OBANSoft: aplicación para el análisis bayesiano objetivo y subjetivo. Estudio de su optimización y paralelización

Manuel Quesada Martínez

12 de julio de 2010

Índice

¿Por qué la aplicación *OBANSoft*?

Motivación: llenar el vacío existente respecto a aplicaciones para el análisis bayesiano de datos con mínima información a priori.

- **Diseño modular** para facilitar:
  - Extensión con nueva funcionalidad.
  - Independencia del modelo estadístico.
- **Integración de tecnologías**.
- Utilizar como **punto de partida** de esta línea de investigación en computación de altas prestaciones.

Grupos de investigación

- (UMU): **Grupo de Computación Científica y Programación Paralela**:

  Experiencia en el desarrollo y optimización de código paralelo. Incluyendo técnicas de autooptimización y en la aplicación de la computación paralela en diversos campos científicos.

- (UMH): **Grupo de Estadística Bayesiana**:

  Experiencia en el desarrollo de códigos de simulación aplicables a la resolución de análisis bayesianos en diversos campos.

Alternativas disponibles para anális bayesianos

Alternativas y **lenguajes** para el análisis bayesiano:

- Algunas librerías de SAS.
- Software bayesiano Winbugs.
- El proyecto R.

Interfaces gráficas para R:

- RCommander.
- JGR.

Alternativas disponibles para anális bayesianos

Alternativas y **lenguajes** para el análisis bayesiano:

- Algunas librerías de SAS. (Software de pago)
- Software bayesiano Winbugs. (Solo análisis subjetivos)
- El proyecto R.

Interfaces gráficas para R: (Insuficientes para su extensión)

- RCommander.
- JGR.

Artefactos, tecnología y herramientas

Se recoge el resumen de los objetos *software*, tecnología y librerías utilizadas:

Elemento Software	Tecnologías	Librerías
Librería estadística	*Java (JSE) + R*	*JRI*
Aplicación de escritorio	*Java Swing*	*Swing*
Paralelización	*R paralelo*	*Snow Fall*

- Se ha programado y probado en sistemas: *Windows XP*, *Ubuntu 8.10* y *MacOS Leopard*
- Los experimentos para los códigos paralelos programados se han realizado en la máquina *Luna.inf.um.es*: procesador Intel Quad Core 4 y 4 GB de RAM.

