XBOINC: A FRAMEWORK FOR VOLUNTEER-BASED BEAM DYNAMICS SIMULATIONS USING XSUITE

C.E. Montanari[1]*, D. Di Croce, M. Giovannozzi, S. Redaelli,
F. Van Der Veken, CERN, Geneva, Switzerland
T. Pieloni, EPFL, Lausanne, Switzerland
[1]also at University of Manchester, Manchester, United Kingdom

Abstract

The Xboinc Python package extends the LHC@Home volunteer computing project by integrating the Xsuite simulation toolkit within the BOINC platform. Designed to be the successor to SixTrack's BOINC implementation, the Xboinc framework enables large-scale single-particle tracking based on the novel simulation package Xsuite. Using the computational power of global volunteer computing, Xboinc on LHC@Home provides scalable resources for advanced accelerator physics simulations, such as dynamic aperture studies. The previous functionalities were extended, notably by enabling collimation performance and loss map simulations thanks to the new Xcoll implementation in Xsuite. The synergy between distributed computing and public engagement ensures the continuity of volunteer contributions to particle accelerator research and paves the way for present and future large-scale simulation campaigns.

INTRODUCTION

Since its launch in 2004, the LHC@Home project [1] has used and contributed to the Berkeley Open Infrastructure for Network Computing (BOINC) [2] to harness the idle computing power of volunteers around the world. Volunteer computing provides large scalable computational resources that are particularly well suited for large-scale simulations with modest data transfer per job. For the accelerator physics community, this distributed computing model has historically provided a sustained computational capacity comparable to large Tier-2 Grid sites and making extremely demanding numerical studies more viable [3, 4].

For nearly two decades, LHC@Home has hosted an accelerator physics-oriented project based on SixTrack, the well-established Fortran-based single-particle tracking code [5]. SixTrack leveraged this global volunteer network to conduct large-scale tracking campaigns. These extensive simulations were critical for evaluating the Dynamic Aperture (DA) [6–8], the region in phase space where particle motion remains stable, for the Large Hadron Collider (LHC) [9], considering particle trajectories for up to $10^5 - 10^6$ turns.

Despite SixTrack's historic success, modern scientific computing increasingly relies on flexible, modular setups that can easily interface with general-purpose libraries. To address this, the Xsuite project was launched in 2021 [10, 11], combining the capabilities of legacy tools like SixTrack into a single, modern Python-based framework designed for conventional CPUs and GPU computing.

To succeed SixTrack's legacy on volunteer networks, the Xboinc Python package has been developed [12]. Xboinc seamlessly integrates the Xsuite simulation toolkit within the BOINC platform, ensuring the continuity of large-scale single-particle tracking by the volunteer community. Furthermore, transitioning to Xsuite greatly expands the physics functionalities available on LHC@Home. Notably, using Xsuite's Xcoll module [13], Xboinc enables complex particle-matter interaction simulations, allowing for advanced collimation performance evaluations and loss map simulations directly on volunteer machines, a feature that was not available in the previous SixTrack implementation. In this paper, we present an overview of the implementation of the Xboinc framework, detailing its current applications, computational advantages, and technical boundaries. Finally, we outline future development strategies to support volunteer computing for present and future particle colliders.

THE XBOINC PACKAGE

The central challenge of deploying Xsuite on BOINC stems from its architecture: Xsuite is a Python framework that generates and compiles C kernels at runtime, making it impossible to distribute as a frozen executable, as was done for SixTrack. Moreover, the majority of the volunteer machines run native Windows, further complicating the deployment. Xboinc addresses this by exploiting a key property of Xsuite: when simulating beam lines without collective effects, the computationally intensive tracking stage executes entirely in compiled C/C++ code. By leveraging Xobjects and pre-compiled kernels tied to a fixed Xsuite version, it is possible to build a universal C/C++ tracker that covers a broad set of tracking use cases while remaining fully BOINC-compatible.

