General comments:

The main contribution of this paper is the implementation of hierarchical clustering using OpenMP and MPI parallelization techniques. The problem presented is of size between 10^5 and 10^6 for either measured or modelled air quality variables. Although the proposed algorithm is a clear improvement of the naïve implementation on a single core, there could be further room for improvement using algorithmic techniques or alternative hierarchical clustering methods.

Despite the parallelization on a High-Performance Computing (HPC) infrastructure with many cores, the wall clock time is still of several hours, while an alternative hierarchical clustering method such as accelerated HDBSCAN* would be expected to run in less than a minute for a problem of similar size (or alternatively allows to run problems of magnitude 10^7 to 10^8 in hours if the asymptotic performance is extrapolated from 1705.07321.pdf (arxiv.org), Figure 6). As such, a comparison with alternative hierarchical clustering algorithms is needed (see specific comment 1).

The paper does not include many references on clustering large datasets in other fields. Discussion about alternative (hierarchical) clustering algorithms should be included. Three algorithmic techniques for accelerating hierarchical clustering should be discussed (implementation might prove more challenging for the latter two): connectivity constraint (see specific comment 2), efficient data structures (see specific comment 3) and triangular inequality for true distance metrics (see specific comment 4). Other works that have provided implementation of parallelization of clustering algorithms (not necessarily hierarchical) on HPC, multi-thread CPUs or GPUs should be included (see specific comment 5).

I would restructure the paper to reduce the emphasis on “an introduction to hierarchical clustering for non-specialists” and focus more on the practical usage and technical analysis of the hierarchical clustering implementation on OpenMP and MPI (specific comments 6-8, technical comments for L65-73 and L137-195).

Finally, the presentation of the pre-processing of the NAPS dataset on the second example was not done in sufficient details for reproducible results (see specific comment 9).

Specific comments:

1.
    a. Add references and discussion on HDBSCAN*, an alternative hierarchical clustering method with lower algorithmic complexity:
        i. Campello, J.G.B. et al., Density-Based Clustering Based on Hierarchical Density Estimates, LNCS, 2013
        ii. *McInnes, L. et al., Accelerated Hierarchical Density Based Clustering, IEEE International Conference on Data Mining Workshops (ICDMW), 2017 (**DOI:** 10.1109/ICDMW.2017.12)*
    b. It is imperative that the authors compare the results against several of the following options to ascertain the value of the proposed implementation:
        i. Scikit-Learn HDBCAN* implementation: sklearn.cluster.HDBSCAN — scikit-learn 1.3.2 documentation. A final assignment of noisy points to the closest clusters could be done as a post-processing step to obtain similar results than hierarchical clustering.

        ii. Accelerated HDBSCAN* implementation: GitHub - scikit-learn-contrib/hdbscan: A high performance implementation of HDBSCAN clustering.
        iii. Scikit-learn implementation of hierarchical clustering (as a baseline): sklearn.cluster.AgglomerativeClustering — scikit-learn 1.3.2 documentation
        iv. Scikit-learn implementation of hierarchical clustering, but with grid-cell connectivity constraints (for air quality model data), see A demo of structured Ward hierarchical clustering on an image of coins — scikit-learn 1.3.2 documentation for a usage example.
        v. Scikit-learn implementation of K-means sklearn.cluster.KMeans — scikit-learn 1.3.2 documentation with the same number of clusters as the results presented in the paper for comparison.
    c. Considerations can be the following:
        i. For which algorithms it is preferable to use pre-computed pairwise dissimilarity matrix? What is the memory requirement to load this matrix in the RAM? What are the memory constraints of these algorithms that the proposed OpenMP/MPI implementation solves for hierarchical clustering?
        ii. Timing comparison. If the proposed algorithm does not compare favorably to others (it is not expected it will according to algorithmic complexity), maybe it could still be used advantageously for the dissimilarity matrix pre-computation?
        iii. How the other results compare to the hierarchical clustering with median linkage and 1-R metric presented in the paper. A quantitative score such as Rand Index could be considered as well as a qualitative comparison. The question is how the clustering results are sensitive to the choice of method (and its optional parameters)? For example, K-means could be very fast, but not very accurate (and losing the flexibility of hierarchical clustering).
        iv. Scaling in function of the number of data points (taking a sub-sample of the dataset) for different number of clusters and data dimension. Presenting the results in log-log plots is the most informative as it allows to easily estimate computational budget for larger datasets.

