MULTISCALE MODEL. SIMUL.
Vol. 16, No. 1, pp. 1–27

STATIONARY AVERAGING FOR MULTISCALE CONTINUOUS TIME MARKOV CHAINS USING PARALLEL REPLICA DYNAMICS*

TING WANG†, PETR PLECHÁČ†, AND DAVID ARISTOFF‡

Abstract. We propose two algorithms for simulating continuous time Markov chains in the presence of metastability. We show that the algorithms correctly estimate, under the ergodicity assumption, stationary averages of the process. Both algorithms, based on the idea of the parallel replica method, use parallel computing in order to explore metastable sets more efficiently. The algorithms require no assumptions on the Markov chains beyond ergodicity and the presence of identifiable metastability. In particular, there is no assumption on reversibility. For simpler illustration of the algorithms, we assume that a synchronous architecture is used throughout the paper. We present error analyses, as well as numerical simulations on multiscale stochastic reaction network models in order to demonstrate consistency of the method and its efficiency.

Key words. Markov chains, Monte Carlo, reversibility, stationary distribution, metastability, parallel replica, stochastic reaction networks, multiscale dynamics, coarse graining

AMS subject classifications. 60J22, 65C05, 65Z05, 82B31, 92E20

DOI. 10.1137/16M1108716

1. Introduction.

We focus on computing stationary averages of continuous time Markov chains in a synchronous architecture. More precisely, if π is the stationary distribution of a continuous time Markov chain (CTMC) and f is a function on the state space, we aim at estimating the average $\pi(f) \equiv \mathbb{E}_\pi[f]$ by taking a time average on a long trajectory of the CTMC. There are many methods for computing stationary averages of stochastic processes. However, the vast majority of them rely on reversibility of the process, e.g., as in Markov chain Monte Carlo [19]. Computational cost of the ergodic (trajectory) averaging becomes prohibitive when the convergence to the stationary distribution is slow due to metastability of the dynamics, for example, in the presence of rare events or large time scale disparities (multiscale dynamics) [20]. A possible remedy for these issues is to use parallel computing in order to accelerate sampling of the state space. For instance, the parallel tempering method (also known as the replica exchange) [12, 10, 9, 17] has been successfully applied to many problems by simulating multiple replicas of the original systems, each replica at a different temperature. However, the method requires the time reversibility of the underlying processes, which is typically not true for processes that model chemical reaction networks or systems with nonequilibrium steady states. In fact, there are not many methods that parallelize Monte Carlo simulation for irreversible processes with metastability, in particular if long-time sampling, such as ergodic averaging, is required. We present a parallel computing approach for CTMCs without time reversibility. One advantage of the proposed algorithms is that they may be used, in principle, on arbitrary CTMCs. The same idea applies to the continuous state

*Received by the editors December 20, 2016; accepted for publication (in revised form) October 6, 2017; published electronically January 2, 2018.
http://www.siam.org/journals/mms/16-1/M110871.html
Funding: The first author was supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program under contract DE-SC0010549. The second author was partially supported by DARPA project W911NF-15-2-0122. The third author was supported by the National Science Foundation via award NSF-DMS-1522398.
†University of Delaware, Newark, DE 19716 (tingw@udel.edu, plechac@math.udel.edu).
‡Colorado State University, Fort Collins, CO 80523 (aristoff@rams.colorado.edu).

space Markov processes. However, gains in efficiency can occur only if the process is metastable.

In this contribution we consider only models described by CTMCs. As a motivating example we study a multiscale chemical reaction network model in which molecules of different types react with different rates depending on their concentrations and reaction rate constants. In this model metastability emerges due to the infrequent occurrence of reactions with small rates which makes the relaxation to the steady state dynamics extremely slow. In the transient regime the finite time distribution can be approximated using the stochastic averaging technique [24, 14], or the tau-leap method [18]. However, the former does not apply for stationary distribution estimation and that the error they introduce on the stationary state is generally difficult to evaluate; the latter can still be computationally expensive for long-time simulations. It is thus desirable to have an efficient algorithm for computing the stationary averages. Thus the proposed algorithm will provide a new multiscale simulation method (in particular for stationary averaging estimation) for the stochastic reaction networks community.

The presented approach builds on the parallel replica (ParRep) dynamics introduced in the context of molecular simulations in [22]. The ParRep method used in the context of stochastic differential equations, e.g., Langevin dynamics, was rigorously analyzed in [15, 16]. The algorithm we present and analyze builds on the recent work of [1, 2] where the ParRep process was studied for discrete-time Markov chains. In our algorithms, each time the simulation reaches a local equilibrium in a metastable set W, R independent replicas of the CTMC are launched inside the set allowing for parallel simulations of the dynamics at this stage. The main contribution of this work is a procedure for using the replicas in order to *efficiently* and *consistently* estimate the exit time and exit state from W, along with the contribution to the *stationary time average* of f from the time spent in W. We emphasize that we are able to handle arbitrary functions (or observables) on the state space, not only those that are piecewise constant, i.e., assuming a single value in each W. In the best case, if there are R replicas, then the simulation leaves a metastable set about R times faster compared to a direct serial simulation. The consistency of our algorithms relies on certain properties of the quasi-stationary distribution (QSD) which are essentially local equilibria associated with the metastable sets.

We propose two algorithms for computing $\pi(f)$, called *CTMC ParRep* and *embedded ParRep*. The former uses parallel simulation of the CTMC, while the latter employs parallel simulation of its embedded chain, which is a discrete time Markov chain (DTMC). CTMC ParRep (resp., embedded ParRep) relies on the fact that, starting at the QSD in a metastable set, the first time to leave the set is an exponential (resp., geometric) random variable and independent of the exit state; see Theorem 2.5 below. The algorithms require some methods for identifying metastable sets, though this need not be done a priori—it is sufficient to identify when the CTMC is currently in a metastable set, and when it exits such set. While both algorithms can be useful for efficient simulation of $\pi(f)$ in the presence of metastability, we expect the embedded ParRep can be significantly more efficient, especially when combined with a certain type of QSD sampling, called Fleming–Viot [3, 4]. Though we focus here on the computation of $\pi(f)$, we note that one of our algorithms, CTMC ParRep, can be used to compute the dynamics of the CTMC on a coarse space in which each metastable set is considered a single (meta-)state. See the discussion below Algorithm 1.

The advantages of the proposed algorithms include: (a) no requirement of time

reversibility for the underlying dynamics; (b) they are suitable for long-time sampling; (c) they may be used, in principle, on arbitrary CTMCs in the presence of metastability.

In section 2, we briefly review CTMCs before defining QSDs and detailing relevant properties thereof. In section 3, we present CTMC ParRep, and study how the error in the algorithm depends on the quality of QSD sampling. In section 4, we present embedded ParRep and provide an analogous error analysis. We detail some numerical experiments on multiscale chemical reaction network model in section 5 in order to demonstrate the consistency and accuracy of the algorithms.

2. Background and problem formulation.

2.1. Continuous time Markov chains. Throughout this paper, $X(t)$ is an irreducible and positive recurrent CTMC with values in a countable set E and π denotes the stationary distribution of $X(t)$. We are interested in computing stationary averages $\pi(f)$ for a bounded function $f : E \to \mathbb{R}$ by using the ergodic theorem

$$\lim_{t\to\infty} \frac{1}{t}\int_0^t f(X(s))ds = \pi(f), \tag{1}$$

which holds almost surely for any initial distribution of $X(t)$. The jump times τ_n and holding times $\Delta\tau_n$ for $X(t)$ are defined recursively by

$$\tau_0 = 0, \qquad \tau_n = \inf\{t > \tau_{n-1} : X(t) \neq X(\tau_{n-1})\},$$

and

$$\Delta\tau_{n-1} = \tau_n - \tau_{n-1}$$

for $n \geq 1$. We assume that $X(t)$ is nonexplosive, that is, $\lim_n \tau_n = \infty$ almost surely for every initial distribution of $X(t)$. This precludes the possibility of infinitely many jumps in finite time. We denote $X_n = X(\tau_n)$ the embedded chain of $X(t)$. It is easy to see that X_n is a DTMC.

Recall that $X(t)$ is completely determined by its infinitesimal generator matrix $Q = \{q(x, y)\}_{x,y\in E}$. Recall that by convention q(x,x) is chosen such that $\sum_y q(x, y) = 0$ and we write $q(x) := -q(x, x)$. Note that irreducibility implies $q(x) > 0$ for all $x \in E$. It is easy to check that X_n has the transition probability matrix $P = \{p(x, y)\}_{x,y\in E}$ satisfying

$$p(x,y) = \begin{cases} \frac{q(x,y)}{q(x)}, & x \neq y, \\ 0, & x = y. \end{cases}$$

We state the following well-known fact for the later reference.

LEMMA 2.1. *For a CTMC $X(t)$ with the corresponding embedded Markov chain X_n, the holding time between successive jumps $\Delta\tau_0, \Delta\tau_1, \ldots, \Delta\tau_i, \ldots$ are independent conditioned on the embedded chain X_n. Moreover, $\Delta\tau_i|\{X_n\}$ is exponentially distributed with the rate $q(X_i)$ and hence $\mathbb{E}[\Delta\tau_i|\{X_n\}] = q(X_i)^{-1}$.*

For details on the above facts, see, for instance, [5].

2.2. The quasi-stationary distribution and metastability. Below, we write $\mathbb{P}$, $\mathbb{E}$ for various probabilities and expectations, the precise meaning of which will be clear from context. We use a superscript $\mathbb{P}^\xi$, $\mathbb{E}^\xi$ to indicate that the initial distribution is ξ. When the initial distribution is δ_x, we write $\mathbb{P}^x$, $\mathbb{E}^x$. The symbol $\sim$ will indicate

equality in probability law. $\mathrm{Re}(\cdot)$ and $|\cdot|$ denote the real part and modulus of a complex number.

Our ParRep algorithms rely on certain properties of quasi-stationary distributions, which we now briefly review. Let $W \subset E$ be fixed and consider the first exit time of $X(t)$ from W, that is,

$$T = \inf\{t > 0; X(t) \notin W\}.$$

We consider also the first exit time of X_n from W,

$$N = \inf\{n > 0; X_n \notin W\}.$$

A QSD of $X(t)$ in W (or X_n in W) is defined as follows.

DEFINITION 2.2. *A probability distribution ν with support in W is a QSD for $X(t)$ in W if for each $y \in W$ and $t > 0$,*

$$\nu(y) = \mathbb{P}^{\nu}(X(t) = y \,|\, T > t). \tag{2}$$

Similarly, a probability distribution μ with support in W is a QSD for X_n in W if for each $y \in W$ and $n > 0$,

$$\mu(y) = \mathbb{P}^{\mu}(X_n = y \,|\, N > n). \tag{3}$$

Throughout, we write ν for a QSD of the CTMC $X(t)$ and μ for a QSD of the embedded chain X_n. The associated set W will be implicit since no ambiguities should arise. We will write

$$\nu_t(A) = \mathbb{P}^{x}(X(t) \in A \,|\, T > t) \tag{4}$$

for the distribution of $X(t)$ conditioned on $T > t$, and

$$\mu_n(A) = \mathbb{P}^{x}(X_n \in A \,|\, N > n) \tag{5}$$

for the distribution of X_n conditioned on $N > n$. Notice we do not make explicit the dependence on the starting point x.

We summarize existence, uniqueness, and convergence properties of the QSD in Theorem 2.3 below (see [6, 21]). In Theorem 2.3 below, for simpler presentation, we assume W is finite. That allows us to characterize convergence to the QSD of $X(t)$ and X_n in terms of spectral properties of their generator and transition matrices. We emphasize, however, there exists more general results to guarantee the convergence to the QSD and hence the finiteness of W is not necessary for consistency of the algorithms proposed in this paper.

