FAIR AI Models in High Energy Physics

Haoyang Li[1,*,**], Javier Duarte[1], Avik Roy[2], Ruike Zhu[2,3], E. A. Huerta[3,4], Daniel Diaz[1], Philip Harris[5], Raghav Kansal[1], Daniel S. Katz[2], Ishaan H. Kavoori[1], Volodymyr V. Kindratenko[2], Farouk Mokhtar[1,6], Mark S. Neubauer[2], Sang Eon Park[5], Melissa Quinnan[1], Roger Rusack[7], and Zhizhen Zhao[2]

[1]University of California San Diego, La Jolla, California 92093, USA
[2]University of Illinois Urbana-Champaign, Urbana, Illinois 61801, USA
[3]Argonne National Laboratory, Lemont, Illinois 60439, USA
[4]The University of Chicago, Chicago, Illinois 60637, USA
[5]Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA
[6]Halıcıoğlu Data Science Institute, La Jolla, California 92093, USA
[7]The University of Minnesota, Minneapolis, Minnesota 55405, USA

Abstract. The findable, accessible, interoperable, and reusable (FAIR) data principles serve as a framework for examining, evaluating, and improving data sharing to advance scientific endeavors. There is an emerging trend to adapt these principles for machine learning models—algorithms that learn from data without specific coding—and, more generally, AI models, due to AI's swiftly growing impact on scientific and engineering sectors. In this paper, we propose a practical definition of the FAIR principles for AI models and provide a template program for their adoption. We exemplify this strategy with an implementation from high-energy physics, where a graph neural network is employed to detect Higgs bosons decaying into two bottom quarks.

1 Introduction

Machine learning, a prominent branch of artificial intelligence, has significantly influenced experimental high energy physics (HEP). For instance, ML played a pivotal role in the 2012 discovery of the Higgs boson [1, 2] and in real-time identification of specific events amidst millions of background occurrences at the CERN Large Hadron Collider (LHC) [3, 4]. To maximize the impact and utility of such AI models, it's recommended to adopt FAIR principles, ensuring they are findable, accessible, interoperable, and reusable. These principles, initially crafted for scientific datasets [5], have been adapted for research software [6–9] and other areas, including AI tool development [10, 11]. However, applying FAIR principles to AI models presents challenges due to the unique nature of AI models. To address this, we propose a definition of FAIR principles specifically for AI models, aiming to improve research reusability and reproducibility. We also outline a method to streamline the creation and release of FAIR AI models.

*e-mail: hal113@ucsd.edu
**Speaker

2 Methods

2.1 FAIR principles for AI models

Researchers have explored applying FAIR principles to research software [6–9]. However, creating machine learning models involves various digital components, from software and data to tools and diverse hardware. Depending on the goal, AI models can be adjusted for speed, parallel operations, or hardware compatibility using different software tools. For accurate replication and reuse, it's vital to detail the entire development journey of the model. To use the AI models on new, possibly raw data, clear instructions for data handling are necessary.

Practically, AI models are often developed in platforms like Scikit-learn [12], TensorFlow [13], PyTorch [14], or ONNX [15] and saved as files. The storage format, whether hardware-specific or general, can influence the behavior and results of the model. Preparation steps, crucial to the performance of the model, might be in separate scripts or within the model. While there is a push to share such codes on platforms like GitHub, many lack essential details, making replication hard [16–18]. This has led to AI reproducibility challenges [17, 19]. Considering these points, we offer a definition for a FAIR AI model in line with the original FAIR data principles [5] for AI in Table 1. In brief, (F) the model and its associated metadata are easy to find for both humans and machines, (A) the model and its metadata are retrievable via standardized protocols, (I) the model interoperates with other models, data, and/or software, and (R) the model is both usable and reusable.

For an ML model to meet FAIR standards, its training dataset must also comply with these guidelines and community standards due to its role in the development of the model. We have detailed FAIR AI principles, influenced by the Research Data Alliance's FAIR4RS Working Group, and shared with the RDA's FAIR4ML group [6–8, 20, 21]. These principles are foundational for FAIR models, but ensuring a model is shareable and adaptable may require more. Reproducibility is challenged by backend optimizations, with frameworks like PyTorch and ONNX introducing potential output variations even on the same hardware [22]. Given hardware and precision variations, we interpret reproducibility as achieving broadly consistent results, allowing slight individual data variations.

