Package 'future.mirai'

July 3, 2024

Version 0.2.2

Depends future

Imports mirai (>= 1.1.0), parallelly, utils

Suggests future.tests, future.apply, listenv

Title A 'Future' API for Parallel Processing using 'mirai'

Description Implementation of the 'Future' API <doi:10.32614/RJ-2021-048> on top of the 'mirai' package <doi:10.5281/zenodo.7912722>. This allows you to process futures, as defined by the 'future' package, in parallel out of the box, on your local machine or across remote machines. Contrary to back-ends relying on the 'parallel' package (e.g. 'multisession') and socket connections, 'mirai_cluster' and 'mirai_multisession', provided here, can run more than 125 parallel R processes.

License GPL (>= 3)

Encoding UTF-8

URL https://future.mirai.futureverse.org, https://github.com/futureverse/future.mirai

BugReports https://github.com/futureverse/future.mirai/issues

RoxygenNote 7.3.2

NeedsCompilation no

Author Henrik Bengtsson [aut, cre, cph] (<https://orcid.org/0000-0002-7579-5165>), Charlie Gao [ctb] (<https://orcid.org/0000-0002-0750-061X>)

Maintainer Henrik Bengtsson <henrikb@braju.com>

Repository CRAN

Date/Publication 2024-07-03 11:40:02 UTC

Contents

future.mirai *future.mirai: A Future API for Parallel Processing using 'mirai'*

Description

The **future.mirai** package implements the Future API using the **mirai** package.

Author(s)

Maintainer: Henrik Bengtsson <henrikb@braju.com> (ORCID) [copyright holder]

Other contributors:

- Charlie Gao <charlie.gao@shikokuchuo.net> (ORCID) [contributor]

See Also

Useful links:

- https://future.mirai.futureverse.org
- https://github.com/futureverse/future.mirai
- Report bugs at https://github.com/futureverse/future.mirai/issues

Examples

```
TRUE
```

mirai_cluster *Mirai-based cluster futures*

Description

Mirai-based cluster futures

Usage

```
mirai_cluster(expr, substitute = TRUE, envir = parent.frame(), ...)
```

Arguments

expr	An R expression.
substitute	If TRUE, argument expr is substitute():ed, otherwise not.
envir	The environment from where global objects should be identified.
...	Additional named elements of the future.

Value

An object of class MiraiFuture.

Examples

```
mirai::daemons(parallelly::availableCores(), dispatcher = FALSE)
plan(mirai_cluster)

# A function that returns a future, note that N uses lexical scoping...
f <- function() future({4 * sum((runif(N) ^ 2 + runif(N) ^ 2) < 1) / N}, seed = TRUE)

# Run a simple sampling approximation of pi in parallel using  M * N points:
N <- 1e6  # samples per worker
M <- 10   # iterations
pi_est <- Reduce(sum, Map(value, replicate(M, f()))) / M
print(pi_est)

plan(sequential)
invisible(mirai::daemons(0)) ## Shut down mirai workers
```

---

mirai_multisession *Mirai-based localhost multisession futures*

Description

Mirai-based localhost multisession futures

Usage

```
mirai_multisession(
  expr,
  substitute = TRUE,
  envir = parent.frame(),
  ...,
  workers = availableCores()
)
```

Arguments

expr	An R expression.
substitute	If TRUE, argument expr is substitute():ed, otherwise not.
envir	The environment from where global objects should be identified.
...	Additional named elements of the future.
workers	The number of parallel processes to use. If a function, it is called without arguments *when the future is created* and its value is used to configure the workers.

Value

An object of class MiraiFuture.

Examples

```
plan(mirai_multisession)

# A function that returns a future, note that N uses lexical scoping...
f <- function() future({4 * sum((runif(N) ^ 2 + runif(N) ^ 2) < 1) / N}, seed = TRUE)

# Run a simple sampling approximation of pi in parallel using  M * N points:
N <- 1e6  # samples per worker
M <- 10   # iterations
pi_est <- Reduce(sum, Map(value, replicate(M, f()))) / M
print(pi_est)

plan(sequential)
invisible(mirai::daemons(0)) ## Shut down mirai workers
```

Index