The Xboinc framework produces two artefacts from this approach: a cross-platform binary executable implementing the tracker and a Python package that manages job submission and results. The binary implements a fixed *input* → *tracking* → *output* pipeline: it receives a serialised beam line and particle ensemble together with tracking parameters (e.g. number of turns), performs tracking, and returns the tracked particles. The binary is compiled for multiple platforms using a CMake + VCPKG build system, currently targeting Linux (64-bit and 32-bit) and native Windows (64-bit and 32-bit), covering the large majority of volunteer machines.

---

* carlo.emilio.montanari@cern.ch

On the physics side, Xboinc supports all beam elements included in the standard Xsuite tracker configuration, partial tracking via `ele_start` and `ele_stop` options, and collimation through Everest-based collimator models from the Xcoll module [13]. The current implementation does not support elements that simulate collective effects, interfaces with external tools (e.g. Geant4, BDSIM, Fluka), or GPU tracking; these remain topics of active development. An important caveat is that, as Xsuite itself does not guarantee bit-level reproducibility across hardware platforms, cross-platform numerical differences at the level of floating-point rounding are expected. Validation tests have confirmed that these fluctuations are comparable in magnitude to those observed between CPU and GPU tracking within Xsuite, and that they are not expected to affect significantly the physics observables of interest, such as a DA estimate based on means over angular scans.

From a scientific user perspective, Xboinc is designed to require minimal changes to an existing Xsuite workflow. The package is available on PyPI and can be installed via `pip`, with each Xboinc release tied to a specific set of Xsuite package versions, ensuring compatibility between the Python submission layer and the deployed binary. After registering with the LHC@Home server and joining the dedicated submitters group, a user replaces the standard `line.track()` call with a `JobSubmitter` object that splits the particle ensemble into batches, serialises the inputs, and submits the jobs to the BOINC server via AFS or EOS file transfers. The results are then retrieved through a `JobRetriever` iterator once the jobs have been completed and validated by the server. The complete documentation is available on the GitHub project [14]. A schematic overview summarising the Xboinc architecture is shown in Fig. 1.

Compared to the previous SixTrack-based BOINC implementation, Xboinc offers substantially improved modularity and maintainability, as its codebase is written in Python and C/C++ rather than legacy Fortran and Perl scripts. Crucially, the native C/C++ binary approach preserves the key strength of the original SixTrack deployment: an extremely low entry barrier for volunteers. Participants need only the standard BOINC client, no container runtime or additional software is required, enabling immediate participation from non-expert users worldwide. A quantitative performance comparison with CERN's local batch system is provided in the following section.

APPLICATIONS AND PERFORMANCE

The first simulation campaigns submitted through Xboinc have focused on DA studies, building on the well-established use case of the former SixTrack implementation. DA evaluations require tracking large ensembles of particles for up to 10^6 turns along a fixed set of radial and angular directions in transverse phase space, making them naturally suited to parallel distribution across volunteer resources [15]. Initial DA campaigns for LHC lattices were run successfully, with identical numerical results confirmed between local and volunteer Linux setups, and comparable results within numerical fluctuations between Linux local runs and volunteers' native Windows setups, thereby confirming the correctness of the Xboinc tracking implementation across platforms and demonstrating the operational readiness of the framework.

A primary driver of on-going Xboinc campaigns is the generation of training datasets for machine learning (ML) surrogate models of the DA [16–18]. Such models use deep neural networks to learn the mapping from a reduced set of machine parameters (optics, tunes, chromaticity, octupole strength, beam-beam parameters, etc.) to the angular DA, enabling predictions orders of magnitude faster than direct tracking. The latest studies have used a Bidirectional Encoder Representations from Transformers (BERT) architecture [19], which has shown promising results in terms of accuracy and generalisation capabilities [17]. However, training such models requires large datasets of the order of at least 10^4 DA evaluations, which would be impractical to generate using only local batch resources.