2. Add discussion on connectivity constraints for clusters:
    a. Are all clusters found connected for air quality model data? From the cluster maps (Figures 4 and 5), it appears to be so. It would be worth mentioning if so or analyzing when it does not occur. Can a similar connectivity constraint be found for station data?
    b. Employing a connectivity constraint (an implementation equivalent to the connectivity keyword argument in sklearn.cluster.AgglomerativeClustering, see recommendation 1.a.iv) could potentially large speed-up and memory savings. I recommend exploring this possibility for further speed-up of the algorithm. Note however that in this case we would need to be careful with the choice of linkage function (such as Ward's criterion) to avoid "the rich getting richer" phenomenon (getting a few very large clusters and many very small clusters).

3. Add references and discussion on more efficient hierarchical clustering algorithms (see technical comments for L10 for more details on computational efficiency comparison):
    a. Improved data structure that speed-up hierarchical clustering (in both theory and practice): *Eppstein,D., Fast Hierarchical Clustering and Other Applications of Dynamic*

    Closest Pairs, ACM Journal of Experimental Algorithmics, 2000 (https://doi.org/10.1145/351827.351829)
    b. *Defays, D., An efficient algorithm for complete link method, The Computer Journal, 1977 (*https://doi.org/10.1093/comjnl/20.4.364)

4. If a true distance metric is used such as the Euclidean distance, then the use of the triangle inequality could reduce memory requirements and potentially speed-up the algorithm. The triangular inequality has been used for K-means in 10.1109/ACCESS.2019.2907885, but it has also been explored for hierarchical clustering. Please mention the references and discuss how exploiting the triangular inequality or other data summarization techniques could potentially speed-up computation while reducing memory requirements.
    a. *Zhou J. and Sander, J., Data Bubbles for Non-Vector Data: Speeding-up Hierarchical Clustering in Arbitrary Metric Spaces, Proceedings VLDB Conference, 2003 (https://doi.org/10.1016/B978-012722442-8/50047-1)*
    b. *Kull M., Fast Clustering in Metric Spaces, Master Thesis, 2004 (pdf: content (ut.ee))*
5. Please add references and discussion on other works doing parallelization of clustering algorithms on MPI/OpenMP, multi-thread CPUs or GPUs (also check references therein and paper citing these works):
    a. *Kweldlo W. and Czochanski P.J., A Hybrid MPI/OpenMP Parallelization of K-Means Algorithms Accelerated Using the Triangle Inequality, IEEE Access, 2019 (**DOI:** 10.1109/ACCESS.2019.2907885)*
    b. *Woodley, A et al., Parallel K-Tree: A multicore, multinode solution to extreme clustering, Future Generation Computer Systems, 2019 (https://doi.org/10.1016/j.future.2018.09.038)*
    c. *Jin C. et al., DiSC: A Distributed Single-Linkage Hierarchical Clustering Algorithm using MapReduce, International Workshop on Data Intensive Computing in the Clouds (DataCloud), November 2013 (pdf: cjinDataCloud13.pdf (northwestern.edu))*
6. Can you expand on why you choose the average linkage function for the examples presented in the paper? Would other linkage functions work as well both in term of computational efficiency and subjective accuracy?
7.
    a. Can you expand on why Pearson's correlation coefficient (1-R) was used as the choice of metric?
    b. This metric will ignore linear transforms (additive and multiplicative shifts in the data), is this a desired feature?
    c. Line 287: Why is the normalization of the species necessary since 1-R is already doing a normalization? Is this step really necessary or I am missing something?
8. The results shown do not take advantage of the hierarchical clustering analysis. Instead, an arbitrary number of clusters is chosen (50 and 100 in the examples). More efficient computation could thus be potentially obtained simply using a highly optimized version of K-means if the number of clusters does not need to be varied. That is, why do we need hierarchical clustering, could other non-hierarchical clustering methods work as well?
9. Not many details were provided on the data pre-processing for stations. Please expand for better reproducibility of the results.

a. How was missing data handled in measurement data? How the algorithm could be used on the COVID-19 year (2020) or on the 209 stations not used in the analysis?
b. Were there any techniques used for quality assurance and quality control of the data, and in particular to remove outliers?
c. Line 286 gives 366,427 data points.
    i. How to arrive at this number? 51 stations x 24 hours x 365 days = 446,760.
    ii. How sensitive the results are to the subsampling of the data? Clearly, it will be costly to perform a sensitivity analysis with the current version of the algorithm on the full dataset and this is why further speed-up of the algorithm would be desirable.
d. Providing the pre-processed subset of NAPS data used in this analysis in open source data repository (and the complete code to automatically generate it) would help to improve the reproducibility of the results.