2.2 Cookiecutter4fair: FAIR AI project template

Software templates can encourage best practices. One example is Cookiecutter Data Science [23], which is tailored for data science projects. It offers a logical and somewhat standardized yet adaptable project framework on GitHub for executing and disseminating data science tasks. Inspired by this, we developed a variant named `cookiecutter4fair` [24], which includes extra functionalities to support our FAIR principles.

2.2.1 Usage

Users can initiate a new FAIR AI project using the command `cookiecutter https://github.com/FAIR4HEP/cookiecutter4fair` command that references a GitHub-hosted template. This process prompts users for various project details, then creates a template as depicted in Fig. 1.

The repository's questions can be adjusted in the `cookiecutter.json` file, and the `Makefile` offers various project commands, emphasizing that analysis operations form a

Table 1. The suggested FAIR guidelines for fully trained AI models that are used for inference solely. Derived from the original FAIR principles, this adaptation begins by substituting "data" with "AI models" and then further modifies based on the distinct features and applications of AI models compared to datasets. To accommodate retraining scenarios, the definition of the "Reusability" principle within these guidelines can be further refined.

F: The AI model, and its associated metadata, are easy to find for both humans and machines.
F1. The AI model is assigned a globally unique and persistent identifier. F2. The AI model is described with rich metadata. F3. Metadata clearly and explicitly include the identifier of the AI model they describe. F4. Metadata and the AI model are registered or indexed in a searchable resource.
A: The AI model, and its metadata, are retrievable via standardized protocols.
A1. The AI model is retrievable by its identifier using a standardized communications protocol. A1.1. The protocol is open, free, and universally implementable. A1.2. The protocol allows for an authentication and authorization procedure, where necessary. A2. Metadata are accessible, even when the AI model is no longer available.
I: The AI model interoperates with other models, data, and/or software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.
I1. The AI model reads, writes and exchanges data in a way that meets domain-relevant community standards. I2. The AI model includes qualified references to other objects, including the (FAIR) data used to train the model.
R: The AI model is both usable (for inference) and reusable (can be understood, built upon, or incorporated into other models and/or software).
R1. The AI model is described with a plurality of accurate and relevant attributes. R1.1. The AI model is given a clear and accessible license. R1.2. The AI model is associated with detailed provenance, such as information about the input data preparation and training process. R2. The AI model includes qualified references to other models and/or software, such as dependencies. R3. The AI model meets domain-relevant community standards.

directed acyclic graph (DAG). If data is on Zenodo [25], it can be downloaded via the `zenodo_get` command line utility [26]. A `Dockerfile` sets up the Python environment with the dependencies specified in `requirements.txt`, and once built, the Docker image can be interactively run. Additional scripts offer more adaptability, and post-creation, users can structure their code and documentation to align with FAIR principles.

2.2.2 Considerations and mapping to FAIR principles

Findability: AI models can be uploaded to GitHub, GitLab, BitBucket, or more specialized hubs like DLHub [27, 28], OpenML [29], MLCommons [30], AI Model Share [31], and

