Reviewer #1

We would like to thank the referee for a very thoughtful and detailed review of our manuscript. Incorporation of the reviewer's suggestions has led to a much improved manuscript. Below we provide a point-by-point response to the reviewer's comments and how we have addressed them in the revised manuscript.

[Comment]: The authors do not provide sufficient information to reproduce the results of their work. What version of pnetcdf was used and and what versions of supporting software such as MPI and Lustre were used? What version of the CMAQ models was used and is the pseudo-code used to perform the experiments publicly available? No reference was provided for either netcdf or pnetcdf libraries.

[Response]: We agree with the reviewer's point of view that there is insufficient software information describing our experiments in the paper. Here is additional information. These are the software packages we used on Edison and Kraken, respectively and will be added to the manuscript in section 4.1:

Edison: cray-mpich/7.0.4, cray-netcdf/4.3.0, parallel-netcdf/1.3.1, lustre: 2.5.0

Kraken: Cray MPT 5.3.5, netcdf 3.6.3, pnetcdf 1.2.0, lustre 2.5.0

We used CMAQ version 5.0.2 for the one day simulation and we have constructed a small scale test code based on the data flow of regular CMAQ: a cycle of input data, data calculation, and output data. In the manuscript, section 4.2, we will provide the pseudo code of this small scale test code. This small scale test code can be obtained upon request.

[Comment]: The data aggregation technique that the authors refer to as a novel new approach is in essence the same technique that pnetcdf, MPIIO and Lustre apply at lower levels in the software stack so the question becomes - why does doing this aggregation at a higher level in the software stack work better than it does at a lower level?

[Response]: Based on the pnetcdf documentation, data aggregation can be done on sequences of small requests with non-blocking I/O. Our data layout is column, row, layer, variable while the spatial domain is decomposed. The amount of data in each processor is not "small" even with smallest domain described in this article and the number of processors is over 128. Hence aggregate data in the software level with respect to the spatial domain, in particular along the column dimension, makes more sense: not only the output data chunk is larger but also the contiguousness of the data.

[Comment]: Total time to perform parallel IO includes both communication time and IO time but the authors make no attempt to separate these factors. In section 5.3 the authors claim that increasing the data size on the processor which is responsible for I/O will translate into higher I/O rates. If this is the case why not just use the original serial approach in which one I/O processor is responsible for all of the data?

[Response]: In our approach, the process consists of two parts: data aggregation (communication time) and data write to disk (I/O time) by a subset of processors. So the sum of the communication time and the I/O time is the true representation of time to move data in each processor to the disk. This timing can be used to compare directly with the parallel I/O approach implemented with pnetCDF which sends data to the disk collectively without any communication.

Figure 11 illustrates the notion of larger chunk has a better I/O rate, we argue that the benefit of data aggregation as a basis for the success of our approach. We won't go to the extreme to have one processor to do the I/O (this is the technique being employed in the current CMAQ model) but rather try to strike for a balance by utilizing parallel I/O technology. Indeed, in this article, we have demonstrated our approach out performs the serial approach which is currently used in the CMAQ model, substantially.

[Comment]: The pnetcdf library contains a number of IO interfaces, collective vs independent and synchronous vs asynchronous, and supports several techniques for data aggregation. The authors do not indicate which pnetcdf interfaces they are using, or whether they experimented with others.

[Response]: In this work, we are using only collective parallel netcdf API. We can take a look at other types of interfaces that pnetcdf provides as future work.

[Comment]: Figures 3-8 are too small and too busy to convey any meaningful information, perhaps tables would be better?

[Response]: We agree that Figures 3 - 8 are too small and too busy however, we believe a 3D plot is the best way to convey performance information with respect to two different variables: stripe count and stripe size, at the same time. In the original plots, positive (solid color) and negative (checkered pattern) bars which denotes the performance comparison of a specific technique, can be clearly distinguished in each scenario. After the publisher's typesetting process of the entire paper, such clear distinction is gone and that is unfortunate. We have re-plotted these graphs with only two colours: red which denotes positive value and blue which denotes negative value. We have replaced Figures 3 – 8 with these new ones in the manuscript.

Reviewer #2

We would like to thank the referee for a very thoughtful and detailed review of our manuscript. Incorporation of the reviewer's suggestions has led to a much improved manuscript. Below we provide a point-by-point response to the reviewer's comments and how we have addressed them in the revised manuscript.