Recall that Q is the infinitesimal generator matrix of $X(t)$ and P is the transition probability matrix of the DTMC X_n. We denote $Q_W = \{q_{xy}\}_{x,y\in W}$ and $P_W = \{p_{xy}\}_{x,y\in W}$ the restrictions of P and Q to W.

THEOREM 2.3. *Let W be finite and nonabsorbing for $X(t)$, and assume P_W is irreducible.*

(a) *The eigenvalues $\lambda_1, \lambda_2, \ldots$ of Q_W can be ordered so that*

$$0 > \lambda_1 > \mathrm{Re}(\lambda_2) \geq \ldots,$$

where λ_1 has the left eigenvector ν which is a probability distribution on W. Moreover, ν is the unique QSD of $X(t)$ in W, and for all $x, y \in W$,

$$|\nu_t(y) - \nu(y)| = |\mathbb{P}^{x}(X(t) = y | T > t) - \nu(y)| \leq C(x) e^{-(\lambda_1 - \beta)t}, \tag{6}$$

with $C(x)$ a constant depending on x, and β any real number satisfying $\mathrm{Re}(\lambda_2) < \beta < \lambda_1$.

(b) *Suppose P_W is also aperiodic. Then the eigenvalues $\sigma_1, \sigma_2, \ldots$ of P_W can be ordered so that*

$$1 > \sigma_1 > |\sigma_2| \geq \ldots,$$

where σ_1 has the left eigenvector μ which is a probability distribution on W. Moreover, μ is the unique QSD of X_n in W and for all $x, y \in W$,

$$|\mu_n(y) - \mu(y)| = |\mathbb{P}^x(X_n = y | N > n) - \mu(y)| \leq D(x) \left(\frac{\gamma}{\sigma_1}\right)^n, \tag{7}$$

with $D(x)$ a constant depending on x, and γ any real number satisfying $|\sigma_2| < \gamma < \sigma_1$.

Proof. We first justify the expression for the eigenvalues. Observe that for $x \neq y \in W$, we have $q(x, y) > 0$ if and only if $p(x, y) > 0$. It follows that Q_W is irreducible if and only if P_W is irreducible; see Definition 2.1 in [21]. Now let I be the all ones column vector, $I(x) = 1$ for $x \in W$. Recall that $q(x, y) \geq 0$ for every $x \neq y \in E$ and $\sum_y q(x, y) = 0$ for every $x \in E$. This implies that $Q_W I \leq 0$ componentwise. Since W is nonabsorbing, for some $x \in W$ and $y \notin W$ we have $q(x, y) > 0$, and it follows that $\sum_{z \in W} q(x, z) < 0$. This shows that at least one component of $Q_W I$ is strictly negative. The expression for the eigenvalues, and the fact that ν is signed (hence a probability distribution, after normalization) now follows from Theorem 2.6 of Seneta [21].

To see ν is the QSD for $X(t)$ in W, we define the stopped process $X^T(t) = X(t \wedge T)$ such that $X(t)$ is absorbed outside W. For any $x, z \in E$, let I_x be the column vector $I_x(z) = 1$ if $x = z$ and $I_x(z) = 0$ otherwise. Finiteness of W ensures that $\mathbb{P}^x(X^T(t) = y) = I_x' e^{Q_W t} I_y$. Thus, for each $y \in W$,

$$\mathbb{P}^\nu(X(t) = y, T > t) = \mathbb{P}^\nu(X^T(t) = y) = \nu e^{Q_W t} I_y = e^{\lambda_1 t} \nu(y)$$

and

$$\mathbb{P}^\nu(T > t) = \mathbb{P}^\nu(X^T(t) \in W) = e^{\lambda_1 t},$$

which leads to $\nu(y) = \mathbb{P}^\nu(X(t) = y | T > t)$.

Now we turn to the convergence to ν. It follows from Theorem 2.7 in [21] that there is a constant $C(x)$ depending on x such that for any real β with $\mathrm{Re}(\lambda_2) < \beta$,

$$\mathbb{P}^x(X(t) = y, T > t) = \mathbb{P}^x(X^T(t) = y) = C(x) e^{\lambda_1 t} \nu(y) + \mathcal{O}(e^{\beta t}) \tag{8}$$

and

$$\mathbb{P}^x(T > t) = C(x) e^{\lambda_1 t} + \mathcal{O}(e^{\beta t}). \tag{9}$$

It follows that

$$|\nu_t(t) - \nu(y)| = |\mathbb{P}^x(X(t) = y \,|\, T > t) - \nu(y)| \leq C(x) e^{-(\lambda_1 - \beta)t}, \qquad \square$$

where $C(x)$ is now a (possibly different) constant depending on x.

The arguments in (b) are similar, using the Perron–Frobenius theorem (Seneta [21, Theorem 1.1]).

For analogous results on the QSD in more general settings, see [6, Theorem 4.5] for CTMCs and [8, Theorem 1] for DTMCs. We are now ready to define metastability.

DEFINITION 2.4. *Let W and λ_i, σ_i be as in Theorem* 2.3.

1. *W is metastable for $X(t)$ if $\lambda_1 \approx 0$ and*

$$|\lambda_1| \ll |\lambda_1 - \mathrm{Re}(\lambda_2)|. \tag{10}$$

$X(t)$ is metastable if it has at least one metastable set W.

2. *W is metastable for X_n if $\sigma_1 \approx 1$ and*

$$\sigma_1 \gg \frac{|\sigma_2|}{\sigma_1}. \tag{11}$$

X_n is metastable if it has at least one metastable set W.

In light of Theorem 2.3, conditions 1–2 in Definition 2.4 essentially say that the time to leave W is large in an absolute sense, and the time to leave W is large relative to the time to converge to the QSD in W. Metastability of the CTMC is not necessarily equivalent to the metastability of its underlying embedded chain, as we now show. Consider $X(t)$ with the infinitesimal generator

$$Q = \begin{pmatrix} -1 & 1/2 & 1/2 & 0 \\ 1/2 & -1 & 1/2 & 0 \\ 0 & \epsilon/2 & -\epsilon & \epsilon/2 \\ 0 & 0 & 1 & -1 \end{pmatrix},$$

where $\epsilon \approx 0$ is positive. Then $W = \{1, 2, 3\}$ is metastable for $X(t)$ but not for X_n, since

$$\sigma_1 \approx 0.81, \quad |\sigma_2| \approx 1/2, \qquad \lambda_1 \approx -\epsilon/2, \quad \mathrm{Re}(\lambda_2) \approx -1/2.$$

Now consider $X(t)$ with the infinitesimal generator

$$Q = \begin{pmatrix} -\epsilon^{-1} & \epsilon^{-1}/2 & \epsilon^{-1}/2 & 0 \\ \epsilon^{-1} - 1 & -\epsilon^{-1} & 1 & 0 \\ 0 & \epsilon^{-1} - 1 & -\epsilon^{-1} & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

Then $W = \{1, 2, 3\}$ is metastable for X_n but not for $X(t)$, since

$$\sigma_1 \approx 1 - \epsilon/5, \quad |\sigma_2| \approx \sqrt{2}/2, \qquad \lambda_1 \approx -1/5, \quad \mathrm{Re}(\lambda_2) \approx -3\epsilon^{-1}/2.$$

Algorithm 1 below requires a collection of metastable sets for $X(t)$, and Algorithm 2 requires a collection of metastable sets for X_n. The only assumption we make on these sets is that they are pairwise disjoint. (The sets may be different for the two algorithms, as noted above.) Throughout we write W to denote a generic metastable set. We emphasize that *we do not assume the metastable sets form a partition of E*: the union of the metastable sets may be a proper subset of E. Here and below, we assume that each W has a unique QSD and that ν_t (and μ_t) converge to the QSD in total variation norm, for any starting point x. Recall that this is true under the assumptions of Theorem 2.3.

We conclude this section by mentioning properties of the QSD which are essential for the consistency of our algorithms in sections 3 and 4 below.

THEOREM 2.5.

1. *Suppose $X(0) \sim \nu$. Then T is exponentially distributed with the parameter $-\lambda_1$:*
$$\mathbb{P}^\nu(T > t) = e^{\lambda_1 t}, \qquad t > 0,$$
and T and $X(T)$ are independent.
2. *Suppose $X_0 \sim \mu$. Then N is geometrically distributed with the parameter $1 - \sigma_1$:*
$$\mathbb{P}^\mu(N > n) = \sigma_1^n, \qquad n = 1, 2, \ldots,$$
and N and X_N are independent.

Proof. The first part of 1 and 2 was shown in Theorem 2.3. For the rest of the proof, see [6]. ∎

3. The CTMC ParRep method.

3.1. Formulation of the CTMC algorithm. In this section, we introduce a method for accelerating the computation of $\pi(f)$, where we recall $f : E \to \mathbb{R}$ is any bounded function and π is the stationary distribution. We call this algorithm, CTMC ParRep, for reasons that will be outlined below. Before we describe CTMC ParRep, we introduce some notation. Throughout, $X^1(t), \ldots, X^R(t)$ will be independent processes with the same law as $X(t)$ and with initial distributions supported in W. Recall that the first exit time of $X(t)$ from W is

$$T = \inf\{t > 0 : X(t) \notin W\}.$$

Similarly, for $r = 1, \ldots, R$, we define the first exit time of $X^r(t)$ from W by

$$T^r = \inf\{t > 0 : X^r(t) \notin W\}$$

and the smallest one among them by

$$T^* = \min_r T^r.$$

We denote the index of the replica with the first exit time T^* by M, i.e.,

$$M = \arg\min_r T^r.$$

T, T^r, T^*, and M depend on W, but we do not make this explicit.

We are in the position to present the CTMC ParRep in Algorithm 1. In this algorithm, we will need user-chosen parameters t_c associated with each metastable set W. Roughly speaking, these parameters correspond to the time for $X(t)$ to converge to the QSD in W. The accumulated value $F(f)_{\text{sim}}$ serves as a quantity that approximates the integral $\int_0^{T_{\text{end}}} f(X(s))\, ds$ when the algorithm terminates. Note that at the end of the algorithm we often have $T_{\text{sim}} \geq T_{\text{end}}$ since the ParRep process could reach T_{end} during the parallel stage. However, this is not an issue as long as T_{sim} is large enough at the end of the algorithm so that the time average is well approximated.

If $X^{\text{par}}(t)$ remains in W for a sufficiently long time (i.e., decorrelation threshold t_c), it is distributed nearly according to the QSD ν of $X(t)$ in W by Theorem 2.3. This means that at the end of the decorrelation stage, $X^{\text{par}}(T_{\text{sim}})$ can be considered a sample of ν.

The aim of the dephasing stage is to prepare a sequence of independent initial states with distribution ν. There are several ways to achieve this. Perhaps the

Algorithm 1 CTMC ParRep

1: Set a decorrelation threshold t_c for each metastable set W. Initialize the simulation time clock $T_{\rm sim} = 0$ and the accumulated value $F(f)_{\rm sim} = 0$. We will write $X^{\rm par}(t)$ for a simulation process that obeys the law of $X(t)$. A complete ParRep cycle consists of three stages.

2: **Decorrelation stage**: Starting at $t = T_{\rm sim}$, evolve $X^{\rm par}(t)$ until it spends an interval of the time length t_c inside the same metastable set W. That is, evolve $X^{\rm par}(t)$ from time $t = T_{\rm sim}$ until time

$$T_{\rm corr} = \inf\{t \geq T_{\rm sim} + t_c : X^{\rm par}(s) \in W \text{ for all } s \in [t - t_c, t] \text{ for some } W\}.$$

Then update

$$F(f)_{\rm sim} = F(f)_{\rm sim} + \int_{T_{\rm sim}}^{T_{\rm corr}} f(X^{\rm par}(t))\, dt,$$

set $T_{\rm sim} = T_{\rm corr}$, and proceed to the dephasing stage.