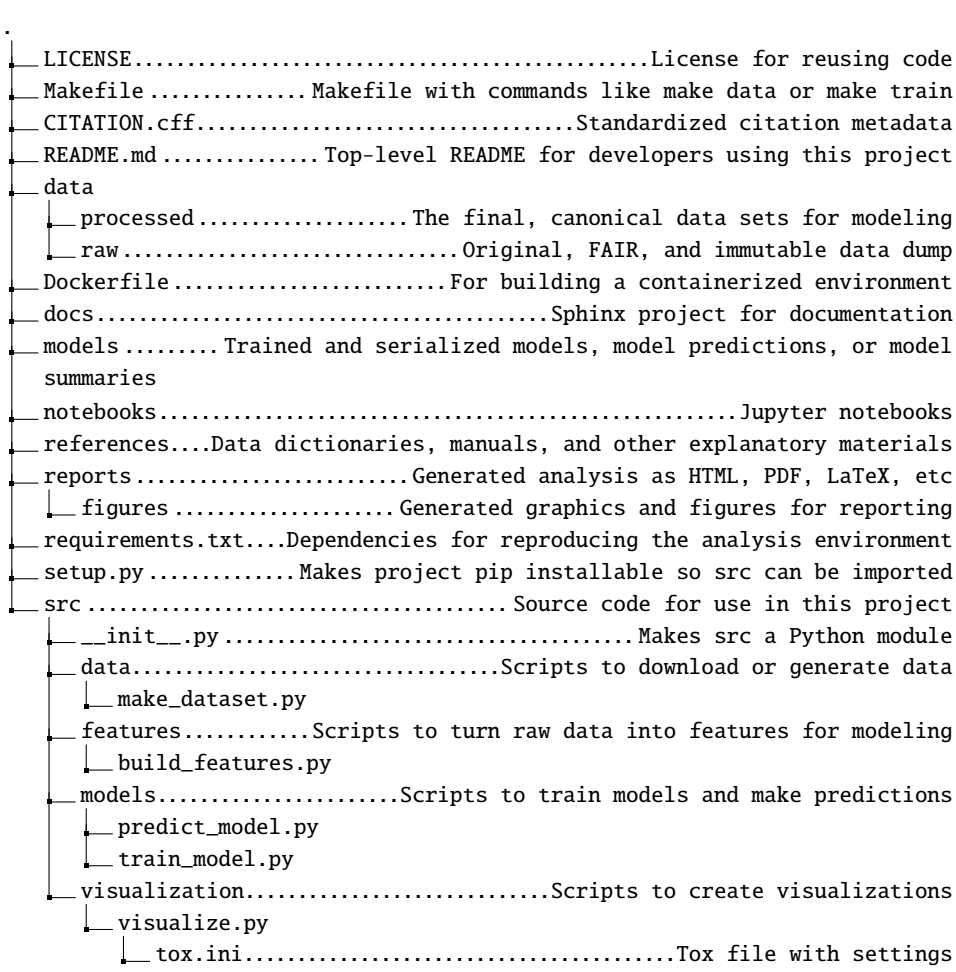

Figure 1. The directory tree of the `cookiecutter4fair` v1.0.0 [24] project template. `src` contains the main Python source code. For generating documentation, the `docs` subdirectory includes a Sphinx project.

Hugging Face [32] for sharing models. Using Zenodo [25] can generate a DOI for the repository and store metadata, and Hugging Face offers DOIs for datasets and models [33].

Accessibility: AI models can be retrieved with an identifier of the standardized protocol. Model repositories should adopt this server-side protocol, while community tools in languages like Python, R, and other programming language should support its client side.

Interoperability: For effective interoperability, it is essential that the metadata of the AI model comprehensively details its design, training, and inputs, encompassing any preprocessing for the raw data and its origin. Utilizing standardized APIs, like those from DLHub, HuggingFace, or NVIDIA Triton Server [34], can further facilitate machine interoperability.

Reusability: For effective reusability, it is crucial to outline the necessary software, tools, and dependencies for effortlessly deploying an AI model to derive insights from datasets in any computing setting. This approach should be independent of specific hardware. Achieving this can be facilitated by container platforms like Docker [35] or Apptainer [36].

Table 2 illustrates how the features of the `cookiecutter4fair` AI project template align with the suggested FAIR principles for AI models. Many features, like generating a license file and Dockerfile, are fully automated. Some, like model uploading to Zenodo, are semi-automated; for instance, the GitHub-Zenodo bridge can auto-generate updated entries for new GitHub releases. The template also fills a `CITATION.cff` file [37] with citation data for Zenodo's use. However, certain tasks, like uploading the model to DLHub, still need manual intervention.

Table 2. Correlation between the current capabilities of the `coookiecutter4fair` AI project template and our proposed FAIR principles for AI models. The $*$ symbol denotes processes that are not entirely automated yet and need additional manual intervention.