Requisitos, funcionalidad y módulos

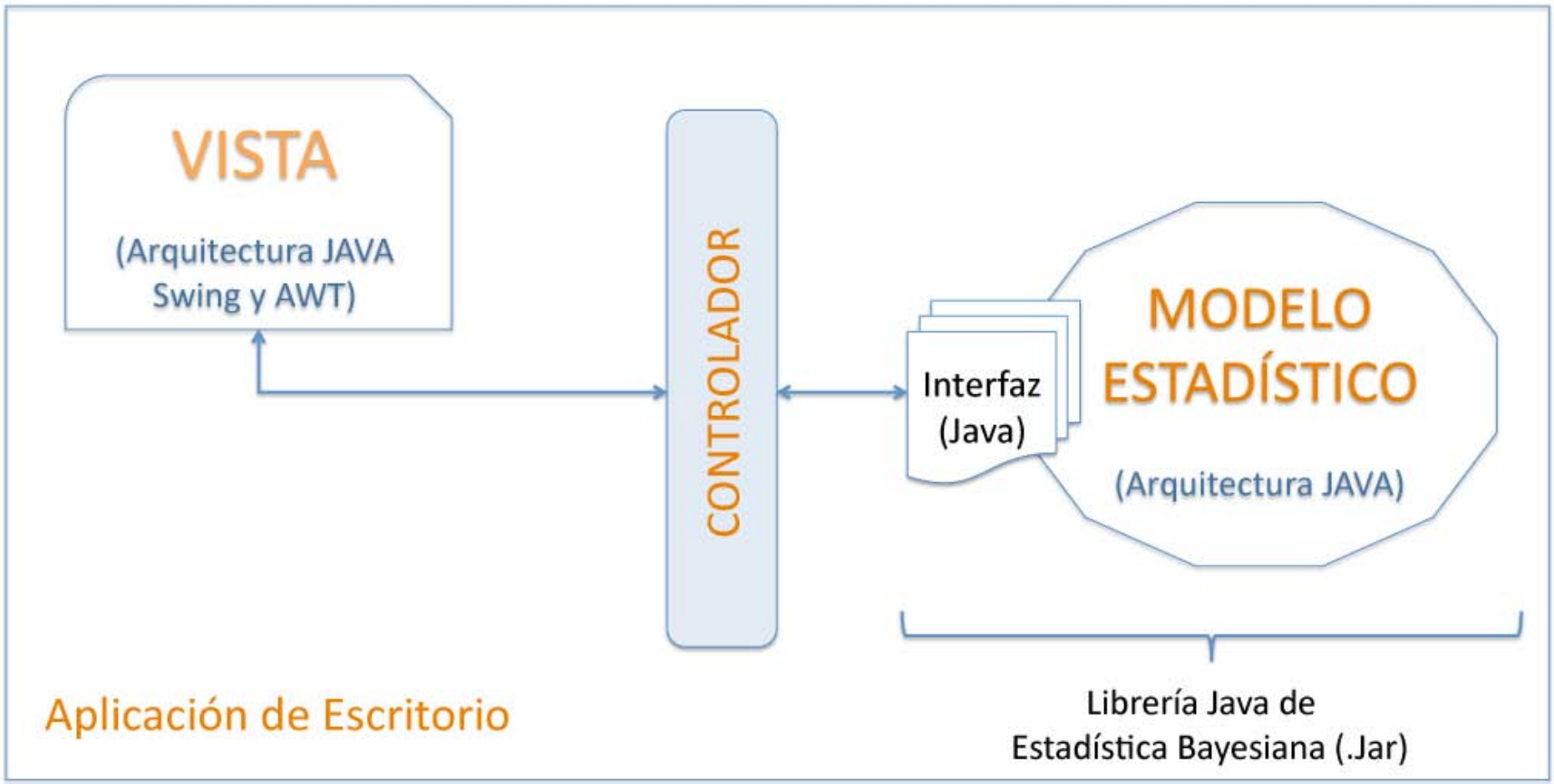
- Para desarrollar una aplicación que infiera con modelos estadísticos bayesianos sobre bancos de datos, se debe **soportar previamente la funcionalidad mínima** para la edición y manipulación de los mismos.
- **Módulos de requisitos (Según su funcionalidad)**:
  - Creación y edición de bancos de datos (*DataFrames*).
  - Descriptivas numéricas y gráficas.
  - Transformaciones sobre los datos.
  - Simulación de datos.
  - Modelización y análisis.