3: **Dephasing stage**: Let W be such that $X^{\rm par}(T_{\rm sim}) \in W$, that is, W is the metastable set from the end of the last decorrelation stage. Generate R independent samples $x_1, \ldots, x_R$ from ν, the QSD of $X(t)$ in W.
Then proceed to the parallel stage.

4: **Parallel stage**: Start R parallel processes $X^1(t), \ldots, X^R(t)$ at $x_1, \ldots, x_R$, and evolve them simultaneously from time $t = 0$ until time $T^* = \min_r T^r$. Here all R processes are simulated in parallel. Then update

$$\begin{aligned} F(f)_{\rm sim} &= F(f)_{\rm sim} + \sum_{r=1}^{R} \int_0^{T^*} f(X^r(s)) ds, \\ T_{\rm sim} &= T_{\rm sim} + RT^*, \end{aligned} \tag{12}$$

set $X^{\rm par}(T_{\rm sim}) = X^M(T^*)$, and return to the decorrelation stage.

5: The algorithm is stopped when $T_{\rm sim}$ reaches a user-chosen terminal time $T_{\rm end}$. The stationary average $\pi(f)$ is estimated as

$$\pi(f) \approx F(f)_{\rm sim}/T_{\rm sim}.$$

simplest is the rejection method. In this procedure, each of the R replicas evolve independently. A parameter t_p similar to the decorrelation threshold t_c is selected. If a replica leaves W before spending a time interval of length t_p in W, it restarts in W from the original initial state. Once all the replicas remain in W for time t_p, we stop and take $x_1, \ldots, x_R$ as the final states of all the replicas in the dephasing stage and use them for the subsequent parallel stage. Besides rejection sampling, another method is a Fleming–Viot based particle sampler; see the discussion after Algorithm 2 below. Finally, we comment that we can reduce the overhead related to the dephasing stage by starting the dephasing stage immediately (instead of waiting for decorrelation stage to finish) when the $X^{\rm Par}(t)$ enters into a new metastable set [3].

The acceleration of CTMC ParRep comes from the parallel stage. Recall that,

for each $r = 1, \ldots, R$, if $x_1, \ldots, x_R$ are independent, identically distributed (iid) with the common distribution ν, then $T^1, \ldots, T_R$ are independent exponential random variables with common parameter λ_1. Using $T^* = \min_r T^r$, it is then easy to check that RT^* has the same distribution as T^1. See Lemma 3.1 below. This means one need only wait for T^* instead of T^1 to observe an exit from W. Note that this is true whether or not W is metastable, so efficiency of the parallel stage does not require metastability. However, the dephasing stage is not efficient if W is not metastable. That is because, in practice, the samples $x_1, \ldots, x_R$ are obtained by simulating trajectories which remain in W for a sufficiently long time t_p. Such samples are hard to obtain when the typical time t_p for $x_1, \ldots, x_R$ to reach the QSD in W is not much smaller than the typical time to leave W.

To see that each parallel stage has a consistent contribution to the stationary average, we make the following two observations. Suppose that $x_1, \ldots, x_R$ are iid samples from ν.

1. The joint law of $(RT^*, X^M(T^*))$ is the same as that of $(T^1, X(T^1))$. That is, the joint distribution of the first exit time and the exit state in the parallel stage is independent of the number of replicas.
2. The expected value of $\sum_{r=1}^{R} \int_0^{T^*} f(X^r(s))ds$ in (12) is the same as that of $\int_0^{T^1} f(X^1(s))ds$. That is, the expected contribution to $F(f)_{\text{sim}}$ from each parallel stage is independent of the number of replicas.

The first observation is a consequence of Theorem 2.5, and the second will be proved in Theorem 3.2 below. Consistency of stationary averages follows from points 1–2 above and the law of large numbers. Since there are indefinitely many parallel stages in a given W, consistency is ensured as long as the expected contribution to $F(f)_{\text{sim}}$ from the parallel stage has the correct expected value. See [1] for details and discussion in a related discrete time version of the algorithm under some idealized assumptions.

The CTMC ParRep algorithm suffers some serious drawbacks. Even if the parallel processors are synchronous, M and T^* may not be known at the wall clock time when the first replica leaves W. The reason is that the holding times for a CTMC are random, while the wall clock time for simulating each jump of the CTMC is always roughly the same. We illustrate this problem in Figure 1. In the worst possible case, in order to determine M and T^*, we must wait for all the replicas to leave W. However, one can set a variable $T_{\min}$ to record the current minimum first exit time over all replicas which have left W, and terminate any replicas which reach time $T_{\min}$ but have not left W, since no replica contributes to the accumulated value past time $T_{\min}$. Since the expected first exit times $\mathbb{E}[T^r], r = 1, \ldots, R$, are roughly the same, if the variance in the number of jumps of $X^r(t)$ before time T^* is small for all $r = 1, \ldots, R$, then we can expect that the parallel stage stops after only a few replicas leave W.

For the same reason, there is another major drawback of CTMC ParRep. If f takes multiple values in W, then the computation of $\sum_{r=1}^{R} \int_0^{T^*} f(X^r(s))ds$ in (12) requires storing the entire history of the value of f on each replica in that parallel stage. We illustrate this drawback by considering the case of two replicas r_1 and r_2 with first exit time T^1 and T^2, respectively. Suppose $T^1 < T^2$ and hence $T^* = T^1$. Let us assume that in terms of wall clock time, r_2 exits W before r_1 does. At the end of the parallel stage (i.e., after both of T^1 and T^2 are sampled) we have the running sum $\int_0^{T^2} f(X^2(s))ds$ from r_2. However, by the CTMC ParRep algorithm, we only need $\int_0^{T^*} f(X^2(s))ds$ indeed. If we only keep track of the running sum, we are unable

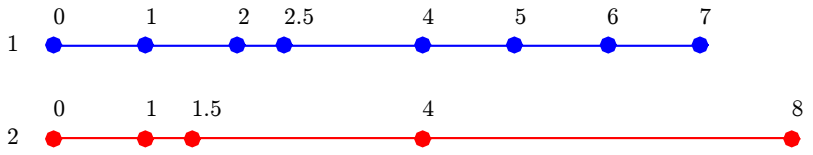

FIG. 1. *The parallel stage of the CTMC ParRep algorithm with two replicas. R*1 *escapes from* W *at* $t = 7$ *with seven transitions while R*2 *escapes at* $t = 8$ *but with only four transitions. In the parallel stage of the CTMC ParRep algorithm, R*2 *escaped from* W *before R*1 *does but* $T^2 > T^1$. *There is no acceleration in this case since the parallel stage does not terminate when R*2 *escapes.*

to recover $\int_0^{T^*} f(X^2(s))ds$ from T^2. Hence, the implementation of the CTMC ParRep might be memory demanding unless one is interested in the equilibrium average of a metastable-set invariant function f, i.e., if $f(x)$ has only one value in each metastable set W. In section 4 we present another algorithm, called embedded ParRep, which addresses these drawbacks.

3.2. Error analysis of CTMC ParRep. Here and below we will write $\mathbb{E}^{\nu^R}$ for the expectation of $(X^1(t), \dots, X^R(t))$ starting at ν^R, where for

$$\nu^R(x_1, \dots, x_R) = \prod_{r=1}^{R} \nu(x_r), \qquad x_1, \dots, x_R \in W.$$

We begin with a simple well-known lemma.

LEMMA 3.1. *Suppose* $T^1, \dots, T^R$ *are iid exponential random variables with the parameter* λ_1. *Then* $T^* = \min_{1 \le r \le R} T^r$ *is exponentially distributed with the parameter* $R\lambda_1$. *In particular,* RT^* *has the same distribution as* T^1.

We now show that if the dephasing sampling is exact, then the expected contribution to the accumulated value $F(f)_{\text{sim}}$ from the parallel step of Algorithm 1 is exact.

THEOREM 3.2. *Suppose in the dephasing step* $(x_1, \dots, x_R) \sim \nu^R$. *Then the expected contribution to* $F(f)_{\text{sim}}$ *from the parallel stage of Algorithm* 1 *is independent of the number of replicas,*

$$\mathbb{E}^{\nu^R}\left[\sum_{r=1}^{R} \int_0^{T^*} f(X^r(s))ds\right] = \mathbb{E}^{\nu}\left[\int_0^{T} f(X(s))ds\right] = \nu(f)\mathbb{E}^{\nu}[T].$$

Proof. First we consider the case with a single replica. We condition on the exit time T^1 and write

$$\mathbb{E}^{\nu}\left[\int_0^{T^1} f(X^1(s))ds\right] = \int_0^{\infty} \mathbb{E}^{\nu}\left[\int_0^{t} f(X^1(s))ds \,\middle|\, T^1 = t\right] \mathbb{P}^{\nu}(T^1 \in dt).$$

Interchanging the two integrals of the right-hand side leads to

$$\int_0^{\infty} \int_s^{\infty} \mathbb{E}^{\nu}\left[f(X^1(s))|T^1 = t\right] \mathbb{P}(T^1 \in dt)ds.$$

Note that the inner integral can be written as $\mathbb{E}^{\nu}\left[f(X^1(s))\mathbb{1}_{T^1>s}\right]$ and hence

$$\mathbb{E}^{\nu}\left[\int_0^{T^1} f(X^1(s))ds\right] = \int_0^{\infty} \mathbb{E}^{\nu}\left[f(X^1(s))|T^1>s\right]\mathbb{P}^{\nu}(T^1>s)ds.$$

Owing to the definition of QSD and the fact that $\mathbb{E}^{\nu}[T^1] = \int_0^{\infty}\mathbb{P}^{\nu}(T^1>s)ds$,

$$\mathbb{E}^{\nu}\left[\int_0^{T^1} f(X^1(s))ds\right] = \nu(f)\mathbb{E}^{\nu}[T^1].$$

In the case of multiple replicas, similar steps can be used to show that

$$\sum_{r=1}^{R}\mathbb{E}^{\nu^R}\left[\int_0^{T^*} f(X^r(s))ds\right] = \sum_{r=1}^{R}\int_0^{\infty}\mathbb{E}^{\nu^R}\left[f(X^r(s))|T^*>s\right]\mathbb{P}^{\nu^R}(T^*>s)ds.$$

Recall that $T^*>s$ if and only if $T^r>s$ for all $r=1,\dots,R$. Using this, the fact that $T^1,\dots,T^r$ are independent, and the definition of the QSD, we get

$$\mathbb{E}^{\nu^R}\left[f(X^r(s))|T^*>s\right] = \mathbb{E}^{\nu}\left[f(X^r(s))|T^r>s\right] = \nu(f).$$

Thus

$$\sum_{r=1}^{R}\mathbb{E}^{\nu^R}\left[\int_0^{T^*} f(X^r(s))ds\right] = \nu(f)\sum_{r=1}^{R}\int_0^{\infty}\mathbb{P}^{\nu^R}(T^*>s)ds = \nu(f)R\mathbb{E}^{\nu^R}[T^*].$$

Finally, the result follows from Lemma 3.1. ∎

The purpose of CTMC ParRep is to efficiently simulate very long trajectories of a metastable CTMC and estimate the equilibrium average $\pi(f)$. CTMC ParRep can produce accelerated dynamics of the CTMC on a coarse state space where each coarse set corresponds to some W; see the discussion below Algorithm 2 below. Our numerical experiments suggest that CTMC ParRep (and also embedded ParRep described below) are consistent for estimating the stationary distribution.

For CTMC ParRep, we justify this claim in Theorem 3.3 below, which shows that, starting in some W and waiting until the simulation leaves W, the error for a complete ParRep cycle in CTMC ParRep compared to direct (serial) simulation vanishes as t_c increases. See Theorem 4.4 below for the analogous result on embedded ParRep. We note that each ParRep cycle produce an error in the estimation of stationary averages that does not disappear as $T_{\text{sim}}\to\infty$. However, we expect that the error vanishes as the thresholds $t_c = t_p \to \infty$. Study of this error is more involved and will be the focus of another work.

