startR: retrieval of multidimensional distributed data sets

Nicolau Manubens[1], Virginie Guemas[1, 2], Pierre-Antoine Bretonnière[1], Julia Giner[1], Alasdair Hunter[1]

(1) Earth Sciences Department, Barcelona Supercomputing Center, Spain (2) Centre National de Recherches Météorologiques, France

The first step in data analysis made easy

◊ **Data retrieval and alignment** is one of the first steps in data analysis, and is often highly **complex and time-consuming**. This is specially crucial in the era of Big Data, where large multidimensional data sets from diverse sources need to be combined and processed.

◊ **startR** is an R project started at BSC aiming to develop a tool that allows the user **to automatically retrieve, homogenize and align** subsets of **multidimensional distributed data sets** for later analysis and operation.

◊ It is an **open source** project that is **open to external collaboration** and funding, and will continuously evolve to support as many data set formats as possible while maximizing its efficiency. **startR** v0.0.2 is currently **available on CRAN** (cran.r-project.org/web/packages/startR).

Design

◊ The **startR** package provides mechanisms to easily combine and retrieve data sets stored in several files and folders in local or remote servers.

Support for any file format can easily be added **by plugging in a "file reader"** function for that specific format, which can usually be written in less than 100 lines of code. File readers for **NetCDF**, **CSV and XLS** formats are currently implemented and included in the package. Retrieval of data from remote file servers, THREDDS servers or other **OPeNDAP-speaking servers** is **supported**.

◊ **startR** provides an abstraction of the data files in a way that the whole involved data can be perceived as a large multidimensional array.

After some initial configuration to help the package recognize how the files are distributed, **the user** can work with the multidimensional abstraction and **does not longer need to worry about the file distribution**. Once the data has been mapped to this abstraction, the user can easily select, transform and arrange data subsets for later analysis. For example, the user can request for automatic **transformations** to be done **on the fly**, such as regridding raster data or averaging across a specific dimension.

◊ R arrays with named dimensions are the fundamental working piece of this package.

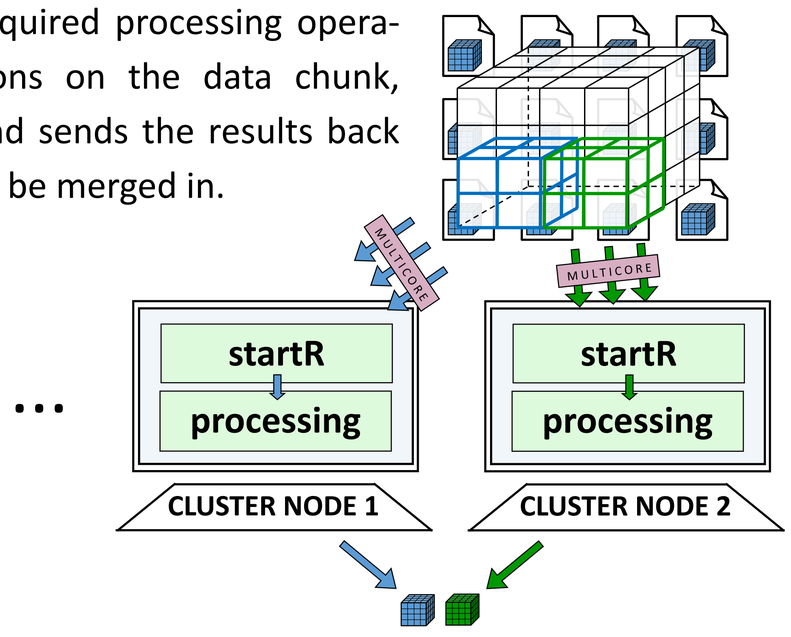
Their **shape can easily be configured**. It will arrange the data with the dimensions in the same order they are requested. Additionally, **metadata can be extracted** and attached as attributes to the data array, as long as the used file reader is able to extract it from the files.

◊ It is designed for an optimal performance.

