NemoProposal

NemoProposal

Abstract

Nemo is a data processing system for flexible employment with different execution scenarios for various deployment characteristics on clusters.

Proposal

Today, there is a wide variety of data processing systems with different designs for better performance and datacenter efficiency. They include processing data on specific resource environments and running jobs with specific attributes. Although each system successfully solves the problems it targets, most systems are designed in the way that runtime behaviors are built tightly inside the system core to hide the complexity of distributed computing. This makes it hard for a single system to support different deployment characteristics with different runtime behaviors without substantial effort.

Nemo is a data processing system that aims to flexibly control the runtime behaviors of a job to adapt to varying deployment characteristics. Moreover, it provides a means of extending the system's capabilities and incorporating the extensions to the flexible job execution.

In order to be able to easily modify runtime behaviors to adapt to varying deployment characteristics, Nemo exposes runtime behaviors to be flexibly configured and modified at both compile-time and runtime through a set of high-level graph pass interfaces.

We hope to contribute to the big data processing community by enabling more flexibility and extensibility in job executions. Furthermore, we can benefit more together as a community when we work together as a community to mature the system with more use cases and understanding of diverse deployment characteristics. The Apache Software Foundation is the perfect place to achieve these aspirations.

Background

Many data processing systems have distinctive runtime behaviors optimized and configured for specific deployment characteristics like different resource environments and for handling special job attributes.

For example, much research have been conducted to overcome the challenge of running data processing jobs on cheap, unreliable transient resources. Likewise, techniques for disaggregating different types of resources, like memory, CPU and GPU, are being actively developed to use datacenter resources more efficiently. Many researchers are also working to run data processing jobs in even more diverse environments, such as across distant datacenters. Similarly, for special job attributes, many works take different approaches, such as runtime optimization, to solve problems like data skew, and to optimize systems for data processing jobs with small-scale input data.

Although each of the systems performs well with the jobs and in the environments they target, they perform poorly with unconsidered cases, and do not consider supporting multiple deployment characteristics on a single system in their designs.

For an application writer to optimize an application to perform well on a certain system engraved with its underlying behaviors, it requires a deep understanding of the system itself, which is an overhead that often requires a lot of time and effort. Moreover, for a developer to modify such system behaviors, it requires modifications of the system core, which requires an even deeper understanding of the system itself.

With this background, Nemo is designed to represent all of its jobs as an Intermediate Representation (IR) DAG. In the Nemo compiler, user applications from various programming models (ex. Apache Beam) are submitted, transformed to an IR DAG, and optimized/customized for the deployment characteristics. In the IR DAG optimization phase, the DAG is modified through a series of compiler "passes" which reshape or annotate the DAG with an expression of the underlying runtime behaviors. The IR DAG is then submitted as an execution plan for the Nemo runtime. The runtime includes the unmodified parts of data processing in the backbone which is transparently integrated with configurable components exposed for further extension.

Rationale

Nemo's vision lies in providing means for flexibly supporting a wide variety of job execution scenarios for users while facilitating system developers to extend the execution framework with various functionalities at the same time. The capabilities of the system can be extended as it grows to meet a more variety of execution scenarios. We require inputs from users and developers from diverse domains in order to make it a more thriving and useful project. The Apache Software Foundation provides the best tools and community to support this vision.

Initial Goals

Initial goals will be to move the existing codebase to Apache and integrate with the Apache development process. We further plan to develop our system to meet the needs for more execution scenarios for a more variety of deployment characteristics.

Current Status

Nemo codebase is currently hosted in a repository at github.com. The current version has been developed by system developers at Seoul National University, Viva Republica, Samsung, and LG.

Meritocracy

We plan to strongly support meritocracy. We will discuss the requirements in an open forum, and those that continuously contribute to Nemo with the passion to strengthen the system will be invited as committers. Contributors that enrich Nemo by providing various use cases, various implementations of the configurable components including ideas for optimization techniques will be especially welcome. Committers with a deep understanding of the system's technical aspects as a whole and its philosophy will definitely be voted as the PMC. We will monitor community participation so that privileges can be extended to those that contribute.

Community

We hope to expand our contribution community by becoming an Apache incubator project. The contributions will come from both users and system developers interested in flexibility and extensibility of job executions that Nemo can support. We expect users to mainly contribute to diversify the use cases and deployment characteristics, and developers to contribute to implement them.

Alignment

Apache Spark is one of many popular data processing frameworks. The system is designed towards optimizing jobs using RDDs in memory and many other optimizations built tightly within the framework. In contrast to Spark, Nemo aims to provide more flexibility for job execution in an easy manner.

