

View Letter

[Close](#)

Date: Apr 07, 2017
To: "Stefan Schmid" stefan.schmid@tu-berlin.de
From: "Theoretical Computer Science" eesserver@eesmail.elsevier.com
Reply To: "Theoretical Computer Science" tcseditorialoffice@elsevier.com
Subject: Review on TCS-D-16-00395

Manuscript Number: TCS-D-16-00395
Manuscript Title: Data Locality and Replica Aware Virtual Cluster Embeddings
Author(s): Carlo Fuerst; Maciek Pacut; Stefan Schmid
Submitted to: Theoretical Computer Science

Dear Professor Stefan Schmid,

Referees have evaluated your paper mentioned above. Their comments are below. On the basis of these comments, the member of the Editorial Board handling your paper has decided that a major revision is needed before your paper can be considered for publication. I would appreciate it if you could revise your paper within fifty six days of this email.

I look forward to receiving the revised version of your paper. You should explain how and where the reviewers' comments have been addressed by your changes to the text (overview of changes). Should you disagree with some of these comments, please explain why.

To upload the source file (LaTeX, or Word) of the revised paper and the overview of changes, please log-in to the system at <https://ees.elsevier.com/tcs/> with username: Your username is: *****

If you need to retrieve password details, please go: to http://ees.elsevier.com/TCS/automail_query.asp selecting the 'Author' button.

When submitting your revised paper, we ask that you include the following items:

Response to Reviewers (mandatory)

This should be a separate file labeled "Response to Reviewers" that carefully addresses, point-by-point, the issues raised in the comments appended below. You should also include a suitable rebuttal to any specific request for change that you have not made. Mention the page, paragraph, and line number of any revisions that are made.

Manuscript and Figure Source Files (mandatory)

We cannot accommodate PDF manuscript files for production purposes. We also ask that when submitting your revision you follow the journal formatting guidelines. Figures and tables may be embedded within the source file for the submission as long as they are of sufficient resolution for Production. For any figure that cannot be embedded within the source file (such as *.PSD Photoshop files), the original figure needs to be uploaded separately. Refer to the Guide for Authors for additional information.
<http://www.elsevier.com/journals/theoretical-computer-science/0304-3975/guide-for-authors>

Highlights (Optional)

Highlights consist of a short collection of bullet points that convey the core findings of the article and should be submitted in a separate file in the online submission system. Please use 'Highlights' in the file name and include 3 to 5 bullet points (maximum 85 characters, including spaces, per bullet point). See the following website for more information
<http://www.elsevier.com/highlights>

Please note that this journal offers a new, free service called AudioSlides: brief, webcast-style presentations that are shown next to published articles on ScienceDirect (see also <http://www.elsevier.com/audioslides>). If your paper is accepted for publication, you will automatically receive an invitation to create an AudioSlides presentation.

Theoretical Computer Science features the Interactive Plot Viewer, see:
<http://www.elsevier.com/interactiveplots>. Interactive Plots provide easy access to the data behind plots. To include one with your article, please prepare a .csv file with your plot data and test it online at

<http://authortools.elsevier.com/interactiveplots/verification> before submission as supplementary material. PLEASE NOTE: Theoretical Computer Science would like to enrich online articles by displaying interactive figures that help the reader to visualize and explore your research results. For this purpose, we would like to invite you to upload figures in the MATLAB .FIG file format as supplementary material to our online submission system. Elsevier will generate interactive figures from these files and include them with the online article on SciVerse ScienceDirect. If you wish, you can submit .FIG files along with your revised submission.

Kind regards,

Paul Spirakis
Editor-in-Chief
Theoretical Computer Science

Comments:

Reviewer #2: The submission initiates the algorithmic study of data locality aware virtual cluster embeddings on datacenter topologies. Namely, it formalizes the virtual cluster embedding as an optimization problem, and extends it with data locality, an aspect that was ignored by existing algorithms, even though it is crucial to the performance of cloud applications such as MapReduce.

