Julia

Treball Dirigit
Llenguatges de Programació

Grau en Enginyeria Informàtica

14778

Universitat Politècnica de Catalunya

Desembre 2024

Índex

1 Introducció

La computaci científica exigeix un alt rendiment, però molts experts prefereixen llenguatges dinàmics més lents per la seva flexibilitat. Aquesta dicotomia va portar els creadors de Julia (Jeff Bezanson, Stefan Karpinski, Viral B. Shah i Alan Edelman) a desenvolupar un llenguatge que superés aquesta limitació. Julia va ser presentat oficialment el 2012 com un projecte obert que combina la senzillesa i dinamisme de Python amb el rendiment de llenguatges com C o Fortran. El seu disseny se centra en satisfer les necessitats dels científics i enginyers, oferint un equilibri entre productivitat i velocitat d'execució.

2 Propòsit

Julia és un llenguatge de programació dissenyat específicament per a la computació científica i numèrica, tot i que també s'utilitza en altres àmbits gràcies a la seva versatilitat. Els seus principals objectius són:

- Combinar el rendiment de llenguatges compilats com C o Fortran amb la flexibilitat de llenguatges dinàmics com Python o MATLAB.
- Proporcionar una sintaxi senzilla i intuïtiva, que afavoreixi la productivitat dels desenvolupadors.
- Oferir un ecosistema integrat que inclou eines per al càlcul paral · lel, la visualització de dades i la integració amb altres llenguatges.

Aquest enfocament fa de Julia una eina poderosa per a científics, enginyers i qualsevol professional que necessiti combinar càlculs complexos amb eficiència i simplicitat.

3 Característiques de Julia

Julia recalca com un llenguatge de programació modern i flexible amb característiques úniques que fan que sigui especialment útil per la computació científica i altres camps. Aquí estan algunes de les seves característiques.

3.1 Paradigma

Julia és un llenguatge multiparadigma que suporta diversos estils de programació, incloent programació imperativa, funcional, orientada a objectes i metaprogramació.

3.1.1 Imperatiu

El paradigma imperatiu en Julia permet escriure codi seqüencial i estructurat:
<MINTED>

3.1.2 Funcional

Julia suporta característiques de programació funcional com funcions d'ordre superior, funcions anònimes i composició de funcions:
<MINTED>

També permet ens permet fer us de la curificació:
<MINTED>

A més d'això, podem fer servit l'operador . per aplicar una funció.
<MINTED>

3.1.3 Orientat a Objectes

Julia implementa un sistema de tipus múltiple i permet comportaments similars a POO (programació orientada a objectes):
<MINTED>

3.2 Sistema d'execució

Julia utilitza un sistema d'execució JIT (Just-In-Time) que compila el codi a màquina a través de LLVM (Low Level Virtual Machine) en temps d'execució, combinant els avantatges de la interpretació i la compilació.

3.3 Sistema de tipus

Julia és un llenguatge dinàmic amb un sistema de tipus opcional que permet especificar tipus per millorar el rendiment del codi. També ens permet definir tipus abstractes i tipus compostos.
<MINTED>
<MINTED>

3.4 Gestió de memòria

Julia utilitza un sistema de recollida de brossa per gestionar la memòria, però també permet als desenvolupadors controlar manualment l'alliberament de memòria.

3.5 Integració amb altres llenguatges

Julia pot integrarse facilment amb altres llenguatges de programació, pot fer crides a funcions natives dels seguents llenguatges:

- **Python**: PyCall.jl
- **R**: RCall.jl
- **C i Fortran**: CCall.jl, FortranCall.jl

4 Principals aplicacions

Avui en dia, Julia és un llenguatge de programació popular en investigacions científiques i enginyeria. Algunes de les seves aplicacions més destacades són:

- **Williams Racing**: L'empresa anglesa utilitza Julia per a la simulació de dinàmica de fluids en la seva fàbrica de F1. Això els permet optimitzar el disseny aerodinàmic dels seus vehicles i millorar el rendiment en pista.
- **AstraZeneca i Pfizer**: Dues empreses farmacèutiques que utilitzen Julia per a la investigació de medicaments. Julia els ajuda a accelerar el procés de descobriment de fàrmacs mitjançant la simulació de reaccions químiques i l'anàlisi de grans volums de dades.
- **NASA**: Utilitza Julia per analitzar dades relacionades amb el canvi climàtic. La capacitat de Julia per manejar càlculs complexos i grans conjunts de dades permet als científics de la NASA modelar fenòmens climàtics i predir canvis futurs amb major precisió.
- **Inversió financera**: Julia és utilitzada en el sector financer per a la modelització de riscos, l'optimització de carteres i l'anàlisi de dades de mercat. La seva velocitat i eficiència permeten als analistes financers prendre decisions més informades en temps real.
- **Intel · ligència artificial i aprenentatge automàtic**: Julia és popular en la comunitat d'IA i aprenentatge automàtic gràcies a la seva capacitat per implementar algoritmes complexos de manera eficient. Llibreries com Flux.jl permeten als desenvolupadors crear models d'aprenentatge profund amb facilitat.