[Comment]: First, altering IOAPI and PARIO to do true parallel I/O is a necessary engineering effort, but it not novel in 2014. (Authors do not spend a lot of time on this point, so I think they understand and would agree with me).

[Response]: We agree with the reviewer that we had a bad choice of word. We have replaced novel with "an application level data aggregation approach" in the manuscript. As a matter of fact, re-engineering PARIO to make it to perform true parallel I/O operation is the only way to overcome I/O bottleneck in the air quality model, CMAQ.

[Comment]: second, application level aggregation is not novel: in climate/weather it has been done/published in GCRM (a cloud resolving model) and PIO (for climate simulations). The approach described here, where the aggregation is done according to MPI processor topology, sounds a tiny bit novel, but does not get a lot of text.

[Response]: We agree with the reviewer that application level aggregation is not novel. Indeed, we were taught in MPI classes to aggregate data for message passing when it is possible. Performing true parallel I/O through pnetCDF and making use of data aggregation in the application level to increase pnetCDF performance in the air quality model, CMAQ is our unique contribution.

Thanks to the reviewer bringing our attention to two additional recent papers:

* Bruce Palmera, Annette Koontza, Karen Schuchardta, Ross Heikesb, David Randallb, "Efficient data IO for a Parallel Global Cloud Resolving Model", Environmental Modelling & Software, Volume 26, Issue 12, December 2011, Pages 1725–1735

* X. M. Huang, W. C. Wang, H. H. Fu, G. W. Yang, B. Wang, and C. Zhang, "A fast input/output library for high-resolution climate models", Geosci. Model Dev., 7, 93–103, 2014

The former paper stresses the importance of utilizing data aggregation to achieve high bandwidth and our work is also based on this fact. The latter paper focuses on having an extra set of processors dedicated for I/O operation to achieve overlapping of computational work and I/O task. We have compared the overall model run time when we considered those extra processors as part of the computational resource versus only dedicated for I/O process. Treating those extra processor for I/O only did poorly since I/O bound (I/O time with respect to the overall model run time is about 15 - 25%).

[Comment]: I am not sure how much tuning the authors did after adopting parallel-netCDF. Evalutaions suggest stripe size and stripe count were the two knobs chosen. As was demonstrated in Behzad and Lu's 2013 SC paper (http://dl.acm.org/citation.cfm?id=2503210.2503278), tuning the I/O stack on machines like Edison and Kraken can have a 7-fold impact on performance. Now it must be said that a further point of the 2013 paper was that it's a burden to expose these detailed tuning approaches to application scientists, so it's ok if the authors only explored those two settings. I just want it explicilty mentioned.

[Response]: We know there are two parameters, "stripe count' and "stripe size", that users can adjust for performance purposes. Our intention was to try to obtain an "optimal" setting with respect to PE configuration and model domain size. In general, a user (scientist) does not know much about tuning I/O stack to obtain better performance. Here we try to provide some easy understandable way to improve I/O performance in scientific applications.

Again thank you so much for bringing our attention to Lu's paper which provides lots of useful information. It will be great if Lu's work could be turned into some simple tools so scientists can use it to achieve optimal I/O performance for their model on different platforms.

[Comment]: Is the simplified CMAQ model used in these experiments available for others to use, or will it be made avaliable? The I/O community is a voracious consumer of such I/O kernels: if you publish the one you have created for CMAQ, then a small battalion of grad students and I/O researchers will add it to their list of kernels they consider when evaluating new i/o strategies and designing new i/o subsystems.

[Response]: We are more than happy to share the simplified CMAQ model, we called it "pseudo" code, with you. Basically this striped down version of CMAQ looks like this (this pseudo code has been added to the manuscript):

```
DO I = 1, 3
    Read in data
    Perform numerical calculation (artificial work)
    Output result
END DO
```

[Comment]: What aspects of the I/O stack made pnetcdf under-perform? Are there lessons to be learned from CMAQ that could be applied to the I/O stack (pnetcdf, MPI-IO, and Lustre layers) that would benefit all applications on Edison and Kraken?

[Response]: In this paper we did not attempt (in fact we don't have such knowledge) to identify which aspect of the I/O stack made pnetCDF under-perform. It is known that pnetCDF and MPI-IO have aggregation capability with respect to I/O requests or messages. Our approach is to apply data

aggregation on the application level. It will be difficult to adopt this approach to the I/O stack since this approach based on the knowledge of spatial domain decomposition which the I/O stack does not have. In this paper, we have demonstrated that scientists can adopt this application level data aggregation technique in an effective and straightforward manner.