The data is retrieved in parallel with the required data reorderings the transformations in **multiple cores**, so that the network connection use is maximized.

***startR** in Big Data workflows*

startR is an ideal retrieval and alignment tool to run heavy computations on computing clusters. Each node uses startR to retrieve and prepare its corresponding chunk from a large data set. Then, it applies the required processing operations on the data chunk, and sends the results back to be merged in.



Prospects *(as of July 2017)*

◊ **Caché-ing requests** and results to avoid repeated data transfers for identical requests close in time.

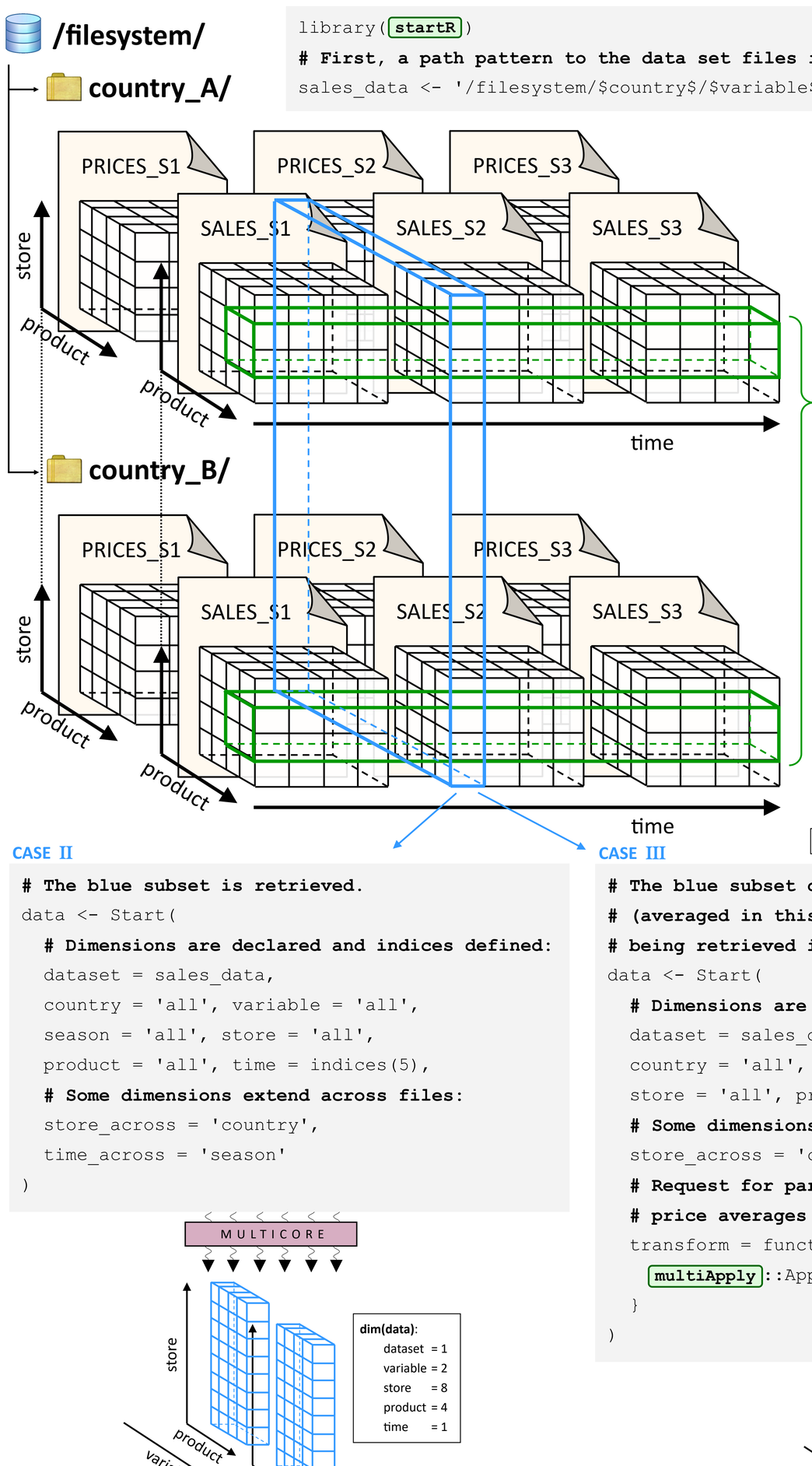
◊ Developing tools to **automatically retrieve** data sets by chunks **and apply** a set of **user-defined operations on a cluster** of workstations.