To extend these surrogate models to the High-Luminosity LHC (HL-LHC) [20] lattice, including weak-strong beam-beam effects, a large-scale DA dataset generation campaign has been run on Xboinc. Approximately 10 000 HL-LHC lattice configurations have been explored, sampling the optical parameter space (round/flat), levelling schemes, tunes, octupole strength, beam emittances, and bunch positions. Each configuration is tracked using a grid of initial conditions made up of 44 angular directions with a radial step of $0.2\,\sigma$ over 10^6 turns, which produces approximately 440 000 individual tracking jobs. The wall-clock time per-job ranges from 10 minutes to 2 hours depending on angular stability, as angles where particles are lost quickly require less computing effort than those where particles survive for the full 10^6 turns. The campaign totals an order of 10^5 CPU-hours of volunteer computing, achieved on the beta channel of the LHC@Home production server, a scale that would be impractical to source from CERN's local batch infrastructure alone. In particular, the campaign attracted strong community participation: volunteers on the LHC@Home forum expressed genuine enthusiasm for contributing to the ML dataset generation effort, further confirming the outreach value of the platform.

Xboinc is designed to complement, not replace, CERN's local HTCondor [21] batch system. Table 1 summarises the key differences for single-particle tracking campaigns. HTCondor offers low latency and full workflow flexibility, but is limited by CERN's batch quota and saturates under heavy load. Xboinc provides access to more than 10^5 volunteer jobs per hour, with no saturation observed to date, at the price of a wall-clock time of 3-4 times longer per-job due to volunteer hardware heterogeneity. For large-scale dataset generation campaigns, such as the recently performed HL-LHC DA study, this parallel channel is essential.

Collimation simulations using the Xcoll module have also been tested under Xboinc. The Everest collimator model enables particle-matter interaction simulations directly on volunteer machines, which had not been previously possible

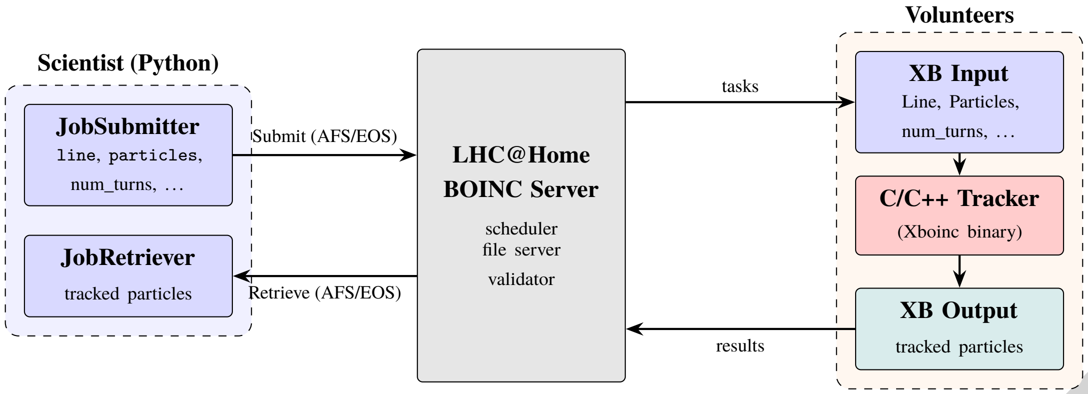

Figure 1: Schematic overview of the Xboinc workflow. A scientist uses the Python package to submit tracking jobs to the LHC@Home BOINC server via automated AFS/EOS file transfers. The server distributes tasks to volunteer machines, each of which runs the Xboinc C/C++ binary following a fixed input→tracking→output pipeline. Results are returned to the scientist upon completion.

Table 1: Comparison between CERN's HTCondor batch system and Xboinc for single-particle tracking campaigns.