Recall we have assumed convergence of $\|\nu_{t_c} - \nu\|_{\text{TV}} \to 0$ as $t_c\to\infty$, for every starting point $x\in E$, where $\|\cdot\|_{\text{TV}}$ denotes total variation norm; see, for instance, Theorem 2.3 for conditions guaranteeing this convergence.

THEOREM 3.3. *Consider CTMC ParRep starting at $x\in W$ in the decorrelation stage. Assume the dephasing stage sampling is exact, that is, $(x_1,\dots,x_R)\sim\nu^R$. Consider the expected contribution to $F(f)_{sim}$ until the first time the simulation leaves W (either in the decorrelation or in the parallel stage),*

$$\Delta F(f)_{sim} \triangleq \mathbb{E}^x\left[\int_0^{t_c\wedge T} f(X(s))\,ds\right] + \mathbb{E}^{x,\nu^R}\left[\mathbb{1}_{T>t_c}\sum_{r=1}^{R}\int_0^{T^*} f(X^r(s))ds\right],$$

where $\mathbb{E}^{x,\nu^R}$ *denotes expectation for* $(X(t), X^1(t), \ldots, X^R(t))$ *with* $X(t)$ *starting at* x *and the replicas* $(X^1(t), \ldots, X^R(t))$ *starting at initial distribution* ν^R*. The error compared to direct (serial) simulation satisfies the bound*

$$\left| \mathbb{E}^x \left[\int_0^T f(X(s)) ds \right] - \Delta F(f)_{sim} \right| \leq \|f\|_\infty \sup_{x \in W} \mathbb{E}^x [T] \, \|\nu_{t_c} - \nu\|_{\mathrm{TV}}. \tag{13}$$

Proof. We estimate

$$\begin{aligned}
&\left| \mathbb{E}^x \left[\int_0^T f(X(s)) ds \right] - \Delta F(f)_{\mathrm{sim}} \right| \\
&= \left| \mathbb{E}^x \left[\int_{t_c \wedge T}^T f(X(s)) ds \right] - \mathbb{E}^{x,\nu^R} \left[\mathbb{1}_{T > t_c} \sum_{r=1}^R \int_0^{T^*} f(X^r(s)) ds \right] \right| \\
&= \left| \mathbb{E}^x \left[\int_{t_c}^T f(X(s)) ds \,\middle|\, T > t_c \right] - \mathbb{E}^{x,\nu^R} \left[\sum_{r=1}^R \int_0^{T^*} f(X^r(s)) ds \,\middle|\, T > t_c \right] \right| \mathbb{P}^x(T > t_c) \\
&\leq \left| \mathbb{E}^x \left[\int_{t_c}^T f(X(s)) ds \,\middle|\, T > t_c \right] - \mathbb{E}^{\nu^R} \left[\sum_{r=1}^R \int_0^{T^*} f(X^r(s)) ds \right] \right|,
\end{aligned}$$

where we used the fact that $X(t)$ and the replicas $(X^1(t), \ldots, X^R(t))$ are independent. By the Markov property,

$$\mathbb{E}^x \left[\int_{t_c}^T f(X(s)) ds \,\middle|\, T > t_c \right] = \mathbb{E}^{\nu_{t_c}} \left[\int_0^T f(X(s)) ds \right].$$

By Theorem 3.2,

$$\mathbb{E}^{\nu^R} \left[\sum_{r=1}^R \int_0^{T^*} f(X^r(s)) ds \right] = \mathbb{E}^{\nu} \left[\int_0^T f(X(s)) \, ds \right].$$

Combining the above estimates and equalities,

$$\begin{aligned}
&\left| \mathbb{E}^x \left[\int_0^T f(X(s)) ds \right] - \Delta F(f)_{sim} \right| \\
&\leq \left| \mathbb{E}^{\nu_{t_c}} \left[\int_0^T f(X(s)) ds \right] - \mathbb{E}^{\nu} \left[\int_0^T f(X(s)) \, ds \right] \right| \\
&= \left| \sum_{x \in W} \mathbb{E}^x \left[\int_0^T f(X(s)) ds \right] \nu_{t_c}(x) - \sum_{x \in W} \mathbb{E}^x \left[\int_0^T f(X(s)) ds \right] \nu(x) \right| \\
&\leq \|f\|_\infty \sup_{x \in W} \mathbb{E}^x [T] \, \|\nu_{t_c} - \nu\|_{\mathrm{TV}}.
\end{aligned}$$

∎

We note that $\mathbb{E}^x[T]$ is uniformly bounded in $x \in W$ if, for instance, P_W is irreducible and W is finite and nonabsorbing for $X(t)$, as in Theorem 2.3. This uniform boundedness guarantees that the right-hand side of (13) vanishes as $t_c \to \infty$.

4. The embedded ParRep method.

4.1. Formulation of the embedded ParRep algorithm. In this section, we introduce another algorithm for accelerating the computation of $\pi(f)$. The algorithm, called embedded ParRep, circumvents the disadvantages of CTMC ParRep discussed above. As mentioned in the previous section, CTMC ParRep can be slow due to the randomness of the holding times. In the worst case, one has to wait until all replicas leave W in order to determine the first exit time T^*. To circumvent this issue we propose an algorithm based on the embedded chain in which the parallel stage terminates as soon as one of the replicas leaves W.

Before we describe embedded ParRep, we introduce some notation. Throughout, $X_n^1, \dots, X_n^R$ will be independent processes with the same law as X_n and with initial distributions supported in W. Moreover, we consider $X_n^1, \dots, X_n^R$ as the embedded chains of $X^1(t), \dots, X^r(t)$ defined above, and let $\Delta\tau_n^1, \dots, \Delta\tau_n^R$ be the corresponding holding times. Recall that the first exit time of X_n from W is

$$N = \inf\{n > 0 : X_n \notin W\}.$$

For $r = 1, \dots, R$, we define the first exit time of X_n^r from W by

$$N^r = \min\{n \in \mathbb{N}; X_n^r \notin W\}$$

and the smallest among them by

$$N^* = \min\{N^r; r = 1, \dots, R\}.$$

Note that it is possible that more than one replica leave W for the first time after N^* transitions. We denote by K the smallest index among these escaped replicas. That is,

$$K = \min\{r = 1, \dots, R; X_{N^*}^r \notin W\}.$$

It is clear from the above definition that $N^K = N^*$. Of course N, N^r, N^*, and K depend on W, but we do not make this explicit.

Here and below we write $\mathbb{E}^{\mu^R}$ for expectation of $(X_n^1, \dots, X_n^R)$ starting at μ^R, where

$$\mu^R(x_1, \dots, x_R) = \prod_{r=1}^{R} \mu(x_r), \qquad x_1, \dots, x_R \in W.$$

We begin by reproducing from [2] Theorems 4.1 and 4.2 below, with proofs for completeness.

THEOREM 4.1. *Suppose* $(X_n^1, \dots, X_n^R)$ *has initial distribution* μ^R. *Then* $R(N^K - 1) + K$ *has the same distribution as* N^1.

Proof. Note that for any $n \geq 0$ and $k = 1, \dots, R$, the event $\{N^K = n, K = k\}$ is equivalent to the event $\{N^1 > n, \dots, N^{k-1} > n, N^k = n, N^{k+1} > n-1, \dots, N^R > n-1\}$. Since $X_n^1, \dots, X_n^R$ are iid and N^1 is geometrically distributed with rate $p = \mathbb{P}^{\mu^R}(N^1 > 1)$ (see Theorem 2.5),

$$\mathbb{P}^{\mu^R}(N^K = n, K = k) = (1-p)^{n(k-1)}(1-p)^{n-1}p(1-p)^{(n-1)(k-1)} = (1-p)^{R(n-1)+k-1}p.$$

That is, $R(N^K - 1) + K$ has geometric distribution with rate p. ∎

THEOREM 4.2. *Suppose* $(X_n^1, \dots, X_n^R)$ *has the initial distribution* μ^R. *Then* $X_{N^K}^K$ *is independent of* $R(N^K - 1) + K$ *and the distribution of* $(X_{N^K}^K, R(N^K - 1) + K)$ *is the same as that of* $(X_{N^1}^1, N^1)$.

Proof. We first prove that $X_{N^K}^K$ is independent of K. Since $X_n^R, \dots, X_n^R$ are iid and N^k is independent of $X_{N^k}^k$ for each k, then $X_{N^k}^k$ is independent of $N^1, \dots, N^R$. Note that $K \in \sigma(N^1, \dots, N^R)$, hence $X_{N^k}^k$ is independent of K for each k. Now observe that for any $A \subset E$,

$$\begin{aligned}\mathbb{P}^{\mu^R}(X_{N^K}^K \in A) &= \sum_{r=1}^{R} \mathbb{P}^{\mu^R}(X_{N^k}^k \in A, K = r) \\ &= \sum_{r=1}^{R} \mathbb{P}^{\mu^R}(X_{N^1}^1 \in A)\mathbb{P}^{\mu^R}(K = r) \\ &= \mathbb{P}^{\mu^R}(X_{N^1}^1 \in A),\end{aligned}$$

that is, $X_{N^K}^K$ and $X_{N^1}^1$ are equally distributed. This implies that $X_{N^K}^K$ is independent of K. To see the independence between $X_{N^K}^K$ and $R(N^K - 1) + K$, note that

$$\begin{aligned}\mathbb{P}^{\mu^R}(X_{N^K}^K \in A, N^K = n, K = r) &= \mathbb{P}^{\mu^R}(X_{N^r}^r \in A, N^r = n, K = r) \\ &= \mathbb{P}^{\mu^R}(X_{N^r}^r \in A, K = r | N^r = n)\mathbb{P}^{\mu^R}(N^r = n) \\ &= \mathbb{P}^{\mu^R}(X_{N^r}^r \in A | N^r = n)\mathbb{P}^{\mu^R}(N^r = n, K = r) \\ &= \mathbb{P}^{\mu^R}(X_{N^r}^r \in A)\mathbb{P}^{\mu^R}(N^r = n, K = r) \\ &= \mathbb{P}^{\mu^R}(X_{N^K}^K \in A)\mathbb{P}^{\mu^R}(N^K = n, K = r)\end{aligned}$$

for any measurable $A \subset E$, $n \in \mathbb{Z}^+$ and $r = 1, \dots, R$. Finally, Theorem 4.1 and the above analysis imply that $(X_{N^K}^K, R(N^K - 1) + K)$ and $(X_{N^1}^1, N^1)$ are equally distributed. ∎

Now we present the embedded ParRep algorithm in Algorithm 2. In this algorithm we will need user-chosen parameters n_c associated with each metastable set W. Roughly, these parameters correspond to the time for X_n to converge to the QSD in W.

The DTMC X_n and holding times $\Delta\tau_n$ are simulated by the stochastic simulation algorithm (SSA); see, for instance, [13], just as in the CTMC ParRep. If X_n^{par} remains in W for a sufficiently long time (i.e., time t_c), it is distributed nearly according to the QSD μ of X_n in W. See Theorem 2.3. This means that at the end of the decorrelation stage, X_n^{par} can be considered a sample of μ.

The aim of the dephasing stage is to prepare a sequence of iid initial states with distribution μ. Like the CTMC ParRep, rejection sampling can be used for the embedded ParRep as well. However, a more natural and efficient option for the embedded ParRep is a Fleming–Viot based sampling procedure [3, 11]. The procedure can be summarized as follows.