Principle	GitHub repository	Zenodo upload	DLHub upload	Docker or Apptainer image	License
Findable	✓				
Accessible		✓	$*$		
Interoperable				✓	
Reusable			$*$	✓	✓

3 A FAIR implementation: $\mathrm{H} \to \mathrm{b\bar{b}}$ interaction network

The interaction network (IN) [38], initially designed to study physical dynamics, was tailored to jet classification. Here, we showcase a FAIR implementation of the interaction network for differentiating $\mathrm{H} \to \mathrm{b\bar{b}}$ jets from quantum chromodynamics (QCD) multijet events [39]. Examining the $\mathrm{H} \to \mathrm{b\bar{b}}$ decay rate and its deviation from the standard model could hint at undiscovered physics.

The FAIR $\mathrm{H} \to \mathrm{b\bar{b}}$ interaction network model was created on GitHub and hosted on Zenodo [40]. The GitHub repository was initialized using the template described in Section 2.2. The repository contains a script that processes raw data from the CERN Open Data portal [41] and scripts for training and predictions to replicate published outcomes, all organized in a Makefile as a DAG. Figure 2 shows the IN model architecture. For a detailed description of the model and chosen hyperparameters, see Moreno et al. [39]. The repository also features two Dockerfiles for reproducible CPU or GPU model training environments, with prebuilt images available on DockerHub. Automated documentation, training workflows, Docker container building, and continuous integration are facilitated through GitHub Actions. Additionally, each new software release on GitHub is assigned a DOI via the Zenodo-GitHub bridge.

The trained ML model has also been made publicly accessible [42] and reusable for inference through DLHub [27, 43], which offers a custom software development kit (SDK) called `dlhub_sdk`, enabling users to package and maintain a trained model with all essential dependencies. After publishing a model, DLHub provides a dedicated API for remote inference tasks using `funcX`, a system that efficiently deploys tasks across various computing environments [44]. The model deployment process is streamlined using a notebook template

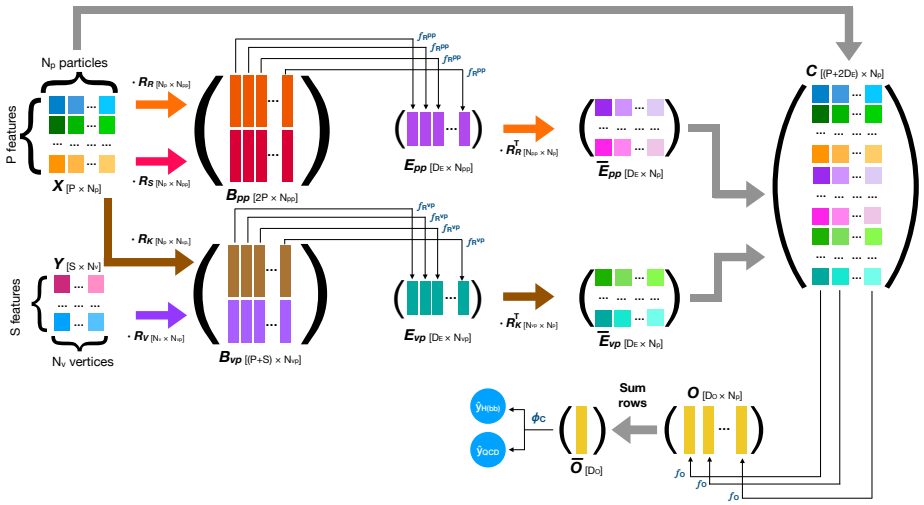

Figure 2. Model architecture and the dataflow in the IN model [39]. The hyperparameters of the model and the dimensions of the input data are provided in Moreno et al. [39].

provided by DLHub developers, which guides users in implementing inference code, declaring dependencies, and adding metadata. Once completed, these templates can be submitted to DLHub developers for further processing. The published model comes with comprehensive details, including a DOI, author list, and usage instructions. Additionally, DLHub's SDK lets users delve into the metadata of the model, detailing its creation and functionalities.

4 Conclusion

We have proposed a practical definition of FAIR principles tailored for machine learning (ML) and broader artificial intelligence (AI) models. To encourage compliance with these guidelines, we have provided a FAIR AI project template and showcased its application using a model that distinguishes H $\rightarrow$ b$\bar{\text{b}}$ events from QCD events.