Formally, the authors model the problem as follows:

- * The physical network (also called substrate network) forms a tree, whose leaves are servers and internal nodes are routers. Each server (or leaf) s may host a number of virtual machines (or "nodes") no larger than $\text{cap}(s)$. The edges may have [bandwidth] capacities, denoted $\text{cap}(e)$, and all have unit lengths. While the assumption that the physical network forms a tree may seem a big restriction from a theoretical standpoint, this conforms with real applications.
- * The data to be processed is distributed over the physical network; the data locations are given and are not subject to optimization. The data consists of $\langle \tau \rangle$ chunk types, denoted $c_1, \dots, c_{\langle \tau \rangle}$, where each chunk type c_i has $r_i = 1$ replicas, stored at different servers. However, a single server may store an arbitrary number of replicas of different chunk types. Lastly, it is important to note that for each chunk type, only one replica has to be chosen for processing.
- * The virtual network is characterized by the number of nodes (virtual servers) to be hosted on the physical servers, subject to the servers capacities.
- * The algorithm is to (i) select the servers to host the nodes, (ii) select one replica per chunk type to be processed, and (iii) assign each selected replica to a node; in the last part there is an important assumption that the nodes load is to be balanced, i.e., that all of them are assigned an equal number of replicas for processing. (So, in particular, the number of chunk types is a multiple of the number of nodes.)
- * The goal is to minimize the "resource footprint", which is a weighted sum of two types of connection costs: the first one, multiplied by b_1 , is total distance between each selected replica and the node processing it (i.e., the physical servers hosting the replica and hosting the node), and the other, multiplied by b_2 , is the total distance between every pair of nodes (again, in actuality: servers that host them).

This describes the problem at its most general, but the authors consider 5 possible independent restrictions, which means that overall they consider $2^5=32$ (potentially) different settings. The *lack* of such restrictions are:

- replica selection (RS), i.e., redundancy of replicas of some chunk type, which necessitates selection of a single replica per each such type
- multiple chunk assignment (MA), i.e., the necessity to assign multiple replicas to each node when there are more chunk types than nodes
- flexible node placement (FP), i.e., the ability to embed/host the nodes at arbitrary servers (as opposed to a scenario where the nodes are already placed)
- node interconnect (NI): $b_2 > 0$, i.e., the cost of pairwise connections of the

nodes does contribute to the objective

- bandwidth constraints (BW): $\text{cap}(e)$ is bounded (as opposed to infinite) at least for some edges of the substrate network.

The article gives a complete classification of all 32 variants: for each variant it either provides a polynomial time algorithm to solve it or a proof of NP-hardness via reduction from the 3D-Matching problem. As one might suspect, the issue with such kind of an article is that none of these multiple results seems particularly interesting. Indeed, all the algorithms are standard applications of either flow techniques (matchings in particular) or dynamic programming, and the reductions from 3D-Matching are straightforward. The only possible exception is Section 5, which gives two reductions that are more involved, as they aim at proving NP-hardness of fairly restricted variants of the problem, namely $\text{FP}+\text{RS}(2)+\text{MA}(4)$ and $\text{RS}(2)+\text{FP}+\text{NI}+\text{BW}$, where the two letter acronyms stand for aforementioned lack of restrictions and $\text{RS}(n)$ means that even though replica selection is required, there are at most n replicas per chunk type. Unfortunately, this part of the article is rather poorly written: Section 5.1 (which gives the first of the two reductions) in particular looks as if no one read it in its current form.

Still, even Section 5 is not too technically involved, so it seems that the largest contribution of the article is introducing a sensible model. I am unable to foresee how much follow-up work it will inspire, given that in principle the authors have completely resolved all of its variants.

Thus, I would rather refrain from suggesting accepting or rejecting the submission, and merely point out that a major revision would be required before the draft could be accepted; specific comments follow at the end.

QUESTIONS / FUTURE WORK that the authors or others might consider interesting:

- 1) All edge lengths in the physical network are unit. What would happen if they were not? Presumably the flow-based algorithms would still work, but perhaps something changes in the dynamic programming approach? I.e., perhaps some of them become pseudopolynomial?
- 2) Perhaps some relaxation of the requirement that replicas are distributed uniformly between nodes makes sense?
- 3) As some of the problems are NP-hard, and moreover of practical interest, considering approximation algorithms seems a worthwhile direction.
- 3) I suppose that one might argue that in the applications the underlying physical network is (more or less) fixed. In such case, when the instance specifies only the data distribution, number of nodes to be embedded, and the b_1 and b_2 coefficients, would some of the previously NP-hard problems become tractable?

SPECIFIC COMMENTS:

SECTION 5 in general: You keep changing the names for instances of the two problems. Be consistent!

SECTION 5.1:

Let me reiterate that I firmly believe that this section requires a major revision. Even though the argument should be easy to follow, this is impossible in the current version. When you do, please provide the calculations. And check if the threshold value is what it should be, because it appears to me that it is slightly off.