Resumen de la metodología aplicada

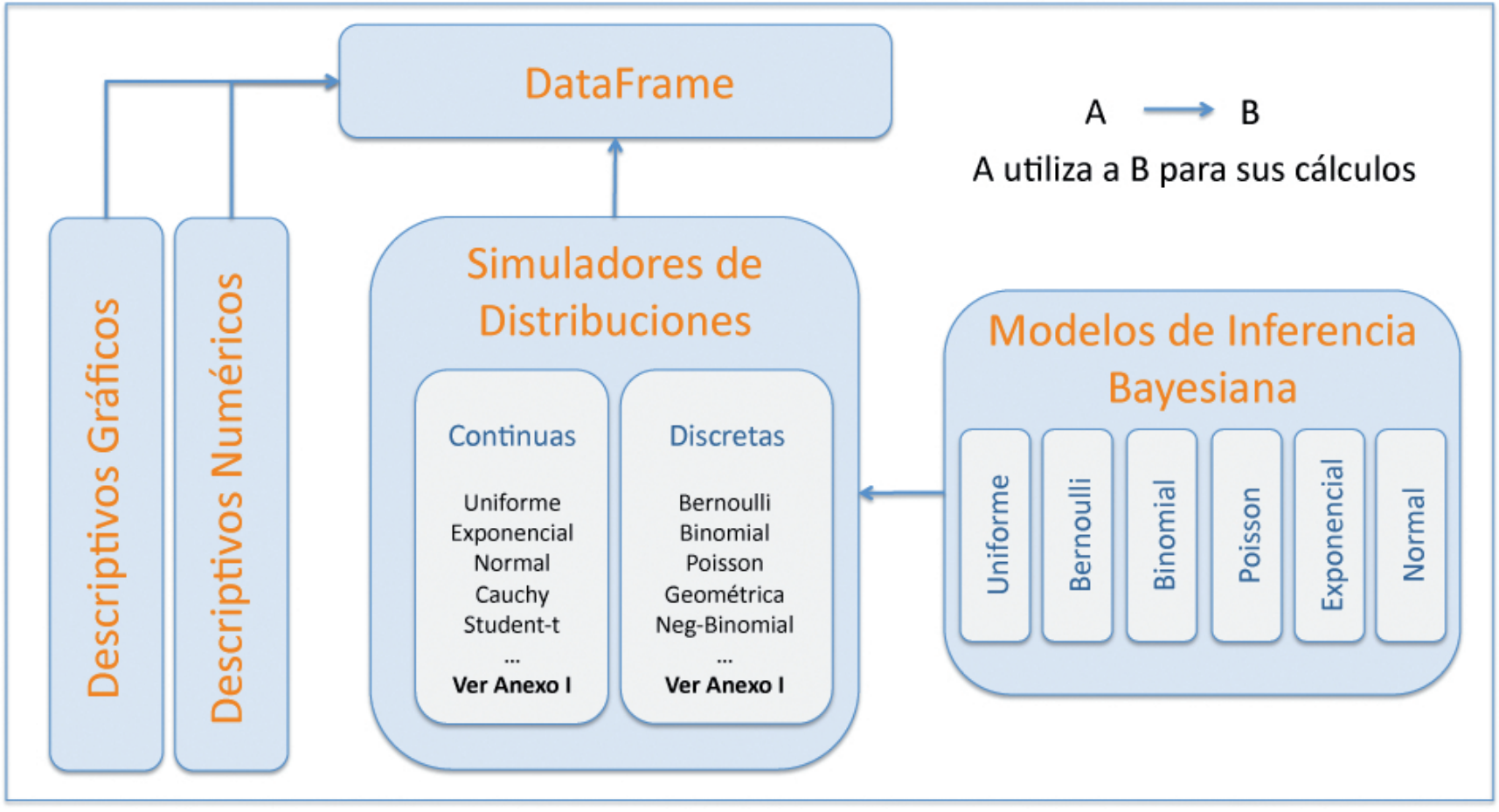
Abordar diversas áreas nos lleva a dividir la metodología en 4 partes:

- **Parte I**: elaboración de un catálogo que recoja las operaciones bayesianas que soporta la aplicación.
- **Parte II**: decidir qué tecnologías y recursos se utiliza para diseñar e implementar la aplicación y la librería de funciones.
- **Parte III**: diseño e implementación de la librería y la aplicación de escritorio.
- **Parte IV**: estudio del rendimiento y paralelización de los algoritmos de simulación.

