

A Distributed Algorithm for Resource Clustering in Large Scale Platforms

Olivier Beaumont, Nicolas Bonichon, Philippe Duchon,
Lionel Eyraud-Dubois, Hubert Larchevêque

Universit de Bordeaux, INRIA Bordeaux Sud-Ouest, Laboratoire Bordelais de
Recherche en Informatique

Abstract. We consider the resource clustering problem in large scale distributed platforms, such as BOINC, WCG or Folding@home. In this context, applications mostly consist in a huge set of independent tasks, with the additional constraint that each task should be executed on a single computing resource. We aim at removing this last constraint, by allowing a task to be executed on a (small) set of resources. Indeed, for problems involving large data sets, very few resources may be able to store the data associated to a task, and therefore may be able to participate in the computations. Our goal is to propose a distributed algorithm for a large set of resources that enables to build clusters, each of which will be responsible for processing a task and storing associated data. From an algorithmic point of view, this corresponds to a bin covering problem with an additional distance constraint. Each resource is associated to a weight (its capacity) and a position in a metric space (its location, based on network coordinates such as those obtained with Vivaldi), and the aim is to build a maximal number of clusters, such that the aggregated power of each cluster (the sum of the weights of its resources) is large enough and such that the distance between two resources belonging to the same cluster is kept small (in order to minimize intra-cluster communication latencies). In this paper, we describe a generic 2-phases algorithm, based on resource augmentation and whose approximation ratio is $1/3$. We also propose a distributed version of this algorithm when the metric space is \mathbb{Q}^D (for a small value of D) and the L_∞ norm is used to define distances. This algorithm takes $O((4^D) \log^2 n)$ rounds and $O((4^D)n \log n)$ messages both in expectation and with high probability, where n is the total number of hosts.

Introduction The past few years have seen the emergence of a new type of high performance computing platforms. These highly distributed platforms, such as BOINC [3], Folding@home [1] and WCG [2] are characterized by their high aggregate computing power, their heterogeneity in terms of resource performances and by the dynamism of their topology, due to node arrivals and departures. Until now, all the applications running on these platforms (Seti@home [4], Folding@home [1],...) consist in a huge number of independent tasks, and all data necessary to process a task must be stored locally in the processing node. The

only data exchanges take place between the master node and the slaves, what strongly limits the set of applications that can be performed on these platforms.

Two kind of applications fit in this model. The first one consists in those, such as Seti@home, where a huge set of data can be arbitrarily split into arbitrarily small amounts of data that can be processed independently on participating nodes. The second one corresponds to Monte-Carlo simulations. In this case, all slaves work on the same data, except a few parameters that drive the simulation. This is for instance the model corresponding to Folding@home.

In this paper, our aim is to extend this last set of applications. More precisely, we consider the case where the data set needed to perform a task is possibly too large to be stored at a single node. This situation is very likely to occur in large scale platforms based on the aggregation of strongly heterogeneous resources. In this case, both processing and storage must be distributed on a small set of nodes that will collaborate to perform the task. The nodes involved in the cluster should have an aggregate capacity (memory, processing power,...) higher than a given threshold, and they should be close enough (the latencies between those nodes should be small) in order to avoid high communication latencies.

In this context, the aim is the following: given a set of weighted items (the weights are the storage capacity of each node), and a metric (based on latencies), to create a maximum number of groups so that the maximal latency between two hosts inside any group is lower than a given threshold, and so that the total storage capacity of any group is greater than a given storage threshold. This problem turns out to be difficult, even if one node knows the whole topology (*i.e.* the available memory at each node and the latency between each pair of nodes). Indeed, even without the distance constraint, this problem is equivalent to the classical NP-complete bin covering problem [6]. Similarly, if the constraint about storage capacity is removed, but the distance constraint is kept, the problem is equivalent to the NP-Complete disk cover problem [12].

Results Due to the lack of space, we refer the interested reader to the companion research report [7] where all the proofs and algorithms are provided in details.

In this paper, we propose a generic greedy 2-phases algorithm, based on resource augmentation and whose approximation ratio is $\frac{1}{3}$. More precisely, we use resource augmentation in the following way. We compare the number of clusters (or bins) created by our algorithm with diameter constraint d to the optimal number of bins that could be created with distance d_{\max} , where $d > d_{\max}$. This resource augmentation is both efficient and realistic. Indeed, if the aggregated memory of the cluster should be larger than a given threshold in order to be able to process the task, the threshold on the maximal latency between two nodes belonging to the same cluster is weaker, and mostly states that nodes belonging to the same cluster should not be too far from each other. Moreover, this resource augmentation enables to prove a constant approximation ratio ($\frac{1}{3}$) whereas approximation ratio without resource augmentation would be exponential in the dimension of the metric space .

The basic structure of this 2-phases greedy algorithm is the following:

Phase 1 Greedily create bins of diameter at most d_{\max}

Phase 2 Greedily create bins of diameter at most $3d_{\max}$.