- Observation 2, points 1 and 2: some chunk types are unique to either of the two subtrees, so clearly, they do not have exactly one replica in the other; please reformulate.
- "Reduction" (page 26):
 - * In point 2: (1) the n nodes should be placed each in a distinct leaf, not all in the same one, as your writing suggests; (2) the 3D-Matching solution selects a triple, not a Triple Gadget --- the latter corresponds to the former

- * In point 3, you refer to "Element Subtree" but, strictly speaking, you have never defined it.
- "Proof of correctness" (right after the "reduction"):
 - * Correctness of what?
 - * (1) The two costs you introduce presumably correspond to something, but you never explain that. (2) Moreover, t is undefined, and should probably not be used at all, since on page 23 you state that t will be used for a triple from T . (3) Similarly, it is only in Lemma 6 that you specify the domain for u .
- Lemma 7: Just like the "Element Subtree", "Unique Subtree" was not defined. And neither was σ .
- Lemma 8: ℓ was not defined. I can only assume it is *not* the ℓ from the previous Lemma, since that one was to prove that $\ell=0$.

SECTION 5.2:

- I believe that you should provide a figure with the tree, similar to the one in Section 5.1
- Element Gadget: you write that it has $4*(\deg(e)-1)+1$ leaves, but it seems that this was copied verbatim from Section 5.1 and that the right number is $\deg(e)$
- The threshold value formula (bottom of page 29):
 - * Could you please increase the font size?
 - * It appears that every single occurrence of $|V|$ in the formula should be replaced by n
 - * Shouldn't the summand for e in the sum on the right end of the first line be $\binom{\deg(e)-1}{2}$ rather than $\deg(e)-1$?
 - * Are you sure of the coefficient (i.e., their lack) next to the terms in the last two lines?
- Lemma 10:
 - * last line of statement: n should be replaced by $|V|$
 - * could you explain in detail how you arrive at the final inequalities?
- Lemma 12 and its proof: It appears that you mean the "Element Gadget", even though you are referring to the "Triple Gadget"

MINOR COMMENTS:

throughout article: I think you never define/explain the term "uplink"

p3l54-55: "constituting parts" --> "constituent parts"

p4l14-17: The sentence "The standard datacenter topologies today are (multi-rooted) fat-tree resp. Clos topologies [2, 21], hierarchical networks recursively made of sub-trees at each level; servers are located at the tree leaves." sounds gibberish --- please reformulate.

p8l48: remove the first 'a' from "can now be computed a from a solution to the"

p8l40: insert space in "withthe"

p11l54: change "we can computed" to either "can be computed" or "we can compute"

p16l26-27: was "communicated" supposed to be "connected"?

p16l44: remove the repetition of "formula"

p19l23 (end of the paragraph): all elements of the *union* of X, Y, Z have to be covered (once), not their Cartesian product.

NP-hardness proofs: I think you should use a single symbol, perhaps τ , for the threshold; $\$Th\$$ rather confusingly looks like two disconnected symbols.

p22-23: Lemma 4 and Theorem 5 --- since you do the easy calculation of the cost a specified solution in proof of Theorem 5, I believe you should also properly explain the argument in the proof of Lemma 4: it is straightforward but definitely no easier than the mere calculations in Theorem 5, as the claim is that that specific solution minimizes the cost.

Reviewer #3: The paper studies a well-motivated problem related to virtualized data centers. It formulates and solves several versions pertaining to the problem of data locality aware virtual cluster embeddings. The model (although a bit complicated) does seem to capture most of the problems related to data aware embeddings. The authors have two layers, the substrate network and the virtual network, and the former needs to get mapped to the latter. They consider several aspects (5, to be exact) namely replica selection, multiple assignment, flexible placement, node interconnect, and bandwidth capacities.

The solutions/algorithms are fairly simple, and I would not claim that the technical content of the paper is substantial. However, the novelty of the problem and the arguably sound formulation merit a publication.

Given the number of problems (32), I would have liked the authors to perhaps create a table; it is hard for a reviewer to keep track of which problems can be solved efficiently and which are hard. Also, the optimization function - the footprint, is just stated and not discussed at all. Perhaps this is a well-agreed upon function in this field (which is new to me). Why is this the idea function to optimize?

Lastly, for the problems that are hard, I would also like the authors to mention how hard they are to approximate, or whether one can get a nice approximate solution.

For further assistance, please visit our customer support site at <http://help.elsevier.com/app/answers/list/p/7923>
Here you can search for solutions on a range of topics, find answers to frequently asked questions and learn more about EES via interactive tutorials. You will also find our 24/7 support contact details should you need any further assistance from one of our customer support representatives.

Close