Apache Tez enables developers to build complex task DAGs with control over the control plane of job execution. In Nemo, a high-level programming layer (ex. Apache Beam) is automatically converted to a basic IR DAG and can be converted to any IR DAG through a series of easy user writable passes, that can both reshape and modify the annotation (of execution properties) of the DAG. Moreover, Nemo leaves more parts of the job execution configurable, such as the scheduler and the data plane. As opposed to providing a set of properties for solid optimization, Nemo's configurable parts can be easily extended and explored by implementing the pre-defined interfaces. For example, an arbitrary intermediate data store can be added.

Nemo currently supports Apache Beam programs and we are working on supporting Apache Spark programs as well. Nemo also utilizes Apache REEF for container management, which allows Nemo to run in Apache YARN and Apache Mesos clusters. If necessary, we plan to contribute to and collaborate with these other Apache projects for the benefit of all. We plan to extend such integrations with more Apache softwares. Apache software foundation already hosts many major big-data systems, and we expect to help further growth of the big-data community by having Nemo within the Apache foundation.

Known Risks

Orphaned Products

The risk of the Nemo project being orphaned is minimal. There is already plenty of work that arduously support different deployment characteristics, and we propose a general way to implement them with flexible and extensible configuration knobs. The domain of data processing is already of high interest, and this domain is expected to evolve continuously with various other purposes, such as resource disaggregation and using transient resources for better datacenter resource utilization.

Inexperience with Open Source

The initial committers include PMC members and committers of other Apache projects. They have experience with open source projects, starting from their incubation to the top-level. They have been involved in the open source development process, and are familiar with releasing code under an open source license.

Homogeneous Developers

The initial set of committers is from a limited set of organizations, but we expect to attract new contributors from diverse organizations and will thus grow organically once approved for incubation. Our prior experience with other open source projects will help various contributors to actively participate in our project.

Reliance on Salaried Developers

Many developers are from Seoul National University. This is not applicable.

Relationships with Other Apache Products

Nemo positions itself among multiple Apache products. It runs on Apache REEF for container management. It also utilizes many useful development tools including Apache Maven, Apache Log4J, and multiple Apache Commons components. Nemo supports the Apache Beam programming model for user applications. We are currently working on supporting the Apache Spark programming APIs as well.

An Excessive Fascination with the Apache Brand

We hope to make Nemo a powerful system for data processing, meeting various needs for different deployment characteristics, under a more variety of environments. We see the limitations of simply putting code on GitHub, and we believe the Apache community will help the growth of Nemo for the project to become a positively impactful and innovative open source software. We believe Nemo is a great fit for the Apache Software Foundation due to the collaboration it aims to achieve from the big data processing community.

Documentation

The current documentation for Nemo is at https://snuspl.github.io/nemo/.

Initial Source

The Nemo codebase is currently hosted at https://github.com/snuspl/nemo.

External Dependencies

To the best of our knowledge, all Nemo dependencies are distributed under Apache compatible licenses. Upon acceptance to the incubator, we would begin a thorough analysis of all transitive dependencies to verify this fact and further introduce license checking into the build and release process.

Cryptography

Not applicable.

Required Resources

Mailing Lists

We will operate two mailing lists as follows:

- Nemo PMC discussions: private@nemo.incubator.apache.org
- Nemo developers: dev@nemo.incubator.apache.org

Git Repositories

Upon incubation: https://github.com/apache/incubator-nemo.
After the incubation, we would like to move the existing repo https://github.com/snuspl/nemo to the Apache infrastructure

Issue Tracking

Nemo currently tracks its issues using the Github issue tracker: https://github.com/snuspl/nemo/issues. We plan to migrate to Apache JIRA.

Initial Committers

- Byung-Gon Chun
- Jeongyoon Eo
- Geon-Woo Kim
- Joo Yeon Kim
- Gyewon Lee
- Jung-Gil Lee
- Sanha Lee
- Wooyeon Lee
- Yunseong Lee
- JangHo Seo
- Won Wook Song
- Taegeon Um
- Youngseok Yang

Affiliations

- SNU (Seoul National University)
    - Byung-Gon Chun
    - Jeongyoon Eo
    - Geon-Woo Kim
    - Gyewon Lee
    - Sanha Lee
    - Wooyeon Lee
    - Yunseong Lee
    - JangHo Seo
    - Won Wook Song
    - Taegeon Um
    - Youngseok Yang
- LG
    - Jung-Gil Lee
- Samsung
    - Joo Yeon Kim
- Viva Republica

    - Geon-Woo Kim

Sponsors

Champions

Byung-Gon Chun

Mentors

- Davor Bonaci
- Hyunsik Choi
- Byung-Gon Chun
- Jean-Baptiste Onofré
- Markus Weimer
- Reynold Xin

Sponsoring Entity

The Apache Incubator