycle. The above result is implied by the following result on paths in the dependence graph: for any path π associated with r, we have $|\pi| \leq H\,(1 + \texttt{Res} - \ \texttt{<}r, s\texttt{>}\ )$.

We will prove this fact by induction on the size of π. First, from the definition of the residue, this result holds trivially for any path π whose size is less than or equal to H.

If π has a size strictly larger than H, then it contains a cycle, and hence a simple cycle, σ. We can write $\pi = \pi_1 \circ \sigma \circ \pi_2$. The path $\pi_1 \circ \pi_2$ is well defined, as the vertex ending π_1 is also the one starting π_2. The path $\pi_1 \circ \pi_2$ has necessarily

a smaller size than π. If we assume by induction that it satisfies the result of the theorem, we can deduce:

$$\begin{aligned}
|\pi| &\leq |\pi_1 \circ \pi_2| + |\sigma| - 1 \leq |\pi_1 \circ \pi_2| + H + 1 - 1 \\
&\leq H(1 + \mathtt{Res} - \ \mathtt{<}r_{\pi_1} + r_{\pi_2}, s\mathtt{>}\ ) + H \\
&\leq H(1 + \mathtt{Res} - \ \mathtt{<}r_{\pi_1} + r_{\pi_2}, s\mathtt{>}\ ) - H\ \mathtt{<}r_\sigma, s\mathtt{>}\ ,\ (\ \text{ since }\ \mathtt{<}r_\sigma, s\mathtt{>}\ \leq -1\ \ ) \\
&\leq H(1 + \mathtt{Res} - \ \mathtt{<}r_\pi, s\mathtt{>}\ ) .
\end{aligned}$$

This bound only depends on the positions in the grid of the two extreme nodes of this path. Hence this result provides an upper bound on the size of any dependence path between two given tasks in the pattern grid. We need slightly more than that: we aim at bounding the maximum weight of any path between two given nodes. Hence we have to capture in addition the combinatorial property of the sum of weights found in such paths.

We can address this second problem as follows: Since $\mathcal{H}$ is a finite set, we can construct a one-to-one correspondence between $\mathbb{Z}^d \times \mathcal{H}$ and $\mathbb{Z}^d$. Let us introduce the following definition, we say that a subset of $\mathbb{Z}^d$ is lattice-connected if it is connected according to the neighbor relation of an undirected lattice. A subset of $\mathbb{Z}^d \times \mathcal{H}$ is called lattice-connected if its associated subset in $\mathbb{Z}^d$ (by the above correspondence) is lattice connected.

Lemma 2. *Suppose that a pattern grid admits a sharp vector s. There exist two constants B, C such that, for any path $\pi : (a \times h) \rightsquigarrow (a' \times h')$*

$$\exists \xi \ \ \textit{a lattice-connected set, such that}\ \ \pi \subseteq \xi \ \ \textit{and}\ \ |\xi| \leq B + C \cdot\ \mathtt{<}a - a', s\mathtt{>}\ .$$

Proof. The proof is an application of Lemma 1. We introduce the *radius* of a pattern grid as the finite maximum

$$\mathtt{Rad} = \max \{||r||_\infty \ \text{ for }\ r \in \Delta_{h,h'}, h, h' \in \mathcal{H}\}\ .$$

It is the maximum difference on one coordinate between vertices $a \times h$ and $a' \times h'$ that are connected by an edge in the pattern grid. One can show that any path π may be augmented into a lattice-connected subset which contains at most $|\pi| \cdot (H + d \cdot \mathtt{Rad})$. A formal version of this argument, based on a one-to-one correspondence between $\mathbb{Z}^d \times \mathcal{H}$ and $\mathbb{Z}^d$ is detailed in [4].

This turns out to be a very powerful tool, because the class of lattice-connected subset, also known as lattice animals, have been well characterized from a combinatorial and probabilistic standpoint [10],[11].

3.3 why is a sharp vector necessary ?

Focusing on dimension $d = 2$, we describe properties of non-sharp pattern grids. These results are only used later to prove that sharpness is a tight condition of scalability. We start by a result showing that a non-sharp pattern grid always exhibits some pathological case: the proof of this lemma may be found in [12].