The R replicas $X_n^1, \dots, X_n^R$, starting in W, evolve until one or more of them leaves W. Then each replica that left W is restarted from the current state of another replica that is currently in W (usually chosen uniformly at random). The procedure stops after the replicas have evolved for $n = n_p$ time steps, where n_p is a parameter similar to n_c. (If all the replicas leave W at the same time, the procedure restarts from the beginning.) With this type of sampling, the number of time steps simulated for each

Algorithm 2 Embedded ParRep

1: Set a decorrelation threshold n_c for each metastable set W. Initialize the simulation time clock $N_{\text{sim}} = 0$ and the accumulated value $F(f)_{\text{sim}} = 0$. We will write X_n^{par} and Δ_n^{par} for a DTMC and holding time process following the law of the embedded chain and holding times of $X(t)$, respectively. A complete ParRep cycle consists of three stages.

2: **Decorrelation stage**: Starting at $n = N_{\text{sim}}$, evolve X_n^{par} and $\Delta\tau_n^{\text{par}}$ until X_n^{par} spends n_c consecutive time steps inside of the same metastable set W. That is, evolve X_n^{par} and $\Delta\tau_n^{\text{par}}$ from time $n = N_{\text{sim}}$ until time

$$N_{\text{corr}} = \inf\{n \geq N_{\text{sim}} + n_c - 1 : X_m^{\text{par}} \in W \text{ for } m \in \{n - n_c + 1, \ldots, n\} \text{ for some } W\}.$$

Then update

$$F(f)_{\text{sim}} = F(f)_{\text{sim}} + \sum_{n=N_{\text{sim}}}^{N_{\text{corr}}-1} f(X_n^{\text{par}})\Delta\tau_n^{\text{par}},$$

set $N_{\text{sim}} = N_{\text{corr}}$, and proceed to the dephasing stage.

3: **Dephasing stage**: Let W be such that $X_{N_{\text{sim}}}^{\text{par}} \in W$, that is, W is the metastable set from the end of the decorrelation stage. Generate R independent samples $x_1, \ldots, x_R$ from μ, the QSD of X_n in W. Then proceed to the parallel stage.

4: **Parallel stage**: Start R parallel processes $X_n^1, \ldots, X_n^R$ at $x_1, \ldots, x_R$, and evolve them and the corresponding holding times $\Delta\tau_n^1, \ldots, \Delta\tau_n^R$ from time $n = 0$ until time N^*. Then update

$$\begin{aligned} F(f)_{\text{sim}} &= F(f)_{\text{sim}} + \sum_{r=1}^{R} \sum_{k=0}^{N^*-2} f(X_k^r)\Delta\tau_k^r + \sum_{r=1}^{K} f(X_{N^*-1}^r)\Delta\tau_{N^*-1}^r, \\ N_{\text{sim}} &= N_{\text{sim}} + R(N^* - 1) + K, \end{aligned} \tag{14}$$

set $X_{N_{\text{sim}}}^{\text{par}} = X_{N^*}^K$, and return to the decorrelation stage.

5: The algorithm is stopped when N_{sim} reaches some user-chosen time N_{end}. The stationary average $\pi(f)$ is estimated as

$$\pi(f) \approx F(f)_{\text{sim}}/F(1)_{\text{sim}}.$$

replica in the dephasing step is the same. In particular, if the R parallel processors are synchronous (i.e., if each processor takes the same wall clock time to simulate one time step), then each processor finishes the dephasing step at the same wall clock time. We comment that the Fleming–Viot technique can be used to estimate the decorrelation and dephasing thresholds as well when they are difficult to choose a priori [3].

The acceleration of the embedded ParRep comes from the parallel stage. Figure 2 shows the diagram for the parallel stage with R replicas. Roughly, we only have to wait N^* time steps instead of N to observe an exit from W. The theoretical wall clock time speedup can be approximately a factor of R. See Theorem 4.1 below. Similar to CTMC ParRep, the parallel step does not require metastability for this time speedup, but if W is not metastable, then the dephasing step will not be efficient. See

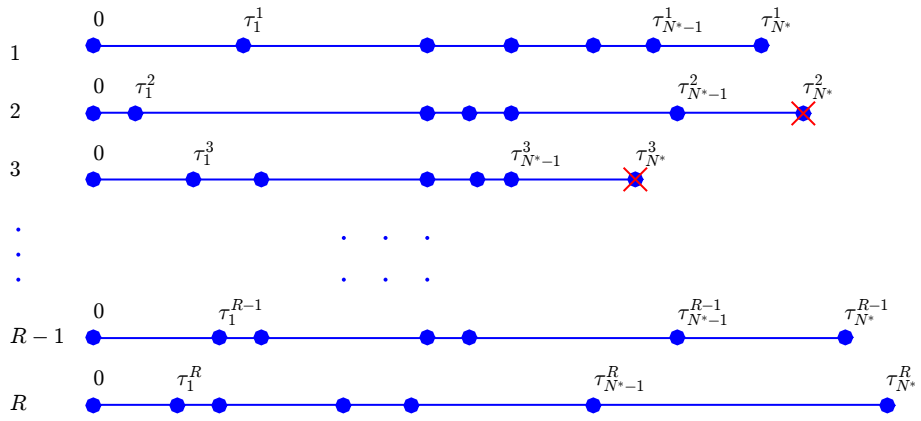

FIG. 2. *The diagram for one parallel stage of the embedded ParRep algorithm with R replicas. Each blue dot represents an exit event along the time line. Both replica 2 and 3 leave W after $N^* = 6$ transitions (the blue dot with the red "x"), in which case $K = 2$. (Figure is in color online.)*

the remarks below Algorithm 1.

Similar to the CTMC ParRep, each parallel stage of the embedded ParRep has a consistent averaged contribution to the stationary average. Suppose that $x_1, \dots, x_R$ are iid samples from μ.

1. The joint law of $(X_{N^K}^K, R(N^K - 1) + K)$ is the same as that of $(X_{N^1}^1, N^1)$. That is, the joint distribution of the first exit time and the exit state for each parallel stage is independent of the number of replicas.
2. The expected value of

$$\sum_{r=1}^{R} \sum_{k=0}^{N^*-2} f(X_k^r)\Delta\tau_k^r + \sum_{r=1}^{K} f(X_{N^*-1}^r)\Delta\tau_{N^*-1}^r$$

is the same as that of

$$\sum_{n=0}^{N-1} f(X_n^1)\Delta\tau_n^1.$$

Hence the expected contribution to $F(f)_{\text{sim}}$ from each parallel stage is independent of the number of replicas. See Theorem 4.3 below.

See Theorems 4.2 and 4.3 for proofs of these statements.

We expect that embedded ParRep is superior to the CTMC ParRep for the following two reasons. First, consider the parallel stages of both algorithms. In the CTMC ParRep, observing the first exit event in the parallel stage is not sufficient to determine T^*. But in embedded ParRep, once any replica leaves W, we know N^*. Thus the embedded ParRep parallel step terminates once any of the replicas leaves W. For this reason we expect the parallel stage of the embedded ParRep to be significantly faster than that of the CTMC ParRep. Second, consider the dephasing stage. For the embedded ParRep, Fleming–Viot sampling is a natural technique because if the processors are synchronous, then they all finish the dephasing stage at the same wall clock time, and only the current states of each processor are needed at each time step to decide where to restart replicas which left W. For asynchronous processors,

one can simply implement a polling time. This is not true, however, for Fleming–Viot sampling with the CTMC ParRep. Indeed, to implement Fleming–Viot sampling with the CTMC ParRep, one would have to store the histories of every replica, and the replicas would finish at potentially very different wall clock times. The rejection method can be slow for both algorithms, particularly when the metastability is weak or when the number of replicas is large.

4.2. Error analysis of the embedded ParRep. Now we are able to show that if the dephasing sampling is exact, then the expected contribution to $F(f)_{\text{sim}}$ from the parallel stage is exact.

THEOREM 4.3. *Suppose in the dephasing step* $(x_1, \dots, x_R) \sim \mu^R$. *Then the expected contribution to* $F(f)_{sim}$ *from the parallel stage of Algorithm* 2 *is the same for every number of replicas.*

$$\mathbb{E}^{\mu^R}\left[\sum_{r=1}^{R}\sum_{k=0}^{N^*-2} f(X_k^r)\Delta\tau_k^r + \sum_{r=1}^{K} f(X_{N^*-1}^r)\Delta\tau_{N^*-1}^r\right] = \mathbb{E}^{\mu}\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right] = \mu(fq^{-1})\mathbb{E}^{\mu}[N],$$

where q *is the function as defined in section* 2.1.

Proof. We first rewrite

$$\begin{aligned}(15)\qquad &\sum_{r=1}^{R}\sum_{k=0}^{N^*-2} f(X_k^r)\Delta\tau_k^r + \sum_{r=1}^{K} f(X_{N^*-1}^r)\Delta\tau_{N^*-1}^r \\ &= \sum_{r=1}^{R}\sum_{i=0}^{N^*-1} f(X_i^r)\Delta\tau_i^r - \sum_{r=K+1}^{R} f(X_{N^*-1}^r)\Delta\tau_{N^*-1}^r.\end{aligned}$$

For the first part, we condition N^* and obtain

$$\mathbb{E}^{\mu^R}\left[\sum_{r=1}^{R}\sum_{i=0}^{N^*-1} f(X_i^r)\Delta\tau_i^r\right] = \sum_{r=1}^{R}\sum_{n=1}^{\infty}\sum_{i=0}^{n-1}\mathbb{E}^{\mu^R}\left[f(X_i^r)\Delta\tau_i^r\mathbb{I}_{N^*=n}\right].$$

Interchanging the iterated summations leads to

$$\sum_{r=1}^{R}\sum_{n=1}^{\infty}\sum_{i=0}^{n-1}\mathbb{E}^{\mu^R}\left[f(X_i^r)\Delta\tau_i^r\mathbb{I}_{N^*=n}\right] = \sum_{r=1}^{R}\sum_{i=0}^{\infty}\mathbb{E}^{\mu^R}\left[f(X_i^r)\mathbb{I}_{N^*>i}\Delta\tau_i^r\right].$$

Notice $N^* > i$ is equivalent to $N^1 > i, \dots, N^R > i$ and $\Delta\tau_i^r$ is independent of N^s for $s \neq r$. Thus

$$\begin{aligned}&\sum_{r=1}^{R}\sum_{i=0}^{\infty}\mathbb{E}^{\mu^R}\left[f(X_i^r)\Delta\tau_i^r|N^*>i\right]\mathbb{P}^{\mu^R}(N^*>i) \\ =&\sum_{r=1}^{R}\sum_{i=0}^{\infty}\mathbb{E}^{\mu^R}\left[f(X_i^r)\Delta\tau_i^r|N^r>i\right]\mathbb{P}^{\mu^R}(N^*>i).\end{aligned}$$

Now by Lemma 2.1 and the definition of the QSD,

$$\begin{aligned}\mathbb{E}^{\mu}\left[f(X_i^r)\Delta\tau_i^r|N^r>i\right] &= \mathbb{E}^{\mu}\left[\mathbb{E}^{\mu}\left[f(X_i^r)\Delta\tau_i^r|\{X_n^r\}_{n=0,1,\dots}\right]|N^r>i\right] \\ &= \mathbb{E}^{\mu}\left[f(X_i^r)\mathbb{E}^{\mu}\left[\Delta\tau_i^r|\{X_n^r\}_{n=0,1,\dots}\right]|N^r>i\right] \\ &= \mathbb{E}^{\mu}\left[f(X_i^r)q(X_i^r)^{-1}\big|N^r>i\right] = \mu(fq^{-1}).\end{aligned}$$

Combining the last four equations gives

$$\mathbb{E}^{\mu^R}\left[\sum_{r=1}^{R}\sum_{i=0}^{N^*-1} f(X_i^r)\Delta\tau_i^r\right] = \mu(fq^{-1})R\mathbb{E}^{\mu^R}[N^*]. \tag{16}$$