References

[1] S. Chatrchyan et al. (CMS), Phys. Lett. B **716**, 30 (2012), 1207.7235

[2] G. Aad et al. (ATLAS), Phys. Lett. B **716**, 1 (2012), 1207.7214

[3] J. Duarte et al., JINST **13**, P07027 (2018), 1804.06913

[4] CMS Collaboration (CMS), CMS Technical Design Report CERN-LHCC-2020-004. CMS-TDR-021 (2020), https://cds.cern.ch/record/2714892

[5] M.D. Wilkinson, M. Dumontier, I.J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.W. Boiten, L.B. da Silva Santos, P.E. Bourne et al., Sci. Data **3**, 160018 (2016)

[6] D.S. Katz, M. Gruenpeter, T. Honeyman, L.J. Hwang, M.D. Wilkinson, V. Sochat, H. Anzt, C.A. Goble (2021), 2101.10883

[7] D.S. Katz, M. Gruenpeter, T. Honeyman, Patterns **2**, 100222 (2021)

[8] N.P. Chue Hong, D.S. Katz, M. Barker, A.L. Lamprecht, C. Martinez, F.E. Psomopoulos, J. Harrow, L.J. Castro, M. Gruenpeter, P.A. Martinez et al., *FAIR Principles for Research Software (FAIR4RS Principles)* (2022)

[9] M. Barker, N. Chue Hong, D. Katz, A.L. Lamprecht, C. Martinez Ortiz, F. Psomopoulos, J. Harrow, L. Castro, M. Gruenpeter, P. Martinez et al., Sci. Data **9**, 622 (2022)

[10] G. Verma, M. Emani, C. Liao, P.H. Lin, T. Vanderbruggen, X. Shen, B. Chapman, *HPCFAIR: Enabling FAIR AI for HPC Applications*, in *2021 IEEE/ACM Workshop on Machine Learning in High Performance Computing Environments (MLHPC)* (2021), p. 58

[11] N. Ravi, P. Chaturvedi, E.A. Huerta, Z. Liu, R. Chard, A. Scourtas, K.J. Schmidt, K. Chard, B. Blaiszik, I. Foster, Sci. Data **9**, 657 (2022), 2207.00611

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg et al., J. Mach. Learn. Res. **12**, 2825 (2011)

[13] M. Abadi et al., *TensorFlow: Large-scale machine learning on heterogeneous systems* (2015), 1603.04467, https://www.tensorflow.org/

[14] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, in *Advances in Neural Information Processing Systems*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Curran Associates, Inc., 2019), Vol. 32, https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf

[15] J. Bai, F. Lu, K. Zhang et al., *Open Neural Network Exchange*, https://github.com/onnx/onnx (2017), https://github.com/onnx/onnx

[16] S. Wattanakriengkrai, B. Chinthanet, H. Hata, R.G. Kula, C. Treude, J. Guo, K. Matsumoto, J. Syst. Softw. **183**, 111117 (2022)

[17] J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Lariviere, A. Beygelzimer, F. d'Alche Buc, E. Fox, H. Larochelle, J. Mach. Learn. Res. **22**, 1 (2021), 2003.12206

[18] B. Haibe-Kains, G.A. Adam, A. Hosny, F. Khodakarami, T. Shraddha, R. Kusko, S.A. Sansone, W. Tong, R.D. Wolfinger, C.E. Mason et al., Nature **586**, E14 (2020)

[19] K. Sinha et al., *ML reproducibility challenge 2022* (2022), https://paperswithcode.com/rc2022

[20] D.S. Katz, *Defining FAIR for machine learning (ML)* (2021), https://www.rd-alliance.org/defining-fair-machine-learning-ml

[21] D.S. Katz, *FAIR software and FAIR ML models* (2022), https://doi.org/10.5281/zenodo.6647819

[22] PyTorch Team, *PyTorch GitHub Issue #87398: Model outputs different values after ONNX export* (2022), https://github.com/pytorch/pytorch/issues/87398\#issuecomment-1338230472

[23] Driven Data, *Cookiecutter data science* (2022), https://drivendata.github.io/cookiecutter-data-science/

[24] FAIR4HEP, *Cookiecutter4fair: v1.0.0* (2022), https://github.com/fair4hep/cookiecutter4fair