	HTCondor	**Xboinc**
Job throughput	1k–10k/h (free); 10–100/h (overload)	>100k/h (unsaturated)
Exec. time	1× (dedicated node)	3–4× slower
Computational power	CERN batch quota	Volunteer-provided
Physics scope	All Xsuite workflows	Single-particle tracking

on LHC@Home. This opens the pathway to distributed loss map computations, which are a key input for collimation system design and machine protection studies [22], although they require more complex workflows than DA evaluations and will require further development of the Xboinc framework before being fully supported.

FUTURE DEVELOPMENTS

The limitations of the current Xboinc framework define a clear development roadmap. Simulations involving collective effects (such as space charge, strong-strong beam-beam effects, wakefields) require continuous inter-particle communication and are fundamentally incompatible with the asynchronous, distributed job model of volunteer computing; they will remain confined to dedicated batch systems. Tracking with external Monte Carlo tools such as Geant4 or Fluka cannot easily be reduced to a self-contained C/C++ binary, but the BOINC Universal Docker Application (BUDA) framework [23] offers a promising pathway: Docker-based BOINC jobs can ship a complete Xsuite environment to volunteers supporting containers, an approach that is becoming increasingly mainstream in the BOINC community. However, container-based tasks inevitably reach a smaller volunteer base than native binaries, as they require Docker support on the host machine. The two approaches are not mutually exclusive: a native binary branch for single-particle tracking and a BUDA branch for broader physics workflows could coexist as separate subprojects, once the native Xboinc application has established sufficient volunteer adoption and server stability. On a shorter time scale, particle monitor objects and impact tables in Xsuite are currently being integrated and tested within Xboinc, expanding diagnostic capabilities. Additional targets include aperture interpolation, lattice variations, and backtracking.

CONCLUSIONS

Xboinc has been successfully deployed as the Xsuite-based successor to SixTrack on LHC@Home, delivering orders of 10^5 CPU-hours for HL-LHC DA studies, the primary investment being software maintenance rather than hardware or batch quota. It functions as a parallel channel alongside standard batch systems: large exploratory campaigns are offloaded to the global volunteer pool, freeing local resources for workflows requiring lower latency or broader physics scope. Beyond its scientific impact, LHC@Home remains an exemplary outreach platform, giving the global public a direct and meaningful role in real accelerator physics research. The promising results presented here call for a strategic decision at CERN on how to proceed with volunteer computing for present and future collider programmes, in order to secure the long-term sustainability and further development of the framework.

ACKNOWLEDGMENTS

This work was carried out under the auspices and with the support of the Swiss Accelerator Research and Technology (CHART) programme and the Swiss Data Science Centre project grant C20-10. The authors would also like to thank the LHC@Home volunteers for their invaluable contributions to the Xboinc campaigns, the IT department, and, specifically, the LHC@Home server team for their support in deploying and maintaining the framework.

REFERENCES

[1] J. Barranco *et al.*, *Open Engineering*, vol. 7, no. 1, pp. 379–393, 2017. doi:doi:10.1515/eng-2017-0042

[2] D. P. Anderson, "BOINC: a platform for volunteer computing", *J. Grid Comput.*, vol. 18, no. 1, pp. 99–122, Mar. 2020. doi:10.1007/s10723-019-09497-9

[3] M. Giovannozzi *et al.*, "LHC@home: a Volunteer Computing System for Massive Numerical Simulations of Beam Dynamics and High Energy Physics Events", in *Proc. IPAC'12*, New Orleans, LA, USA, May 2012, May 2012, paper MOPPD061, pp. 505–507. https://jacow.org/IPAC2012/papers/MOPPD061.pdf

[4] D. Cameron *et al.*, "All Grown-Up: 18 Years of LHC@Home", *Eur. J. Phys. Web of Conf.*, vol. 295, p. 04004, 2024. doi:10.1051/epjconf/202429504004

[5] R. De Maria *et al.*, "SixTrack version 5: status and new developments", *J. Phys. Conf. Ser.*, vol. 1350, no. 1, p. 012129, Nov. 2019. doi:10.1088/1742-6596/1350/1/012129