Technical (line-by-line):

L1: Although it is rather subjective, I would not call a dataset with between 10^5 and 10^6 data points a "very large dataset". For example, by comparing to Table 1 of *Woodley et al. 2019*, we see examples of other works with data sets of size between 10^4 to 10^9 while the work of the referenced paper uses a dataset of size 10^11. To be a bit more precise, I suggest changing the title to "An Implementation of Hierarchical Clustering Analysis on High-Performance Computing Platforms for Large Air Quality Datasets"

L10: "Modern implementations of the algorithm have O(n^2log(n)) computational complexity and memory O(n^2) usage." That statement is not true. For example, even Defays' 1977 CLINK algorithm for complete-linkage hierarchical clustering has a complexity of O(n^2) and O(n^2) memory. Eppstein's 2000 fast hierarchical clustering can achieve O(n^2 log^2 n) time complexity and O(n) space or alternatively O(n^2) time and O(n^2) space. Accelerated HDBCAN* from McInnes 2017 has a time complexity of O(n log n), but it is a slightly different hierarchical clustering approach which excludes some data points as noise/outliers.

L53: Provide citation for hierarchical clustering

L59-60: Provide citation for PMF and K-means

L63: "memory required scales as the number of input data squared" -> only if all pair of distance are pre-computed, alternative implementation can trade-off memory requirements and computational complexity, see comment for line 10.

L65-73: "worse, the computation time scales as the number of input data cubed" -> only for a naïve implementation of the algorithm. Consider removing the example of time and space requirements for the naïve O(n^3) implementation as it should not be considered in practice for larger problems.

L84: "To solve these problems, we turn to parallel computing." -> Although parallel computing on a high-performance computing platform is a sensible solution to squeeze out further performance, it would have been preferable to first seek to optimize the clustering algorithms themselves with appropriate data structures and constraints or to select a more efficient hierarchical clustering alternative such as HDBSCAN*. Only after this optimization is performed, we should consider parallel computing to push further the performance. I would rephrase.

L113: “HC” -> “High-Performance Computing (HPC)”

L114 and 494: “To our knowledge this is the first time a hierarchical clustering analysis has been performed on such large datasets”. -> might be true for air quality data, but certainly not true in general. Although only references to clustering using DBSCAN and K-means are known to have been pushed to the extreme (up to 10^11 data points in Woodley et al. 2019), the accelerated HBDSCAN* hierarchical clustering of McInnes et al. 2017 has been tested for datasets of 200,000 data points (Python implementation running in less than a minute for 50-dimensional data).

L124: “The data could be any sort of 2-dimensional data” -> This statement is confusing as it refers to the fact that the data can be stored in a 2D-array. Usually, the number rows will be the data size (number of samples) whereas the number of columns is the number of dimensions of the data. Please rephrase to clarify that is not the data that is 2-dimensional but that the array in which the data is stored has rows and columns. It could be clarified here that some data can be vectorized to combine several “dimensions” (time, spatial, different variables, etc.).

L137-195: I don’t find this example necessary, interesting, or useful. I would replace the example with a more general discussion on hierarchical clustering, the choice of metrics, linkage functions and its different applications in air quality (see specific comments 6-7).

L231: “operated by Shared Services Canada” -> that might not be clear to what is refers to a general audience, might want to add “the department responsible for providing computing infrastructure for the Government of Canada”.

L298: “which would be prohibitive using (…) K-means” -> Efficient implementations of K-means scale generally better than hierarchical clustering, so this is not a fair comparison. For example, see Woodley et al. 2019, Table 1 and McInnes et al., 2017, Figure 6.

L304: Isn’t the wall clock limit 6 hours on the current Shared Services Canada infrastructure?

L451: “The authors would like to emphasize that there may be many considerations required to obtain the best performance on any given high-performance computing cluster.” -> could you enumerate a few of them?