[25] European Organization For Nuclear Research, OpenAIRE, *Zenodo* (2013), https://www.zenodo.org/

[26] D. Völgyes, *Zenodo_get: A downloader for zenodo records* (2020), https://github.com/dvolgyes/zenodo_get

[27] Z. Li, R. Chard, L. Ward, K. Chard, T.J. Skluzacek, Y. Babuji, A. Woodard, S. Tuecke, B. Blaiszik, M.J. Franklin et al., J. Parallel. Distrib. Comput. **147**, 64 (2021)

[28] K. Chard, M. Lidman, B. McCollam, J. Bryan, R. Ananthakrishnan, S. Tuecke, I. Foster, Future Gener. Comput. Syst. **56**, 571 (2016)

[29] J. Vanschoren, J.N. van Rijn, B. Bischl, L. Torgo, SIGKDD Explorations **15**, 49 (2013)

[30] MLCommons, *MLCommons* (2022), https://mlcommons.org

[31] AI Model Share Project, *AI Model Share Platform* (2022), https://www.modelshare.org/

[32] Hugging Face, *Hugging Face* (2024), https://www.huggingface.co/

[33] S. Luccioni, S. Bouchot, C. Akiki, A. Leroy, *Introducing DOI: the digital object identifier to datasets and models* (2022), https://huggingface.co/blog/introducing-doi

[34] NVIDIA, *NVIDIA Triton Inference Server*, https://developer.nvidia.com/nvidia-triton-inference-server (2022)

[35] D. Merkel, Linux J. **2014** (2014)

[36] G.M. Kurtzer, V. Sochat, M.W. Bauer, PLoS ONE **12** (2017)

[37] S. Druskat, J.H. Spaaks, N. Chue Hong, R. Haines, J. Baker, S. Bliven, E. Willighagen, D. Pérez-Suárez, A. Konovalov, *Citation File Format* (2021), https://citation-file-format.github.io/

[38] P.W. Battaglia, R. Pascanu, M. Lai, D. Rezende, K. Kavukcuoglu, *Interaction Networks for Learning about Objects, Relations and Physics*, in *Advances in Neural Information Processing Systems*, edited by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, R. Garnett (Curran Associates, Inc., 2016), Vol. 29, 1612.00222, https://proceedings.neurips.cc/paper/2016/file/3147da8ab4a0437c15ef51a5cc7f2dc4-Paper.pdf

[39] E.A. Moreno, T.Q. Nguyen, J.R. Vlimant, O. Cerri, H.B. Newman, A. Periwal, M. Spiropulu, J.M. Duarte, M. Pierini, Phys. Rev. D **102**, 012010 (2020), 1909.12285

[40] J.M. Duarte, B. Li, A. Roy, R. Zhu, *Hbb Interaction Network: v0.1.1* (2022), https://github.com/FAIR4HEP/hbb_interaction_network

[41] CMS Collaboration, J. Duarte, *Sample with jet, track and secondary vertex properties for Hbb tagging ML studies (HiggsToBBNTuple_HiggsToBB_QCD_RunII_13TeV_MC)* (2019), CERN Open Data Portal

[42] Moreno, E. A., Nguyen, T. Q., Vlimant, J.-R., Cerri, O., Newman, H. B., Periwal, A., Spiropulu, M., Duarte, J. M., Pierini, M., Zhu, R., Roy, A., Huerta, E. A., *FAIR Interaction Network Model for Higgs Boson Detection*, The Data and Learning Hub for Science (DLHub) (2022)

[43] R. Chard, Z. Li, K. Chard, L. Ward, Y. Babuji, A. Woodard, S. Tuecke, B. Blaiszik, M.J. Franklin, I. Foster, *DLHub: Model and data serving for science*, in *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)* (IEEE, 2019), p. 283

[44] R. Chard, Y. Babuji, Z. Li, T. Skluzacek, A. Woodard, B. Blaiszik, I. Foster, K. Chard, *funcX: A federated function serving fabric for science*, in *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing* (Association for Computing Machinery, New York, NY, USA, 2020), HPDC '20, p. 65, ISBN 9781450370523, 2005.04215