Spark on HPC

What is Spark?

SPARK is a programming language and verification toolset designed for developing high-assurance software systems. It is based on the Ada programming language and extends it with features for formal verification and high-reliability programming.

SPARK emphasizes formal verification to ensure the correctness and safety of software systems. This is achieved through the use of formal methods such as static analysis, model checking, and automated theorem proving.

SPARK has a strong type system that enforces correctness and safety at compile-time, catching many errors before they can occur at runtime. It also provides a rich set of annotations and contracts for specifying program behavior and ensuring correctness.

SPARK is often used in safety-critical and mission-critical systems, such as aerospace, defense, and medical devices. It has been certified to the highest levels of safety and security standards, such as DO-178B/C and Common Criteria.

For software documentation, SPARK provides a rich set of tools for generating documentation from code annotations and contracts. This includes generating API documentation, functional specifications, and safety manuals. SPARK also supports various documentation formats, such as HTML and LaTeX, to facilitate the creation of high-quality documentation.

Links:

Official Website

Manual

Versions Available:

The following versions are available on the cluster:

- spark-v3.0.3

How to load Spark?

To load Spark, use the following commands:

```
#Load the Spark module
module load spark/3.0.3
```

To verify if the module is loaded correctly, use the following command,

```
# List all the module loaded in the environment
module list
```

In a fresh environment, this should show multiple dependency module loaded along with spark.

How to use Spark?

Here are few outlines of how to use Spark:

1. Write SPARK code: SPARK code is written in the Ada programming language with additional annotations and contracts that specify program behavior and constraints. The SPARK language restricts certain constructs and features of Ada to ensure high-reliability programming.
2. Verify SPARK code: SPARK includes a suite of verification tools that can be used to verify program correctness and safety. These tools include static analyzers,

model checkers, and theorem provers. Verification can be done at compile-time or run-time, depending on the tool used.

3. Generate documentation: SPARK includes tools for generating documentation from code annotations and contracts. This documentation can be in various formats, such as HTML or LaTeX, and can include API documentation, functional specifications, and safety manuals.
4. Test SPARK code: SPARK code should be tested like any other code to ensure that it meets functional requirements and quality standards. SPARK includes tools for unit testing and integration testing and can also be integrated with external testing frameworks.
5. Deploy SPARK code: Once the SPARK code has been verified and tested, it can be deployed to the target platform. SPARK supports various platforms, such as bare metal, Linux, and RTOS.

Note that using SPARK can require specialized skills and knowledge, such as formal verification and high-reliability programming. It is recommended to seek training and guidance from experts in the field.

Where to find help?

If you are confused or need help at any point, please contact OIT at the following address.

https://ua-app01.ua.edu/researchComputingPortal/public/oitHelp