A similar argument can be applied to the second term on the right-hand side of (15). First we condition N^* and K simultaneously such that

$$\begin{aligned}&\mathbb{E}^{\mu^R}\left[\sum_{r=K+1}^{R} f(X_{N^*-1}^r)\Delta\tau_{N^*-1}^r\right]\\ =&\sum_{n=1}^{\infty}\sum_{r=1}^{R}\sum_{r=k+1}^{R}\mathbb{E}^{\mu^R}\left[f(X_{n-1}^r)\Delta\tau_{n-1}^r|N^*=n,K=k\right]\mathbb{P}^{\mu^R}(N^*=n,K=k).\end{aligned}$$

Interchanging the second and third summations the right-hand side equals

$$\sum_{n=1}^{\infty}\sum_{r=2}^{R}\sum_{k=1}^{r-1}\mathbb{E}^{\mu^R}\left[f(X_{n-1}^r)\Delta\tau_{n-1}^r|N^*=n,K=k\right]\mathbb{P}^{\mu^R}(N^*=n,K=k).$$

Recall that

$$N^*=n,\,K=k \iff N^1>n,\ldots,N^{k-1}>n,\,N^k=n,\,N^{k+1}>n-1,\ldots,N^R>n-1.$$

Thus, using independence of $X_n^1,\ldots,X_n^R$ and the definition of the QSD,

$$\begin{aligned}&\sum_{n=1}^{\infty}\sum_{r=2}^{R}\sum_{k=1}^{r-1}\mathbb{E}^{\mu^R}\left[f(X_{n-1}^r)\Delta\tau_{n-1}^r|N^*=n,K=k\right]\mathbb{P}^{\mu^R}(N^*=n,K=k)\\ =&\sum_{n=1}^{\infty}\sum_{r=2}^{R}\sum_{k=1}^{r-1}\mathbb{E}^{\mu}\left[f(X_{n-1}^r)\Delta\tau_{n-1}^r|N^r>n-1\right]\mathbb{P}^{\mu^R}(N^*=n,K=k)\\ =&\mu(fq^{-1})\sum_{n=1}^{\infty}\sum_{r=2}^{R}\sum_{k=1}^{r-1}\mathbb{P}^{\mu^R}(N^*=n,K=k)\\ =&\mu(fq^{-1})(R-\mathbb{E}^{\mu^R}[K]).\end{aligned}$$

Combining the last three equations leads to

$$\mathbb{E}^{\mu}\left[\sum_{r=K+1}^{R} f(X_{N^*-1}^r)\Delta\tau_{N^*-1}^r\right] = \mu(fq^{-1})(R-\mathbb{E}^{\mu^R}[K]). \tag{17}$$

Subtracting (17) from (16), we have

$$\mathbb{E}^{\mu^R}\left[\sum_{r=1}^{R}\sum_{i=0}^{N^*-1} f(X_i^r)\Delta\tau_i^r - \sum_{r=K+1}^{R} f(X_{N^*-1}^r)\Delta\tau_{N^*-1}^r\right] = \mu(fq^{-1})\mathbb{E}^{\mu^R}\left[R(N^*-1)+K\right].$$

Now the result follows since

$$\mu(fq^{-1})\mathbb{E}^{\mu^R}\left[R(N^*-1)+K\right] = \mu(fq^{-1})\mathbb{E}^{\mu}[N]$$

by Theorem 4.2. In particular, when $R=1$ we have $N^*=N$ and $K=1$, and thus

$$\mathbb{E}^{\mu}\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right] = \mu(fq^{-1})\mathbb{E}^{\mu}[N].$$

∎

We now prove an analog of Theorem 3.3 for the embedded ParRep. Recall we have assumed convergence of $\|\mu_{n_c} - \mu\|_{\mathrm{TV}} \to 0$ as $n_c \to \infty$, for every starting point $x \in E$. See, for instance, Theorem 2.3 for conditions guaranteeing this convergence.

THEOREM 4.4. *Consider the embedded ParRep starting at $x \in W$ in the decorrelation stage. Assume the dephasing stage sampling is exact, that is, $(x_1, \dots, x_R) \sim \mu^R$. Consider the expected contribution to $F(f)_{sim}$ up until the first time the simulation leaves W (either in the decorrelation stage or in the parallel stage):*

$$\Delta F(f)_{\mathrm{sim}} \triangleq \mathbb{E}^x\left[\sum_{n=0}^{n_c \wedge N-1} f(X_n)\Delta\tau_n\right] + \mathbb{E}^{x,\mu^R}\left[\mathbb{1}_{N>n_c}\sum_{r=1}^{R}\sum_{k=0}^{N^*-2} f(X_k^r)\Delta\tau_k^r + \mathbb{1}_{N>n_c}\sum_{r=1}^{K} f(X^r_{N^*-1})\Delta\tau^r_{N^*-1}\right],$$

where $\mathbb{E}^{x,\mu^R}$ denotes expectation for $(X_n, X_n^1, \dots, X_n^R)$ with X_n starting at x and the replicas $(X_n^1, \dots, X_n^R)$ starting at the initial distribution μ^R. The error compared to a direct (serial) simulation satisfies the bound

$$\left|\mathbb{E}^x\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right] - \Delta F(f)_{sim}\right| \le \|f\|_\infty \sup_{x\in W}\mathbb{E}^x[T]\,\|\mu_{n_c} - \mu\|_{\mathrm{TV}}. \tag{18}$$

Proof. The proof is similar to that for the CTMC ParRep,

$$\begin{aligned}
&\left|\mathbb{E}^x\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right] - \Delta F(f)_{\mathrm{sim}}\right| \\
&= \left|\mathbb{E}^x\left[\sum_{n=n_c\wedge N}^{N-1} f(X_n)\Delta\tau_n\right] - \mathbb{E}^{x,\mu^R}\left[\mathbb{1}_{N>n_c}\sum_{r=1}^{R}\sum_{k=0}^{N^*-2} f(X_k^r)\Delta\tau_k^r + \mathbb{1}_{N>n_c}\sum_{r=1}^{K} f(X^r_{N^*-1})\Delta\tau^r_{N^*-1}\right]\right| \\
&\le \left|\mathbb{E}^x\left[\sum_{n=n_c}^{N-1} f(X_n)\Delta\tau_n \,\middle|\, N > n_c\right] - \mathbb{E}^{\mu^R}\left[\sum_{r=1}^{R}\sum_{k=0}^{N^*-2} f(X_k^r)\Delta\tau_k^r + \sum_{r=1}^{K} f(X^r_{N^*-1})\Delta\tau^r_{N^*-1}\right]\right|.
\end{aligned}$$

By the Markov property

$$\mathbb{E}^x\left[\sum_{n=n_c}^{N-1} f(X_n)\Delta\tau_n \,\middle|\, N > n_c\right] = \mathbb{E}^{\mu_{n_c}}\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right].$$

Owing to Theorem 4.3,

$$\mathbb{E}^{\mu^R}\left[\sum_{r=1}^{R}\sum_{k=0}^{N^*-2} f(X_k^r)\Delta\tau_k^r + \sum_{r=1}^{K} f(X^r_{N^*-1})\Delta\tau^r_{N^*-1}\right] = \mathbb{E}^{\mu}\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right].$$

Therefore,

$$\begin{aligned}
&\left|\mathbb{E}^x\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right] - \Delta F(f)_{\text{sim}}\right| \\
&\leq \left|\mathbb{E}^{\mu_{n_c}}\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right] - \mathbb{E}^{\mu}\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right]\right| \\
&= \left|\sum_{x\in W}\mathbb{E}^x\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right]\mu_{n_c}(x) - \sum_{x\in W}\mathbb{E}^x\left[\sum_{n=0}^{N-1} f(X_n)\Delta\tau_n\right]\mu(x)\right| \\
&\leq \|f\|_\infty \sup_{x\in W}\mathbb{E}^x[T]\|\mu_{n_c} - \mu\|_{TV}
\end{aligned}$$

with the last equation coming from the fact that $\mathbb{E}^x[\sum_{n=0}^{N-1}\Delta\tau_n] = \mathbb{E}^x[T]$. ∎

5. Numerical experiments. We present two numerical examples from the stochastic reaction networks in order to demonstrate the consistency and efficiency of the ParRep algorithms.

5.1. Reaction networks with linear propensity. We consider the following stochastic reaction network:

$$\varnothing \longrightarrow A \rightleftharpoons B \longrightarrow C \longrightarrow \varnothing \tag{19}$$

taken from [7], where A, B, and C represent reacting species. The time evolution of the population (the number of species) in the reaction network is commonly modeled as a CTMC $X(t) = (X_1(t), X_2(t), X_3(t))$ with state space $E \subset \mathbb{Z}_+^3$. The jump rate of each reaction is governed by the propensity function (intensity) $\lambda_j(x), j = 1, \ldots, 5$, such that for all $t > 0$,

$$\lambda_j(x) = \lim_{h\to 0}\frac{\mathbb{P}(X(t+h) = x + \eta_j | X(t) = x)}{h},$$

where η_j is the state change vector associated with the jth reaction. We list the reactions and their corresponding propensity functions and state change vectors in Table 1.

TABLE 1
Reactions, propensity functions, and state change vectors.

Reaction	Propensity function	State change vector
$\varnothing \longrightarrow A$	$\lambda_1(x) = c_1$	$\eta_1 = (1, 0, 0)$
$A \longrightarrow B$	$\lambda_2(x) = c_2 x_1$	$\eta_2 = (-1, 1, 0)$
$B \longrightarrow A$	$\lambda_3(x) = c_3 x_2$	$\eta_3 = (1, -1, 0)$
$B \longrightarrow C$	$\lambda_4(x) = c_4 x_2$	$\eta_4 = (0, -1, 1)$
$C \longrightarrow \varnothing$	$\lambda_5(x) = c_5 x_3$	$\eta_5 = (0, 0, -1)$

In this numerical experiment, we take the initial state $x_0 = (5, 10, 10)$ and the rate constants

$$(c_1, c_2, c_3, c_4, c_5) = (0.1, 100, 100, 0.01, 0.01).$$

With this choice of parameters the timescale separation is about $\epsilon = 10^{-4}$ and hence the process $X(t)$ demonstrates metastability. The reactions $A \to B$ and $B \to A$ occur with a much higher probability than the other reactions and hence we call $A \to B$ and

$B \to A$ fast reactions and the other reactions slow reactions. The occurrence of slow reactions is a rare event. We define the observables $f_1(x) = x_1 + x_2$ and $f_2(x) = x_3$, the collection of sets $\{W_{m,n}\}_{m,n\in\mathbb{Z}_+}$ with

$$W_{m,n} = \{x \in E : f_1(x) = m, f_2(x) = n\}$$

form a full decomposition of the state space E. Note that both the total population of species A and B (i.e., $f_1(X(t))$) and the population of species C (i.e., $f_2(X(t))$) remain constant until one of the slow reactions occurs. Hence the typical sojourn time for $X(t)$ in each $W_{m,n}$ is very long comparing to the transition time between any two states that are in $W_{m,n}$. In this case, we say $X(t)$ is metastable in $W_{m,n}$. For example, with the initial population $x_0 = (1, 1, 0)$, the states $(1, 1, 0), (2, 0, 0)$, and $(0, 2, 0)$ form a metastable set since the fast reactions $A \to B$ and $B \to A$ occur with a significantly higher probability than slow reactions and only the occurrence of the slow reactions can allow the process to move from the metastable set to another metastable set. Note that both observables f_1 and f_2 defined above are invariant in each metastable set, we call them slow observables. In general, an observable f is called a slow observable if it is invariant in each metastable set $W_{m,n}$, i.e., there is a constant $C(m, n)$ such that $f(x) = C(m, n)$ for each $x \in W_{m,n}$. An observable is called a fast observable if it is not slow (e.g., $f(x) = x_1$).

This kind of two-scale problem arises in many fields other than the stochastic reaction networks, such as the queuing theory and population dynamics. Estimation of the distributions of two-scale processes can be computationally prohibitive due to the insufficient sampling of the rare events. Therefore, it is desirable to apply the two ParRep algorithms proposed in this paper to accelerate the long time simulation and estimate the stationary distribution.