- **Simulació de sistemes**: Julia s'utilitza per simular sistemes físics, químics i biològics en diverses disciplines científiques. La seva capacitat per executar càlculs paral · lels i distribuir tasques entre múltiples processadors la fa ideal per a simulacions a gran escala.

5 Problemes i limitacions

Malgrat les seves moltes avantatges, Julia presenta algunes limitacions que els usuaris han d'assumir:

- **Ecosistema de paquets**: Tot i que està creixent ràpidament, l'ecosistema de paquets de Julia encara no és tan ampli com els de llenguatges més madurs com Python o R. Això pot limitar la disponibilitat de biblioteques especialitzades per a certes aplicacions.
- **Temps de compilació**: Julia utilitza compilació Just-In-Time (JIT), la qual pot introduir retards en l'inici de l'execució de programes, especialment en scripts corts o interactius.
- **Gestió de la memòria**: Encara que Julia té un sistema de recollida de brossa, la gestió de la memòria pot ser menys intuïtiva comparada amb altres llenguatges, cosa que pot portar a problemes de rendiment si no es gestiona correctament.
- **Compatibilitat i interoperabilitat**: Integrar Julia amb altres llenguatges o sistemes pot ser més complex, tot i que s'estan desenvolupant solucions per millorar aquest aspecte.
- **Documentació i eines de depuració**: Encara que la documentació està millorant, algunes àrees poden ser menys detallades. Les eines de depuració i el suport integrat en entorns de desenvolupament també poden ser menys robustes comparades amb altres llenguatges.

6 Perspectiva de futur

El futur de Julia es presenta prometedor per diverses raons:

- **Creixement de la comunitat**: La comunitat de Julia continua creixent, amb més desenvolupadors i empreses adoptant el llenguatge.
- **Millores en rendiment**: Els desenvolupadors de Julia segueixen optimitzant el compilador i reduint els temps de latència.
- **Expansió de l'ecosistema**: Cada vegada hi ha més paquets disponibles, cobrint més àrees d'aplicació.
- **Adopció en la indústria**: Grans empreses i institucions estan adoptant Julia per a projectes crítics.

- **Intel · ligència Artificial**: Julia està ben posicionat per ser un llenguatge líder en IA i aprenentatge automàtic.
- **Computació quàntica**: S'està desenvolupant suport per a computació quàntica, obrint noves possibilitats.

7 Fonts d'informació i Avaluació de la Qualitat

Les fonts d'informació utilitzades per a aquest document provenen de diverses plataformes reconegudes i fiables. La documentació oficial de Julia proporciona una base sòlida i autoritzada sobre el llenguatge de programació. Els articles de científics de renom i publicacions en blogs especialitzats com Scientific Coder i JuliaLang aporten exemples pràctics i context aplicat. A més, les fonts com Simple Wikipedia ofereixen una introducció accessible i concisa. Els estudis de casos de JuliaHub i articles de mitjans com Medium complementen aquesta bibliografia amb aplicacions reals i perspectives actuals. En conjunt, les referències seleccionades garanteixen una cobertura equilibrada entre teoria i pràctica, assegurant la qualitat i fiabilitat de la informació presentada.

8 Bibliografia

Referències

[1] Julia Documentation. *Julia v1.0 Documentation*.
Available at: https://docs.julialang.org/en/v1/

[2] Scientific Coder. *Straightforward Functional Programming Examples in Julia*.
Available at: https://scientificcoder.com/straightforward-functional-programming-examples-in-julia

[3] Simple Wikipedia. *Julia (programming language)*.
Available at:
https://simple.wikipedia.org/wiki/Julia_(programming_language)

[4] JuliaLang. *Why We Created Julia*.
Available at:
https://julialang.org/blog/2012/02/why-we-created-julia

[5] Test Subjector. *The Julia Compilation Process*.
Available at: https://testsubjector.github.io/blog/2020/03/26/The-Julia-Compilation-Process

https://medium.com/@filip.sekan/pros-and-cons-of-julia-in-data-science-3f386cb71757

[6] Filip Sekan. *Pros and Cons of Julia in Data Science*.
Available at: https://medium.com/@filip.sekan/pros-and-cons-of-julia-in-data-science-3f386cb71757

[7] JuliaHub. *JuliaHub Case Studies*.
Available at: https://info.juliahub.com/case-studies