Theorem 1. *The 2-phases greedy algorithm provides a $\frac{1}{3}$ -approximation algorithm of max_DCBC problem, using a resource augmentation of factor $2 + \frac{d}{d_{\max}}$ on the maximal diameter of a bin.*

An extension of the generic 2-phases greedy algorithm with approximation ratio $\frac{2}{5}$ with the same resource augmentation is also possible. These results are to be compared to some classical results for bin covering in centralized environment without the distance constraint. In this (much easier) context, a *PTAAS* (polynomial-time asymptotic approximation scheme) has been proposed for bin covering [10], *i.e.* algorithms A_ϵ such that for any $\epsilon > 0$, A_ϵ can perform, in a polynomial time, a $(1 - \epsilon)$ -approximation of the optimal when the number of bins tends towards the infinite. Many other algorithms have been proposed for bin covering, such as [6], that provides algorithms with approximation ratio of $\frac{2}{3}$ or $\frac{3}{4}$, still in a centralized environment.

This paper is a follow-up to [8], where the case of a one-dimensional metric space is considered. In order to estimate the positions of the nodes involved in the large scale platform, we rely on mechanisms such as Vivaldi [9,11] that associate to each node a set of coordinates in a low dimension metric space, so that the distance between two points approximates the latency between corresponding hosts. Here, we consider the case where resource locations are given by their coordinates in a metric space with arbitrary dimension. Moreover, in a large scale dynamic environment such as BOINC, where nodes connect and disconnect with a high churn, it is unrealistic to assume that a node knows all platform characteristics. Therefore, in order to build the clusters, we need to rely on fully distributed schemes, where a node makes the decision to join a cluster based on its position, its weight, and the weights and positions of its neighbor nodes. Therefore, we also propose a distributed version of this algorithm when the metric space is \mathbb{Q}^D (for a small value of D) and the infinity norm is used to define distances.

Theorem 2. *There exists an algorithm, running in parallel for 4^D disjoint intervals, that uses $O(4^D n \log n)$ messages and, in a synchronous execution model where each message takes unit time, $O(4^D \log^2 n)$ rounds, both in expectation and with high probability, where n is the total number of hosts.*

Moreover, we claim that this algorithm can be used in practice, since its implementation only relies on classical distributed data structures, such as skip graphs [5].

In future works, we plan to adapt the algorithm to the case where several characteristics must be satisfied simultaneously (for instance, a task may require both a large aggregated memory and a large disk storage capacity). Another interesting work is to compare the performances of the distributed algorithm we propose with the gossip-based approach. Gossip-based algorithm complexities are usually very difficult to establish, but these algorithms have been proved very efficient to exploit locality [?],[14]. At last, we need to adapt the algorithm to the

case where the metric space is not \mathbb{Q}^D . Indeed, if network coordinates systems based on landmarks [13] used \mathbb{Q}^D (for values of D of order 10) as underlying metric space, more recent coordinate systems, such as Vivladi [9] rely on much more sophisticated metric spaces (but based on 3 coordinates only).

References

1. Folding@home. <http://folding.stanford.edu/>.
2. World community grid. <http://www.worldcommunitygrid.org>.
3. David P. Anderson. Boinc: A system for public-resource computing and storage. In *GRID '04: Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing*, pages 4–10, Washington, DC, USA, 2004. IEEE Computer Society.
4. David P. Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, and Dan Werthimer. Seti@home: an experiment in public-resource computing. *Commun. ACM*, 45(11):56–61, 2002.
5. J. Aspnes and G. Shah. Skip graphs. *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 384–393, 2003.
6. S.F. Assmann, D.S. Johnson, D.J. Kleitman, and J.Y.T. Leung. On a dual version of the one-dimensional bin packing problem. *Journal of algorithms*, 5(4):502–525, 1984.
7. O. Beaumont, N. Bonichon, P. Duchon, L. Eyraud-Dubois, and H. Larcheveque. A distributed algorithm for resource clustering in large scale platforms. Research report, INRIA Bordeaux Sud-Ouest, France, October 2008.
8. O. Beaumont, N. Bonichon, P. Duchon, and H. Larcheveque. Distributed approximation algorithm for resource clustering. In LNCS Series Springer, editor, *Proceedings of SIROCCO'08*, volume 5058, pages 61–73, 2008.
9. R. Cox, F. Dabek, F. Kaashoek, J. Li, and R. Morris. Practical, distributed network coordinates. *ACM SIGCOMM Computer Communication Review*, 34(1):113–118, 2004.
10. J. Csirik, D.S. Johnson, and C. Kenyon. Better approximation algorithms for bin covering. *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 557–566, 2001.
11. F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: a decentralized network coordinate system. *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 15–26, 2004.
12. M. Franceschetti, M. Cook, and J. Bruck. A geometric theorem for approximate disk covering algorithms, 2001.
13. T.S.E. Ng and Hui Zhang. Predicting internet network distance with coordinates-based approaches. In IEEE, editor, *Proceedings of INFOCOM'02*, pages 170–179, 2002.
14. S. Voulgaris, D. Gavidia, and M. van Steen. CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays. *Journal of Network and Systems Management*, 13(2):197–217, 2005.