[6] M. Giovannozzi, W. Scandale, and E. Todesco, "Dynamic aperture extrapolation in presence of tune modulation", *Phys. Rev. E*, vol. 57, no. 3, p. 3432, Mar. 1998. doi:10.1103/PhysRevE.57.3432

[7] M. Giovannozzi, "A proposed scaling law for intensity evolution in hadron storage rings based on dynamic aperture variation with time", *Phys. Rev. Spec. Top. Accel. Beams*, vol. 15, no. 2, p. 024001, Feb. 2012. doi:10.1103/PhysRevSTAB.15.024001

[8] A. Bazzani, M. Giovannozzi, E. H. Maclean, C. E. Montanari, F. F. Van der Veken, and W. Van Goethem, "Advances on the modeling of the time evolution of dynamic aperture of hadron circular accelerators", *Phys. Rev. Accel. Beams*, vol. 22, no. 10, p. 104003, Oct. 2019. doi:10.1103/PhysRevAccelBeams.22.104003

[9] O. S. Brüning *et al.*, *LHC design report*. Geneva: CERN, 2004. doi:10.5170/CERN-2004-003-V-1

[10] G. Iadarola *et al.*, "Xsuite: an integrated beam physics simulation framework", in *Proc. IPAC'24*, Nashville, TN, May 2024, pp. 2623–2626. doi:10.18429/JACoW-IPAC2024-WEPR56

[11] G. Iadarola *et al.*, "Xsuite: an integrated beam physics simulation framework", Sep. 2023. doi:10.48550/arXiv.2310.00317

[12] Xsuite Team, Xboinc: Xsuite BOINC interface, Feb. 2026. https://github.com/xsuite/xboinc

[13] F. F. Van der Veken *et al.*, in *Proc. HB'23*, Geneva, Switzerland, Mar. 2023, pp. 474–478. doi:10.18429/JACoW-HB2023-THBP13

[14] Xsuite Team, Xsuite BOINC interface documentation, 2026, https://xsuite.github.io/xboinc/

[15] M. Giovannozzi and E. McIntosh, "Development of parallel codes for the study of nonlinear beam dynamics", *Int. J. Mod. Phys. C*, vol. 08, no. 02, pp. 155–170, Nov. 1997. doi:10.1142/S0129183197000163

[16] D. Di Croce *et al.*, "Optimizing dynamic aperture studies with active learning", *J. Instrum.*, vol. 19, no. 04, p. P04004, Apr. 2024. doi:10.1088/1748-0221/19/04/P04004

[17] D. Di Croce, M. Giovannozzi, C. E. Montanari, T. Pieloni, S. Redaelli, and F. F. Van der Veken, "Assessing the performance of deep learning predictions for dynamic aperture of a hadron circular particle accelerator", *Instruments*, vol. 8, no. 4, p. 50, 2024. doi:10.3390/instruments8040050

[18] C. E. Montanari *et al.*, "Machine learning techniques for uncertainty estimation in dynamic aperture prediction", *Computers*, vol. 14, no. 7, p. 287, 2025. doi:10.3390/computers14070287

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding", May 2019. doi:10.48550/arXiv.1810.04805

[20] G. Apollinari *et al.*, *High-Luminosity Large Hadron Collider (HL–LHC)*. Geneva: CERN, 2017. doi:10.23731/CYRM-2017-004

[21] P. Llopis, C. Lindqvist, N. Høimyr, D. van der Ster, and P. Ganz, "Integrating HPC into an agile and cloud-focused environment at CERN", *EPJ Web Conf.*, vol. 214, p. 07025, 2019. doi:10.1051/epjconf/201921407025

[22] B. Salvachua *et al.*, "Decomposition of beam losses at LHC", in *Proc. IPAC'17*, Copenhagen, Denmark, May 2017, May 2017, pp. 88–91. doi:10.18429/JACoW-IPAC2017-MOPAB009

[23] D. P. Anderson, BUDA overview, Dec. 2024, https://github.com/BOINC/boinc/wiki/BUDA-overview