Lemma 3. *If we consider a family of vectors $\mathcal{C}$ in $\mathbb{Z}^2$ that does not admit a sharp vector, then one of the following statements is true:*

(*i*) *It contains the vector* 0.

(*ii*) *It contains two opposite vectors: there exist e and f in $\mathcal{C}$ such that*

$$\exists \alpha \in \mathbb{R}, \alpha > 0\,, \quad \textit{such that} \quad e = -\alpha \cdot f\,.$$

(*iii*) *It contains a generating triple: there exist e, f, g in $\mathcal{C}$ such that*

$$\begin{cases} \texttt{<}e, f\texttt{>} < 0 & \texttt{<}e, g\texttt{>} < 0 \\ \texttt{<}\tilde{e}, f\texttt{>} > 0 & \texttt{<}\tilde{e}, g\texttt{>} < 0 \end{cases} \textit{, with} \quad \texttt{<}\tilde{e}, e\texttt{>} = 0\,.$$

Let us define a pattern grid as *irreducible* if its dependence graph is strongly connected (*i.e.* there always exists a path leading from h to h', for any h and h'). A non-irreducible pattern grid can be decomposed using the strongly connected components of the dependence graph, and studied separately. The next result, a consequence of Lemma 3 proves that dependence paths in non-sharp pattern grid cannot be bounded as in Lemma 1. Due to space constraint, we omit the proof which may be found in Appendix B.2 of [12].

Corollary 1. *We consider a pattern grid, irreducible, with dimension $d = 2$ that does not admit a sharp vector. We pick an arbitrary vertex of this graph as an origin. There exists a vertex $v \times h$, such that we can build a path of size arbitrary large from $v \times h$ to the origin.*

4 Scalability

In this section, we establish the main result of this paper: under a moment condition, a distributed system represented by a sharp pattern grid is scalable. Moreover, we prove that the rate of completion along any direction converges to a deterministic constant. We then prove that the sharp condition is necessary for scalability, at least for dimension 2, when one avoids degenerate cases. A simple example is provided to illustrate how sharpness is a stronger condition than the one defined in [9].

4.1 The sharp case

Moment condition. The weight of $a \times h$ is supposed to follow that depends only on h and is upper bounded by $\bar{s}$, for the stochastic ordering,

$$\forall u \in \mathbb{R}\,, \quad \text{we have} \quad \mathbb{P}\left[\mathtt{W}(a \times h) \geq u\right] \leq \mathbb{P}\left[\bar{s} \geq u\right]\,.$$

Condition 2 *We assume* $\displaystyle\int_0^{+\infty} P(\bar{s} \geq u)^{1/d} du < \infty\,.$

Condition 2 implies $\mathbb{E}[(\bar{s})^d] < +\infty$. It is implied by $\mathbb{E}[(\bar{s})^{d+\epsilon}] < +\infty$ for any positive ϵ, but it is slightly more general.

Theorem 1. *Let $\mathcal{G}_{\mathtt{patt}}$ be a pattern grid satisfying Condition 1 and 2, then*

(*i*) *The system is scalable.*

$$\exists M \in \mathbb{R} \quad \textit{such that almost surely} \quad \limsup_{m\to\infty} \frac{1}{m} T_{(m\cdot a)\times h} < M\,.$$

(*ii*) *If there exists a path $\pi : a \times \mathtt{beg} \rightsquigarrow 0 \times \mathtt{beg}$ with non-negative coordinates (i.e. such that $\mathtt{W}(\pi) \neq -\infty$ and $T_{a\times \mathtt{beg}} \neq -\infty$), then*

$$\lim_{m\to\infty} \frac{T_{(m\cdot a)\times \mathtt{beg}}}{m} = l \in \mathbb{R} \quad \textit{almost surely and in } \mathbb{L}^1.$$

As a consequence, the throughput of an infinite number of queues organized in tandem is positive, with or without blocking, whenever Condition 2 is verified by the service time. Condition 2 is almost tight since one can build a counter-example when $\mathbb{E}[(\bar{s})^d] = \infty$ [1].

