Uncertain Heterogeneous Algorithmic Teamwork (UHAT)

Matthew Carter, Simon Maskell, Paul Spirakis and Edward Pyzer-Knapp

EPSRC Centre for Doctoral Training in Distributed Algorithms, University of Liverpool, Liverpool, UK

Overview of Project

Recent collaboration between the University of Liverpool, IBM Research and STFC Hartree Centre developed **Sequential Monte Carlo (SMC) Samplers capable of exploiting large supercomputers** [1]. However, **supercomputing resources aren't widely accessible because: (1) they tend to be held in academic and government institutions, and (2) they're expensive.**

With this in mind, UHAT [2] aims to develop **opportunistic Sequential Monte Carlo Samplers** which **exploit large teams of commodity hardware** including desktop computers and laptops; GPU's; and mobile phones. This will allow users to access the benefits of SMC Samplers at a fraction of the cost.

Project Aims

- Develop the infrastructure that allows heterogeneous compute resources to operate effectively as a team
- Increase accessibility by interfacing with common probabilistic programming languages
- Use the developed infrastructure to solve a problem that is beneficial to society

Sequential Monte Carlo Samplers

SMC Samplers are **population-based algorithms for performing approximate numerical Bayesian inference**. SMC Samplers have been utilized in a range of applications including epidemiology and manufacturing.

In an SMC Sampler, the set of samples evolve through a series of **propagation, resampling and reweighting steps**. At the Kth iteration, all generated particles can be combined through a *particle recycling* step.

At each iteration, samples are assigned an importance weight

$$w(x_k^{(i)}) = w(x_{k-1}^{(i)}) \frac{\pi(x_k^{(i)}) \mathcal{L}(x_k^{(i)}|x_{k-1}^{(i)})}{\pi(x_{k-1}^{(i)}) \overrightarrow{q}(x_{k-1}^{(i)}|x_k^{(i)})} \quad \text{(eq 1)}$$

Quantities of interest are then estimated

$$\pi(x,y) = \pi(x,y|k) \quad \text{(eq 2)}$$
$$\approx \sum_i^N \tilde{w}_k^i \delta(x - x_k^i) = \hat{f}$$

At the Kth iteration, estimates can be recycled

$$\tilde{f} = \sum_{k=1}^{K} \lambda_k \hat{\mathrm{f}}_k \quad \text{(eq 3)}$$

HTCondor

HTCondor is an open-source high-throughput computing framework for distributing compute jobs on idle desktop computers [3]. The University of Liverpool has a HTCondor pool which consists of 1,900 cores with 2GB of memory per core [4].

Making UHAT Accessible

Probabilistic programming languages (PPLs) allow users to succinctly define probabilistic models and leverage SOTA **MCMC algorithms to perform inference**. UHAT interfaces to the **Stan PPL via PyBindStan** [5] and uses an SMC Sampler to perform inference.

CondorSMCStan: An opportunistic SMC Sampler that interfaces with Stan deployed on HTCondor

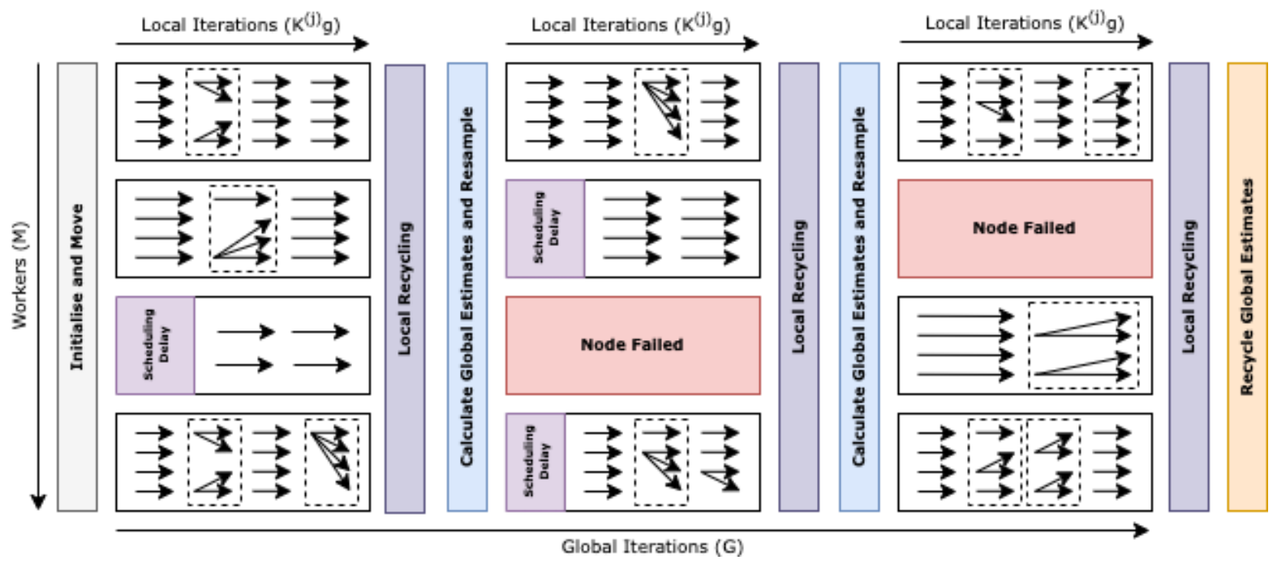
CondorSMCStan is a **Python package which distributes an SMC Sampler on commodity hardware using HTCondor.**

CondorSMCStan consists of four modules which collectively work together to distribute the SMC Sampler:

- CFMPI: Coordinator-Follower Message Passing Interface
- CondorTools: Interacting with the Condor scheduler
- PyBindStan: Python interface to the Stan
- SMCS: Modulated SMC Sampler

Continuous integration through Github actions is employed to ensure that code submitted to the repository conforms to PEP8 standards. When a pull request is submitted, code is validated using: Flake8, Black, iSort and MyPy linters. Furthermore, a unit test is associated with each component of CondorSMCStan and the validity of each component is tested using the PyTest module.

The package was developed and tested on the University of Liverpool's HTCondor pool. Moving forward, we intend to execute CondorSMCStan in the cloud – the module will be distributed on a Kubernetes cluster using HTCondor to orchestrate jobs.



Execution of CondorSMCStan on a HTCondor cluster using **4 workers which each handle a varying number of samples** (represented by the arrows, where the length of the arrow shows the propagation time). At each global iteration (G), **workers evolve samples for a fixed amount of time rather than a fixed number of steps** which helps account for varying capability of compute nodes in the network. Whilst the samples are evolved, **workers perform resampling and recycling on their local set of samples.** Following each global iteration, **the network is synchronised and resampling and recycling are performed over all workers**.

Future Work

- Increase robustness of work scheduling
- Increase the heterogeneity of the network
- Integrate cloud and supercomputing platforms in the network
- Auto-tune SMC Sampler hyperparameters during run-time

References

[1] A Single SMC Sampler on MPI that Outperforms a Single MCMC Sampler. Varsi, A., Kekempanos, L., Thiyagalingam, J. & Maskell, S.
[2] Uncertain Heterogeneous Algorithmic Teamwork. PhD Project in the EPSRC CDT in Distributed Algorithms.
[3] HTCondor Software Suite https://htcondor.org/
[4] University of Liverpool HTCondor pool http://condor.liv.ac.uk/
[5] PyBindStan: A Python interface to Github. Riddel, A., Hartikainen, A. & Carter, M.

Engineering and Physical Sciences Research Council

UNIVERSITY OF LIVERPOOL

IBM Research Europe