Signal Processing 122 (2016) 152–163

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

A proof of uniform convergence over time for a distributed particle filter

Joaquín Míguez [a,*], Manuel A. Vázquez [b]

[a] School of Mathematical Sciences, Queen Mary University of London, Mile End Rd, E1 4NS London, UK
[b] Departamento de Teoría de la Señal y Comunicaciones, Universidad Carlos III de Madrid, Avenida de la Universidad 30, 28911 Leganés, Madrid, Spain

ARTICLE INFO

Article history:
Received 7 April 2015
Received in revised form
23 November 2015
Accepted 25 November 2015
Available online 17 December 2015

Keywords:
Particle filtering
Convergence analysis
Wireless sensor networks
Parallelization
Distributed algorithms

ABSTRACT

Distributed signal processing algorithms have become a hot topic during the past years. One class of algorithms that have received special attention are particles filters (PFs). However, most distributed PFs involve various heuristic or simplifying approximations and, as a consequence, classical convergence theorems for standard PFs do not hold for their distributed counterparts. In this paper, we analyze a distributed PF based on the non-proportional weight-allocation scheme of Bolic *et al* (2005) and prove rigorously that, under certain stability assumptions, its asymptotic convergence is guaranteed uniformly over time, in such a way that approximation errors can be kept bounded with a fixed computational budget. To illustrate the theoretical findings, we carry out computer simulations for a target tracking problem. The numerical results show that the distributed PF has a negligible performance loss (compared to a centralized filter) for this problem and enable us to empirically validate the key assumptions of the analysis.

1. Introduction

Distributed signal processing algorithms have become a hot topic during the past years, propelled by fast technological developments in the fields of parallel computing, on one hand, and wireless sensor networks (WSNs), on the other. In parallel computing, algorithms are optimized to run fast on a set of concurrent processors (e.g., in a graphics processing unit (GPU) [39]), while signal processing methods for WSNs are designed for their implementation over a collection of low-power nodes that communicate wirelessly and share the processing tasks [36]. Popular techniques in the WSN arena include consensus-based estimators [18,27,26], diffusion-based adaptive algorithms [30,6,7] and distributed stochastic filters, including Kalman filters [38,37] and particle filters (PFs) [24,28,15,16]. While consensus and diffusion algorithms require many iterations of message passing for convergence, PFs are *a priori* better suited for online estimation and prediction tasks. Unfortunately, most distributed PFs (DPFs) rely on simplifying approximations and their convergence cannot be guaranteed by the classical theorems in [9,13,3]. One exception is the Markov chain distributed particle filter (MCDPF), for which analytical results exist [28]. However, the MCDPF converges asymptotically as sets of samples and weights are retransmitted repeatedly over the network according to a random scheme. From this point of view, it is as communication-intensive as consensus algorithms and, therefore, less appropriate for online processing compared to classical PFs.

The implementation of PFs on parallel computing systems has received considerable attention since these methods were originally proposed in [19]. The efficient implementation of PFs on parallel devices such as GPUs

* Corresponding author.
E-mail addresses: j.miguez@qmul.ac.uk (J. Míguez), mvazquez@tsc.uc3m.es (M.A. Vázquez).

http://dx.doi.org/10.1016/j.sigpro.2015.11.015
0165-1684/

and multi-core CPUs is not as straightforward as it seems *a priori* because these Monte Carlo algorithms involve a resampling step which is inherently hard to parallelize. This issue is directly addressed in [5], where two parallel implementations of the resampling step are proposed. While the approach of [5] is sound, the authors focus on implementation issues and no proof of convergence of the resulting PFs is provided. Only very recently, a number of authors have proposed distributed particle filtering schemes with provable convergence [41,40]. These methods have a fairly broad scope (the methodology in [41] can actually be seen as a generalization of the techniques in [5]) yet they appear to be less suitable for practical implementations under communications or computing power constraints, as they involve considerable parallelization overhead [40] or depend on the centralized computation of certain statistics that involve the whole set of particles in the filter [41].

The goal of this paper is to provide a rigorous proof of convergence for a DPF that relies on the distributed resampling with non-proportional weight-allocation scheme of [5] (later adapted for implementation over WSNs in [36]). Under assumptions regarding the stability of the state–space model underlying the PF, we prove that this algorithm converges asymptotically (as the number of particles generated by the filter increases) and uniformly over time. Time-uniform convergence implies that the estimation errors stay bounded without having to increase the computational effort of the filter over time. We provide explicit convergence rates for the DPF and discuss the implications of this result and the assumptions on which the analysis is based. The theoretical investigation is complemented by computer simulations of an indoor target tracking problem. For this specific system, we first show that the performance of the centralized and distributed PFs is very similar and then proceed to validate numerically a key assumption used in the analysis, related to the degree of cooperation among processing elements in the distributed computing system on which the algorithm is run.