Proof. We prove a slightly more general result, that for any a there exists $M \in \mathbb{R}$ such that, almost surely

$$\max_{h,h'\in\mathcal{H}} \limsup_{m\to\infty} \frac{1}{m} \left(\sup_{\pi:\ ((m\cdot a)\times h)\rightsquigarrow(0\times h')} \mathtt{W}(\pi) \right) \leq M < +\infty$$

Theorem 1.1 in [11] tells us that in a lattice $\mathbb{Z}^d$, where weights satisfy Condition 2, the maximum weight of a lattice-connected set ξ grows linearly: There exists $N \in \mathbb{R}$ such that, when $n \to \infty$,

$$\frac{1}{n} \left(\max_{\xi \ \text{ latt. conn., } \ |\xi|=n\ ,\ 0\in\xi} \mathtt{W}(\xi) \right) \to N < \infty \ \text{ almost surely and in } \mathbb{L}^1.$$

For any fixed h and h', as a consequence of Lemma 2, a path π leading from $((m \cdot a) \times h)$ to $(0 \times h')$ is contained in a lattice connected subset ξ with size smaller than

$$|\xi| \leq (H^2 + d \cdot \mathtt{Rad} \cdot H)(1 + \mathtt{Res} + m \cdot \ \texttt{<}a, s\texttt{>}\ )\,,$$

where s is a sharp vector for this grid. All these connected subset contain in particular $0 \times h'$. The weight of a path is then upper bounded by the maximum weight of a connected subset that contains this fixed point. We deduce

$$\limsup_{m\to\infty} \frac{1}{m} \sup_{\pi:\ ((m\cdot a)\times h)\to(0\times h')} \mathtt{W}(\pi) \leq (H^2 + d \cdot \mathtt{Rad} \cdot H) \cdot \ \texttt{<}a, s\texttt{>}\ \cdot N < \infty\,.$$

The proof of (*ii*) relies on the sub-additive ergodic theorem. It is omitted due to space constraint and may be found in the Appendix A of [12].

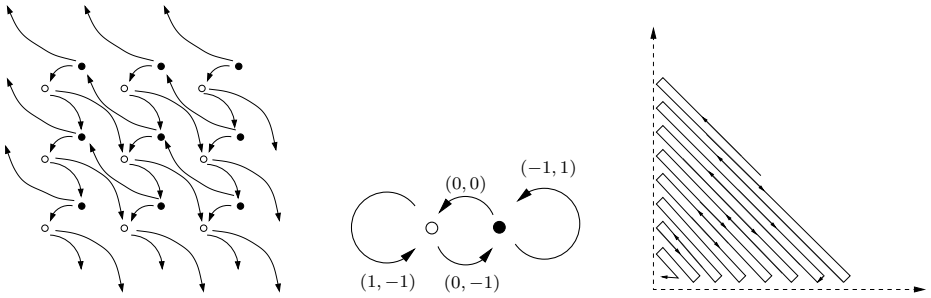

Fig. 4. Example of a pattern grid that avoids deadlock but does not satisfy the sharpness condition: represented via pattern grid (left), dependence graph (middle), shape of a path with a "super-linear" size (right).

4.2 The non-sharp case

Let us first illustrate with an example the case of a non-sharp pattern grid. The pattern grid represented in Figure 4 avoids deadlock since one cannot build a cycle in the dependence graph associated with a null vector. It is not sharp as two opposite vectors are associated with cycles in the dependence graph.

As shown in Figure 4 on the right, one can construct a path from any vertex to the origin by following first the top left direction (remaining only over black vertices), following an edge towards a white vertex before crossing the y-axis, then following a right bottom line, remaining on white vertices. It can be seen that starting from coordinate (m, k), the length of this path for large m and k is of the order of $(m + k)^2$. This proves that the $T_{(m,0)\times b} \approx m^2$, and thus that the growth rate associated with this direction is more than linear (*i.e.* its associated completion rate in this direction is zero).

