Use cases for a new scheduling algorithm in OpenStack

We refer in practice to the large number of Data Centers which support research or Public Administration (PA). They are characterized by the fact that:

- computing and storage resources are bought in general on a yearly basis from a budget which is previously agreed with the stakeholders (e.g. research teams or various organizations in the PA)
- the amount of required resources is reviewed regularly (usually every year)
- stakeholder verifies the incoming requests based on the past usage of existing resources

When new hardware ((in terms of cpu, ram, storage, etc) is acquired, the Data Center has to decide how to partition these resources among the stakeholder teams. The partition policy is defined in terms of shares of usage (i.e. the assigned percentage of the Data Center CPU) and it is enforced for a well defined period (typically one year, when new budgets and allocations are redefined). The share is calculated according to the results of scientific evaluations or the available level of financing the teams have put in the Data Center.
A contract is then made with each team agreeing on the amount of IaaS computing resources they can use, let's say per year. With the current OpenStack scheduling solution each user group gets by the administrators a fixed quota of the existing computing resources which corresponds to the agreed amount. Past experience is showing that using this simplistic algorithm bring to a very low global efficiency in resource usage (down to as low as 50%) and then to increased costs for the users.
We observed that the main reason of the missing saturation of resources is due by the discontinuity in the team's activities. Typically each team requires virtual machines of three very different durations in relations to different types of activities they need to carry out in the Data Centers:

1. Type1 unlimited duration time (UVM)
2. Type2 limited, but planned for a well defined date and for a medium-long (typically from 1 to 3 weeks) duration time (LVM)
3. Type3 limited but with short (typically from few hours to 1 day) duration time (SVM)

Activities of Type1 refer in general to services for hosting WEB sites or any data access services (e.g. FTP service, Metadata service or AAI services) which must be "forever" available.

Activities of Type2 refer to planned synchronous activities related to data analysis and visualization. People sit in front of a screen and interact with data using a cluster of VMs to produce quickly and interactively results that can be immediately analyzed to decide the next step to be done. In general these occur before important deadlines as those for the submission of a paper to a Conference in Science or the release of important results in other domains when scientists or managers need to go through sensible data to extract relevant information for their decisions or reports. This requirement is quite similar to the one addressed by the Blazar project (https://wiki.openstack.org/wiki/Blazar, previously known as climate) that was originally scoped for calendar based scheduling (such as in an HPC environment where you want to be sure to get 500 VMs from time X to time Y or reserving resources for a school class).

Finally the activities of Type3 embrace an asynchronous model and refer to the serial systematic and repetitive sequential simulations or analysis of a large number of different data objects or data files. This is done usually by activating a large number of different analysis requests using the same application and virtual environment but changing the input and output objects or files.
The teams do not have, in general, coincident periodic peak of Type3 activities. There are periods in which only some teams are very productive while some others have small or no activities at all. Sometimes it happens that all teams are very active at the same time. For sure the Data Center administrators want to optimize their return of investment and need the best algorithms for the dynamic allocation of all the available resources according to the following rules:

1. the Data Center must be as much as possible fully utilized (hopefully at 100%)
2. at the end of any "medium-long" period (e.g. a week, month) each team has consumed exactly the assigned share of usage
3. at any time any team having not yet saturated its own share gets VMs for its activities
4. there is no need to have the share respected at any instant

This scheduling approach is very different from the one addressed by Blazar project. Indeed the main goal is to maximize the productivity by complying a fair resource share policy which guarantees at any time the access to the VMs to any team which has not yet saturated his share.

Definitely this is not a simple issue because the site administrator of large Data Centers has to guarantee to its teams (O100) the right to completely expend the assigned share (periodically accounted) and to start their activities at any time if the share is not yet consumed at all.

If a resource becomes free, several teams will compete for them. The choice of the team to whom to give the first free resource must depend on the comparison of what has already been provided and thus is based on the number of incoming requests as well as the past usage. Storing incoming requests allows to prioritize incoming requests in a fair manner to ensure that each team get on average what they have been promised to get. Moreover, storing incoming requests instead of rejecting them immediately improves the possibility to keep resources completely busy as users may not be polling the system manually at the right time.

To make it more clear, for simplicity let's say that we have only two teams, "A" and "B" who are undertaking activities of Type3. The site administrator assigns at the beginning of a period of resource planning to "A" the 70% of resources (i.e. the share that is a concept different from the "quota" in the cloud's terminology) and therefore "B" has the remaining 30%. The site administrator wish to be able to make this partitioning dynamic, in such a way that "B" can use even the resources belonging to "A" just if not used and vice versa in order of being able to always have saturated its total capacity.

This is a typical scenario when the scientific teams or managers or public administrators require VMs for their asynchronous data analysis processing characterized by serial activations of an application for a limited time window (i.e. the time necessary to complete the calculations inherent to a number of files or objects). Whenever the time window expires the VM is automatically released.

This is a well known IT problem which has already been tackled and solved in the past by the Batch Systems schedulers which have been managing large CPU clusters constituted by tens of thousands of bare core CPUs for about 20 years and more recently of VMs. The core component of the Batch Systems which handles the resource allocation is the fair share scheduler which is made aware of the partitioning policy adopted and the historical resource usage. In particular it provides a dynamic priority algorithm and guarantees that the usage of the resources is distributed among users and teams according to their share (i.e provide a fair-share scheduling) by considering the portion of the resources allocated to them (i.e. share) and the evaluation of the effective resource usage consumed in the recent past. The decision policy takes into account a very large variety of attributes required by the users by including, for instance, network and storage in addition to the typical ones as CPU and memory.

Finally activities of Type1 simply decrease the total amount of resources effectively available to a team for the fair share scheduling of activities of Type2 and Type3 during the whole time period taken into consideration by the site administrator for the establishment of the share.