The rest of the paper is organized as follows. In Section 2 we describe the DPF of interest. In Section 3 we prove a uniform convergence result for this filter and discuss the implications of such result. Computer simulations are presented in Section 4 and, finally, Section 5 is devoted to the conclusions.

2. A distributed particle filtering algorithm

2.1. State space systems and the standard particle filter

The stochastic filtering problem consists in tracking the posterior probability distribution of the state variables of a random dynamic system. Often, the problem is restricted to the (broad) class of Markov state space systems with conditionally independent observations. Let $\{X_n\}_{n \ge 0}$ denote the discrete-time random sequence of the system state variables, taking values on the d_x-dimensional set $\mathcal{X} \subseteq \mathbb{R}^{d_x}$, and let $\{Y_n\}_{n \ge 1}$ denote the corresponding sequence of observations, taking values on $\mathbb{R}^{d_y}$. The systems of interest are modeled by triplets of the form $\{\tau_0(dx), \quad \tau_n(dx|x_{n-1}), \quad g_n(y_n|x_n)\}_{n \ge 1}$, where τ_0 is the prior probability measure associated to the random variable (r.v.) X_0, $\tau_n(dx|x_{n-1})$ is a Markov kernel that determines the probability distribution of X_n conditional on $X_{n-1} = x_{n-1}$, and $g_n(y_n|x_n)$ is the conditional probability density function (pdf) of the random observation Y_n, given the state $X_n = x_n$, with respect to (w.r.t.) the Lebesgue measure. The latter is most often used as the *likelihood* of $X_n = x_n$ given the observation $Y_n = y_n$. We write g_n as a function of x_n explicitly, namely $g_n^{y_n}(x_n) \triangleq g_n(y_n|x_n)$, to emphasize this fact.

The goal in the stochastic filtering problem is to sequentially compute the posterior probability measures of X_n given the observations $Y_{1:n} = y_{1:n}$, denoted $\pi_n(dx)$, for $n = 0, 1, \ldots$ (note that $\pi_0 = \tau_0$). Except for a few particular cases, e.g., the Kalman [25,2] and Beneš [3] filters, π_n cannot be computed exactly and numerical approximations are pursued instead. PFs are recursive Monte Carlo algorithms that generate random discrete approximations of the probability measures $\{\pi_n; n \ge 1\}$ [9,13,3]. At time n a typical particle filtering algorithm produces a set of N random samples (often termed *particles*) and associated importance weights, $\Omega_n = \{x_n^{(i)}, w_n^{(i)*}\}_{i=1}^N$ with $W_n = \sum_{i=1}^N w_n^{(i)*}$, and approximate π_n by the way of the random probability measure $\pi_n^N = (1/W_n) \sum_{i=1}^N w_n^{(i)*} \delta_{x_n^{(i)}}$, where δ_x denotes the Dirac (unit) delta measure located at x.

It is common to analyze the convergence of PFs in terms of the approximation of integrals w.r.t. π_n [14,9,3,13,33]. To be specific, let $f: \mathcal{X} \to \mathbb{R}$ be a real function integrable w.r.t. π_n. Then we denote

$$(f, \pi_n) \triangleq \int f(x) \pi_n(dx)$$

and approximate the latter integral (generally intractable) as

$$(f, \pi_n) \approx (f, \pi_n^N) = \int f(x) \pi_n^N(dx) = \frac{1}{W_n} \sum_{i=1}^N w_n^{(i)*} f\left(x_n^{(i)}\right).$$

2.2. A distributed particle filter

We describe a PF based on the distributed resampling with non-proportional allocation (DRNA) scheme of [5, Section IV.A.3] (see also [32,4,36]). Assume that the set of weighted particles

$$\Omega_n = \{x_n^{(i)}, w_n^{(i)*}\}_{i=1}^N$$

can be split into M disjoint sets,

$$\Omega_n^m = \{x_n^{(m,k)}, w_n^{(m,k)*}\}_{k=1}^K, \quad m = 1, \ldots, M, \text{such that}$$

$\Omega_n = \cup_{m=1}^M \Omega_n^m$,each of them assigned to an independent processing element (PE). The total number of particles is $N = MK$, where M is the number of PEs and K is the number of particles per PE. At the m-th PE, $m = 1, \ldots, M$, we additionally keep track of the aggregated weight

$$W_n^{(m)*} = \sum_{k=1}^K w_n^{(m,k)*}$$

for that PE.