We now prove that the phenomenon found above is not an exceptional case but that it always occurs when the sharp condition is not verified. Just like for the study made in Section 3.3, we consider irreducible pattern grid with dimension 2. The proof of the following result is in Appendix B.3 in [12].

Theorem 2. *Let $\mathcal{G}_{\texttt{patt}}$ be an irreducible pattern grid with dimension 2 that does not admit a sharp vector and $\mathbf{T}$ be any solution of Eq.(5).*

We assume that there exists a, with only positive coordinates, and $h \in \mathcal{H}$, such that a path $a \times h \rightsquigarrow 0 \times h$ exists and always has non-negative coordinates. We also assume that all weights are non negative and not identically null, then

$$\exists a' \in \mathbb{Z}^d\,,\ \lim_{m\to\infty} \frac{T_{(m\cdot a')\times h}}{m} = +\infty \ \textit{ almost surely and in expectation.}$$

5 Concluding Remarks

In this paper we have proved a general result on the scalability of distributed systems. It was obtained in two steps. First, we have proved that the dependence

paths in a general precedence relation can be bounded by a scalar product whenever a *sharp vector* exists. Second, the completion of tasks associated with the system have been analyzed taking advantage of subadditive ergodic theory. The scalability result implies that the random set that contains all completed tasks grows linearly with time, in any direction. This work refines the classification of distributed computing systems introduced in [9] via Uniform Recurrence Equations (UREs).

Some aspects of this method have not been included in the paper, due to space constraint. Let us now review them briefly. All the results presented here can be shown for the case of a random pattern grid (*i.e.* where the collection of edges is random, with a distribution invariant per translation). Similarly, the same result can be obtained with different boundary conditions, allowing to construct stationary regimes and analyze stability of large scale system. Another important extension that we could not describe in this paper is when the topology does not follow a lattice but different types of infinite graphs, such as trees. We refer to [4] for a first account of these extensions, which will be described more in future work. In a more longer term, we wish to study how other synchronization between nodes (like the presence of conflicting services) may impact scalability.

References

1. Martin, J.: Large tandem queuing networks with blocking. Queuing Systems, Theory and Applications **41** (2002) 45–72
2. Baccelli, F., Chaintreau, A., Liu, Z., Riabov, A.: The one-to-many TCP overlay: A scalable and reliable multicast architecture. In: Proceedings of IEEE INFOCOM. Volume 3. (2005) 1629–1640
3. He, J., Chaintreau, A.: BRADO: scalable streaming through reconfigurable trees (extended abstract). In: Proceedings of ACM Sigmetrics. (2007) 377–378
4. Chaintreau, A.: Processes of Interaction in Data Networks. PhD thesis, INRIA-ENS (2006) www.di.ens.fr/~chaintre/research/AugustinChaintreauPhD.pdf.
5. Jelenkovic, P., Momcilovic, P., Squillante, M.S.: Buffer scalability of wireless networks. In: Proceedings of IEEE INFOCOM. (2006) 1–12
6. Xia, C., Liu, Z., Towsley, D., Lelarge, M.: Scalability of fork/join queueing networks with blocking. In: Proceedings of ACM Sigmetrics. (2007) 133–144
7. Chen, L., Reddy, K., Agrawal, G.: Gates: A grid-based middleware for processing distributed data streams. In: HPDC '04: Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing. (2004) 192–201
8. Kingman, J.: Subadditive ergodic theory. Annals of Probability **1**(6) (1973) 883–909
9. Karp, R.M., Miller, R.E., Winograd, S.: The organization of computations for uniform recurrence equations. J. ACM **14**(3) (1967) 563–590
10. Gandolfi, A., Kesten, H.: Greedy lattice animals II: linear growth. Annals Appl. Prob. **1**(4) (1994) 76–107
11. Martin, J.: Linear growth for greedy lattice animals. Stochastic Processes and their Applications **98**(1) (2002) 43–66
12. Chaintreau, A.: Sharpness: a tight condition for scalability. Technical Report CR-PRL-2008-04-0001, Thomson (2008)