◊ Inclusion of new file readers to **support additional file formats**.

Example

The following illustration represents an example of a distributed data set with sales data from a multinational supermarket chain, with a number of stores in each country. Each month of the year, all the stores record the number of sales for each product in one file, and the retail price of each product in another file. Each new season, a separate new file is started.

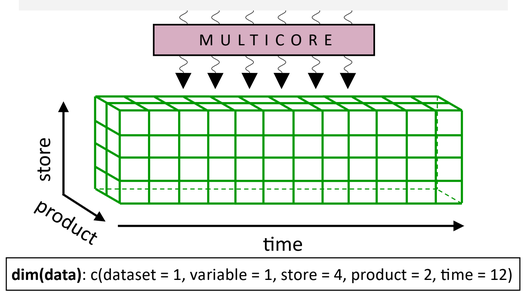
The blue and green wired boxes represent subsets of data being retrieved with **startR**. The gray rectangles contain the code used to retrieve the subsets.




```
library(startR)
# First, a path pattern to the data set files is built. $wildcards$ can be used.
sales_data <- '/filesystem/$country$/$variable$_S$season$.xls'
```

CASE I

```
# The green subset is retrieved.
data <- Start(
  # Dimensions are declared and
  # indices defined for each:
  dataset = sales_data,
  country = 'all',
  variable = 'SALES',
  season = 'all',
  store = indices(2:3),
  product = indices(3:4),
  time = 'all',
  # Some dimensions extend
  # across files:
  store_across = 'country',
  time_across = 'season'
)
```



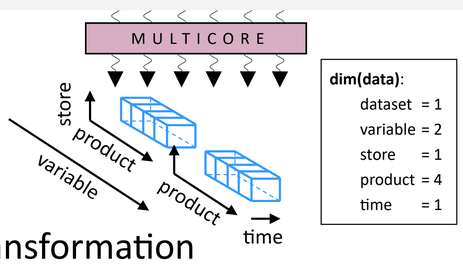
dim(data): c(dataset = 1, variable = 1, store = 4, product = 2, time = 12)

CASE II

```
# The blue subset is retrieved.
data <- Start(
  # Dimensions are declared and indices defined:
  dataset = sales_data,
  country = 'all', variable = 'all',
  season = 'all', store = 'all',
  product = 'all', time = indices(5),
  # Some dimensions extend across files:
  store_across = 'country',
  time_across = 'season'
)
```


CASE III

```
# The blue subset can be processed on the fly
# (averaged in this example) at the same time it is
# being retrieved in parallel.
data <- Start(
  # Dimensions are declared and indices defined:
  dataset = sales_data,
  country = 'all', variable = 'all', season = 'all',
  store = 'all', product = 'all', time = indices(5),
  # Some dimensions extend across files:
  store_across = 'country', time_across = 'season',
  # Request for parallel computation of sale and
  # price averages across the stores:
  transform = function(x, ...) {
    multiApply::Apply(x, mean, 'store')
  }
)
```



This simple example does not demonstrate the full selection, transformation and alignment capabilities of **startR**. Besides, Start() supports data sources with any number of dimensions in any order and with any names. Also, multiple local or remote sources in other formats, such as NetCDF, can be combined in a single call. In this multidimensional array framework, the package **multiApply** becomes specially useful to efficiently operate components.

European Commission

MINISTERIO DE ECONOMÍA, INDUSTRIA Y COMPETITIVIDAD