We apply both the CTMC ParRep and the embedded ParRep to estimate the stationary averages of the slow observables f_1 and f_2. The stationary distribution of the fast observable $f_3(x) = x_1$ is also computed using the embedded ParRep. On the other hand, for the reaction network (19) under consideration, one can calculate the stationary distribution analytically since it only involves mono-molecular reactions. In fact, it can be shown that the stationary distribution is a multivariate Poisson distribution [7], that is,

$$\pi(x_1, x_2, x_3) = \frac{\bar{\lambda}_1^{x_1}\bar{\lambda}_2^{x_2}\bar{\lambda}_3^{x_3}}{x_1!x_2!x_3!} e^{-(\bar{\lambda}_1+\bar{\lambda}_2+\bar{\lambda}_3)}, \tag{20}$$

where

$$\bar{\lambda}_1 = \frac{c_1(c_3 + c_4)}{c_2c_4}, \quad \bar{\lambda}_2 = \frac{c_1}{c_4}, \quad \bar{\lambda}_3 = \frac{c_1}{c_5}.$$

Hence the exact stationary averages of the slow observables f_1 and f_2 are $\pi(f_1) = 20.001$ and $\pi(f_2) = 10$ and the exact stationary averages of the fast observable $f_3(x) = x_1$ is 10.001. We use this exact result to compare with our result from numerical simulation.

Our simulations compare the CTMC ParRep and the embedded ParRep with the SSA [13]. In Figure 3, we demonstrate the estimation of $\pi(f_1)$ using the CTMC ParRep and the embedded ParRep with various numbers of replicas ($R = 10, 20, \ldots, 100$) and with SSA ($R = 1$). Similarly, Figure 4 shows the estimation of $\pi(f_2)$. Note that only the embedded ParRep is used to compute the stationary average of the fast variable $f(x) = x_1$ since the CTMC ParRep is not efficient for fast observables as we

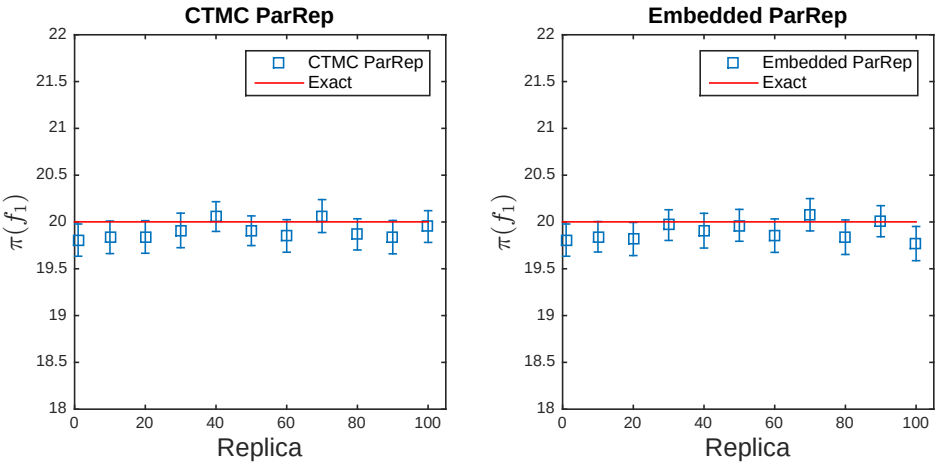

FIG. 3. *The stationary average of the slow observable* $f_1(x) = x_1 + x_2$ *computed with the CTMC ParRep (left) and with the embedded ParRep (right). The user-specified terminal time is* $T_{\text{end}} = 10^4$ *in the simulation.*

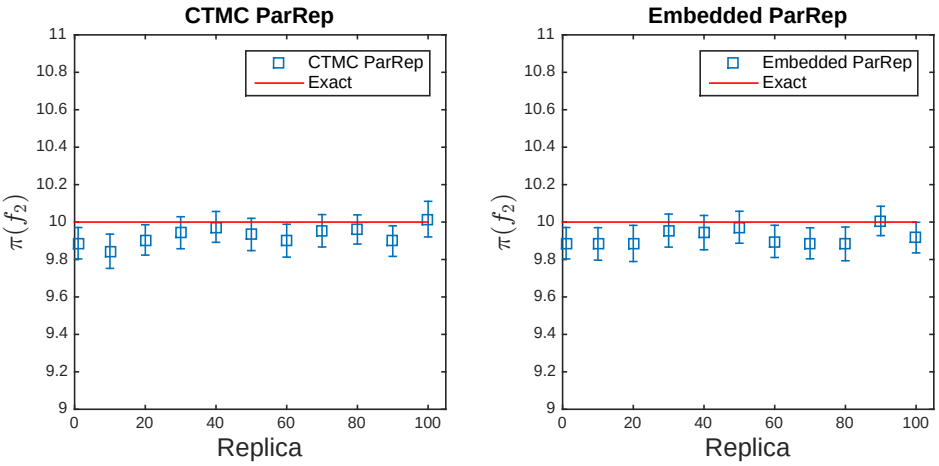

FIG. 4. *The stationary average of the slow observable* $f_2(x) = x_3$ *computed with the CTMC ParRep (left) and with the embedded ParRep (right). The user-specified terminal time is* $T_{\text{end}} = 10^4$ *in the simulation.*

commented at the end of section 3.1. Currently, the rejection sampling is used for dephasing and the decorrelation and dephasing thresholds are taken to be $t_c = t_p = 0.01$ for the CTMC ParRep and $n_c = n_p = 15$ steps for the embedded ParRep. In Figure 5, the estimation for the fast observable and speedup are shown. It can be seen that with ten replicas, the speedup factor is about 4.5 for the CTMC ParRep and 5.5 for the embedded ParRep. When the number of replicas increases, the embedded ParRep becomes much more efficient than the CTMC ParRep. However, even the embedded ParRep is far away from the linear speedup (with 100 replicas, about 27 times faster than SSA). This sublinear speedup comes from the fact that when the number of replicas is large, the acceleration is offset by the inefficient rejection sampling based dephasing procedure. We expect that the embedded ParRep would be more efficient if the Fleming–Viot particle processes are used for dephasing.

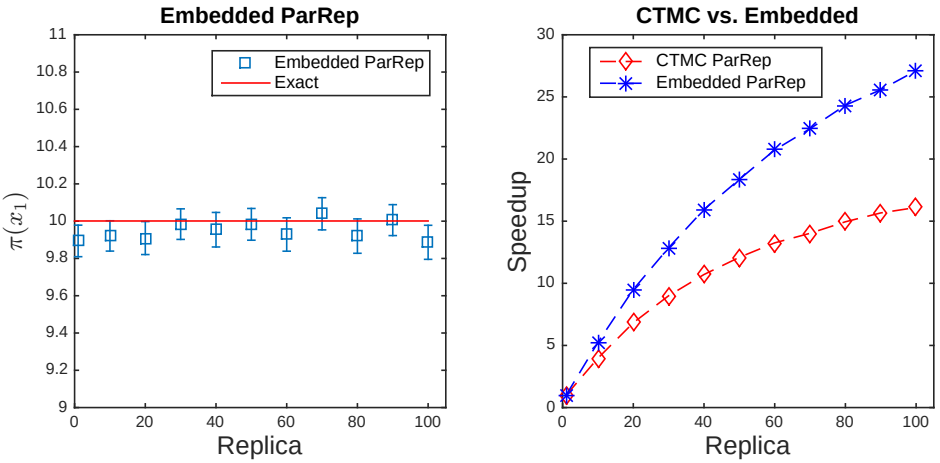

FIG. 5. *The stationary average of the fast observable $f_3(x) = x_1$ computed with the embedded ParRep (left) and the speedup comparison between the CTMC ParRep and the embedded ParRep (right). The user-specified terminal time is $T_{\text{end}} = 10^4$ in the simulation.*

5.2. Reaction networks with nonlinear propensity. In the second example, we focus on the following network from [24]:

$$S_1 \rightleftharpoons S_2, \quad S_1 \rightleftharpoons S_3, \quad 2S_2 + S_3 \rightleftharpoons 3S_4. \tag{21}$$

The propensity function and state change vector associated with each reaction is shown in Table 2. Note that by the law of mass action, the reactions $2S_2 + S_3 \rightleftharpoons 3S_4$ have nonlinear propensity functions.

TABLE 2
Reactions, propensity functions, and state change vectors.

Reaction	Propensity function	State change vector
$S_1 \longrightarrow S_2$	$\lambda_1(x) = c_1 x_1$	$\eta_1 = (-1, 1, 0, 0)$
$S_2 \longrightarrow S_1$	$\lambda_2(x) = c_2 x_2$	$\eta_2 = (1, -1, 0, 0)$
$S_1 \longrightarrow S_3$	$\lambda_3(x) = c_3 x_1$	$\eta_3 = (-1, 0, 1, 0)$
$S_3 \longrightarrow S_1$	$\lambda_4(x) = c_4 x_3$	$\eta_4 = (1, 0, -1, 0)$
$2S_2 + S_3 \longrightarrow 3S_4$	$\lambda_5(x) = c_5 x_2(x_2 - 1)x_3$	$\eta_5 = (0, -2, -1, 3)$
$3S_4 \longrightarrow 2S_2 + S_3$	$\lambda_6(x) = c_6 x_3(x_3 - 1)(x_3 - 2)$	$\eta_6 = (0, 2, 1, -3)$

Throughout this example, we choose the initial state $x_0 = (3, 30, 30, 30)$ and the reaction rate constants

$$(c_1, c_2, c_3, c_4, c_5, c_6) = (0.1, 0.1, 0.1, 0.1, 2, 2)$$

for simulation. In this reaction network, $2S_2 + S_3 \rightleftharpoons 3S_4$ are fast reactions due to the cubic form of the propensity functions. The rest of the reactions are considered as slow reactions. We plot time evolution of the total propensity of reactions 5 and 6 versus the total propensity of reactions 1 to 4 in Figure 6. The timescale separation is more than $\epsilon = 10^{-4}$ as shown in the plot. The slow observables $f_1(x) = x_2 + x_3 + x_4$ and $f_2(x) = x_1$ remain unchanged until any of the slow reactions occur. Therefore, the state space E can be partitioned as a disjoint union of metastable sets in terms of slow observables f_1 and f_2. That is, $E = \cup W_{m,n}$, where $W_{m,n} = \{x \in E : f_1(x) = m, f_2(x) = n\}$.

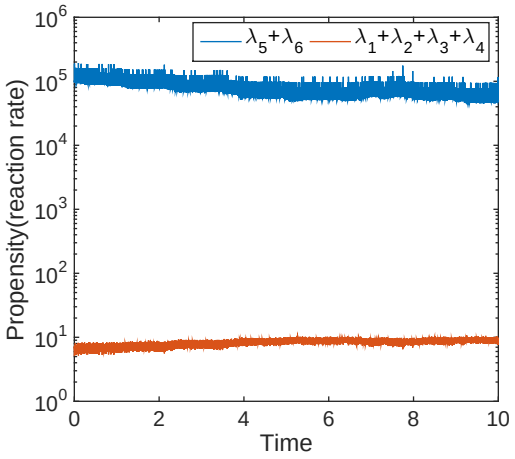

FIG. 6. *The timescale separation between fast reactions and slow reactions. The blue curve above and the red curve below show the time evolution of* $\lambda_5+\lambda_6$ *and* $\lambda_1+\lambda_2+\lambda_3+\lambda_4$*, respectively. It can be seen that the total propensity (reaction rate) of the last two reactions are more than* 10^4 *larger than that of the first four reactions. (Figure is in color online.)*

In the numerical simulation of (21), we apply the embedded ParRep with rejection sampling and with Fleming–Viot sampling, respectively. We are interested in the stationary average of the fast variable $\pi(x_4)$. The user-specified terminal time is chosen to be $T_{\text{end}} = 10^3$, which is large enough for the system to be well into the stationary dynamics. Figure 7 shows the estimated result (with confidence interval) for $\pi(x_4)$ with the rejection sampling based embedded ParRep (left) and the Fleming–Viot sampling based embedded ParRep (right), each with different decorrelation and dephasing thresholds. The corresponding speedup factor is shown in Figure 8, where the left plot shows the speedup for $n_c = n_p = 20$ and the right plot shows the speedup for $n_c = n_p = 60$. It can be seen that when the decorrelation and dephasing thresholds are small (i.e., 20), there is no performance enhancement (as shown in the left plot) when the rejection sampling is replaced by the Fleming–Viot sampling. This is consistent with our expectation that most of the replicas finish the dephasing stage after 20 transitions (that is, no replicas escape the metastable set in 20 transitions) and hence the Fleming–Viot sampling is not needed to improve the performance. However, when the thresholds are increased to 60, the Fleming–Viot sampling based embedded ParRep outperforms the rejection sampling based embedded ParRep especially when the number of replica is large, as shown in the right plot. We expect that the Fleming–Viot sampling based ParRep would be more advantageous than the rejection sampling based ParRep when large n_c and n_p are needed, e.g., when the time scale separation is very large (say $\epsilon = 10^{-10}$).

Finally, we comment that in many cases of stochastic reaction network models the timescales of the dynamics could change over time. For instance, the last two reactions are slow if we choose $(100, 3, 3, 3)$ as the initial state in this numerical example. However, when the counts of S_2 and S_3 increase, the last two reactions become fast. If we still define the last two reactions as the slow reactions, then the parallel stage will not be activated in which case the ParRep becomes equivalent to SSA. A possible remedy for this issue is to use dynamic partition of slow and fast reactions. See [24] for a detailed discussion. We will deal with this issue in a separate work [23].

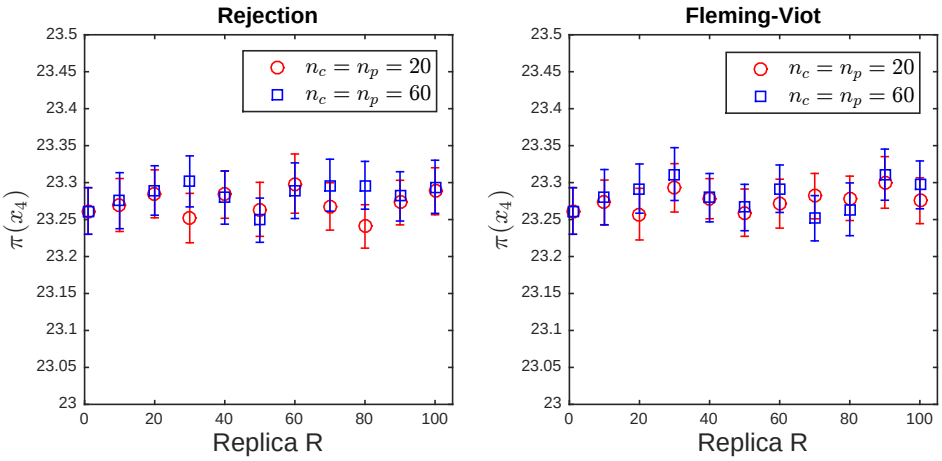

FIG. 7. *Stationary average (with confidence interval) of the fast observable x_4 computed with 20 decorrelation and dephasing steps and 60 decorrelation and dephasing steps, respectively. Both the rejection sampling and the Fleming–Viot sampling are used for the dephasing stage.*

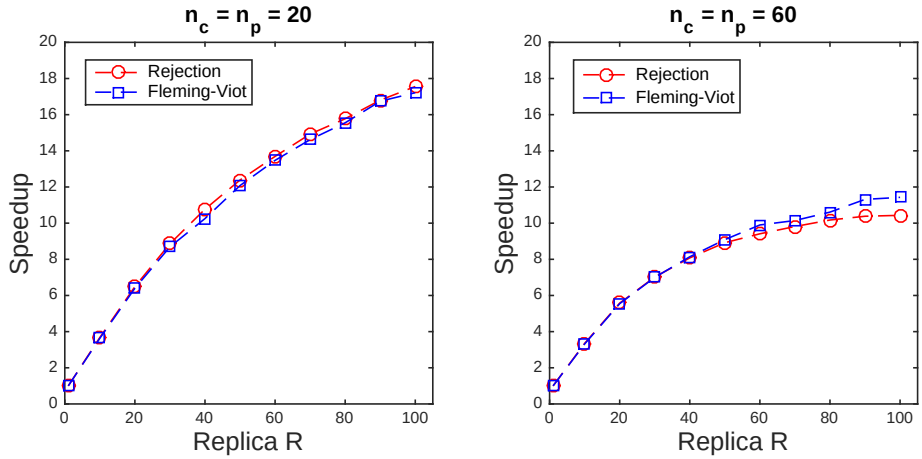

FIG. 8. *Speedup of the ParRep with the rejection sampling based dephasing and the ParRep with Fleming–Viot sampling based dephasing.*

6. Conclusions. This paper proposes a new method for simulating metastable CTMCs and estimating its stationary distribution with an application to stochastic reaction network models. The method is based on the parallel replica dynamics which first appeared in [22]. The ParRep method proposed here does not require the reversibility (detailed balance) of the simulated Markov chain, which is the necessary assumption for most accelerated algorithms for metastable dynamics simulation. This makes the ParRep particularly well suited for a stochastic reaction network model where the reversibility is not satisfied in general.

To accelerate the estimation of stationary distribution of a metastable CTMC, our method introduces a source of error: we sample an approximation of the QSD of the metastable set in each decorrelation and dephasing stage. However, our error analysis shows that on average the error from each ParRep cycle decays exponentially assuming the dephasing stage sampling is exact. Moreover, our numerical examples

also suggest the consistency of the ParRep method. The global error analysis for ParRep (i.e., the error accumulated over the entire simulation) is much more involved and will be the focus of our future work.

The mathematical theory underlying the ParRep method predicts that we could achieve approximately linear speedup in terms of the number of replicas. However, due to the computation in the decorrelation and dephasing stages, the acceleration achieved in practical implementations is sublinear. Nevertheless, we observe a considerable performance enhancement in presented numerical examples. We believe further speedup is possible with a better parallel implementation of the algorithm on massively parallel clusters.

In the numerical examples considered in this paper, we define the metastable sets in terms of the slow observable and assume that the partition of fast and slow reactions are fixed with time. However, it is quite common that the timescales of the dynamics can change over time in many cases, especially in stochastic reaction model. In many models the separation of time scales can change the timescales over the course of system's evolution. For example, such a situation also occurs in stochastic reaction networks with a multimodal stationary distribution. In such a case the partition of the fast and slow reactions changes when the process leaves from the neighborhood of a current stable stationary point and move to the neighborhood of another stable stationary point. In this case, a different strategy (rather than fast/slow reactions) can be used to define the metastable sets. The ParRep method for dynamics with bistability is discussed in [23].

The algorithms developed in this paper assume that the underlying processors are synchronous. However, we believe both the CTMC ParRep and the embedded ParRep can be implemented in asynchronous architectures as well. In particular, the idea for handling asynchronous processors discussed in [15] (section 3) can, in principle, be applied to the embedded ParRep as well. We will focus on formalizing these synchronization ideas in our future work.

Acknowledgment. We would like to thank the anonymous referees for the comments that helped improve the manuscript.

REFERENCES

[1] D. Aristoff, *The parallel replica method for computing equilibrium averages of Markov chains*, Monte Carlo Methods Appl., 21 (2015), pp. 255–273.

[2] D. Aristoff, T. Lelièvre, and G. Simpson, *The parallel replica method for simulating long trajectories of Markov chains*, Appl. Math. Res. Express. AMRX, 2014 (2014), pp. 332–352.

[3] A. Binder, T. Lelièvre, and G. Simpson, *A generalized parallel replica dynamics*, J. Comput. Phys., 284 (2015), pp. 595–616.

[4] J. Blanchet, P. Glynn, and S. Zheng, *Theoretical Analysis of a Stochastic Approximation Approach for Computing Quasi-stationary Distributions*, preprint, https://arxiv.org/abs/1401.0364, 2014.

[5] P. Brémaud, *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*, Springer-Verlag New York, 1999.

[6] P. Collet, S. Martínez, and J. San Martín, *Quasi-stationary Distributions: Markov Chains, Diffusions and Dynamical Systems*, Springer-Verlag Berlin, Heidelberg, 2013.

[7] S. L. Cotter, K. C. Zygalakis, I. G. Kevrekidis, and R. Erban, *A constrained approach to multiscale stochastic simulation of chemically reacting systems*, J. Chem. Phys., 135 (2011), 094102.

[8] P. Del Moral and A. Doucet, *Particle motions in absorbing medium with hard and soft obstacles*, Stochastic Anal. Appl., 22 (2004), pp. 1175–1207.

[9] P. Dupuis, Y. Liu, N. Plattner, and J. D. Doll, *On the infinite swapping limit for parallel tempering*, Multiscale Model. Simul., 10 (2012), pp. 986–1022, https://doi.org/10.1137/

110853145.
[10] D. J. Earl and M. W. Deem, *Parallel tempering: Theory, applications, and new perspectives*, Phys. Chem. Chem. Phys., 7 (2005), pp. 3910–3916.
[11] P. A. Ferrari and N. Maric, *Quasi stationary distributions and Fleming-Vot processes in countable spaces*, Electron. J. Probab., 12 (2007), pp. 684–702.
[12] C. J. Geyer, *Markov chain Monte Carlo maximum likelihood*, in Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface, 1991, pp. 156–163.
[13] D. T. Gillespie, *Exact stochastic simulation of coupled chemical reactions*, J. Phys. Chem., 81 (1977), pp. 2340–2361.
[14] A. Hashemi, M. Nunez, P. Plecháč, and D. G. Vlachos, *Stochastic averaging and sensitivity analysis for two scale reaction networks*, J. Chem. Phys., 144 (2016), 074104.
[15] C. Le Bris, T. Lelievre, M. Luskin, and D. Perez, *A mathematical formalization of the parallel replica dynamics*, Monte Carlo Methods Appl., 18 (2012), pp. 119–146.
[16] D. Perez, B. P. Uberuaga, and A. F. Voter, *The parallel replica dynamics method–coming of age*, Comput. Mater. Sci., 100 (2015), pp. 90–103.
[17] N. Plattner, J. Doll, P. Dupuis, H. Wang, Y. Liu, and J. Gubernatis, *An infinite swapping approach to the rare-event sampling problem*, J. Chem. Phys., 135 (2011), 134111.
[18] M. Rathinam, L. R. Petzold, Y. Cao, and D. T. Gillespie, *Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method*, J. Chem. Phys., 119 (2003), pp. 12784–12794.
[19] R. Y. Rubinstein and D. P. Kroese, *Simulation and the Monte Carlo Method*, Wiley Ser. Probab. Stat. 707, John Wiley & Sons, New York, 2011.
[20] C. Schütte and M. Sarich, *Metastability and Markov State Models in Molecular Dynamics*, Courant Lect. Notes Math. 24, AMS, Providence, RI, 2013.
[21] E. Seneta, *Non-negative Matrices and Markov Chains*, Springer, New York, 2006.
[22] A. F. Voter, *Parallel replica method for dynamics of infrequent events*, Phys. Rev. B, 57 (1998), R13985.
[23] T. Wang and P. Plecháč, *Parallel replica dynamics method for bistable stochastic reaction networks: Simulation and sensitivity analysis*, J. Chem. Phys., 147 (2017), 234110.
[24] E. Weinan, D. Liu, and E. Vanden-Eijnden, *Nested stochastic simulation algorithm for chemical kinetic systems with disparate rates*, J. Chem. Phys., 123 (2005), 194107.