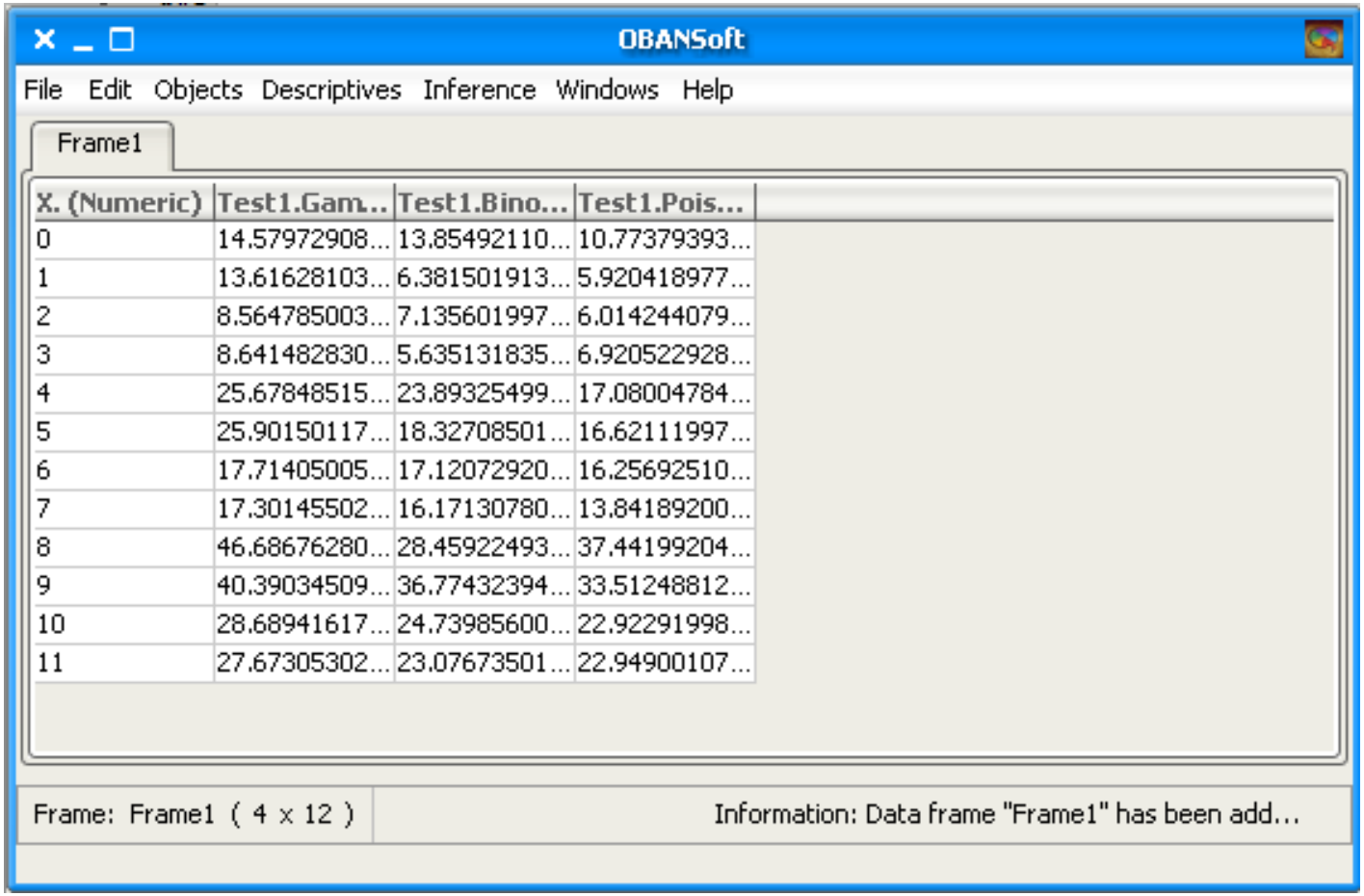
El modelo Modelo-Vista-Controlador



Objetos del Modelo



Objetos de la Vista



Objetos del Controlador

Controlador principal que gestiona todos los eventos que requieren la participación del "MainForm": **MainController**.

Controladores por módulos:

FileController

EditController

DescriptiveController

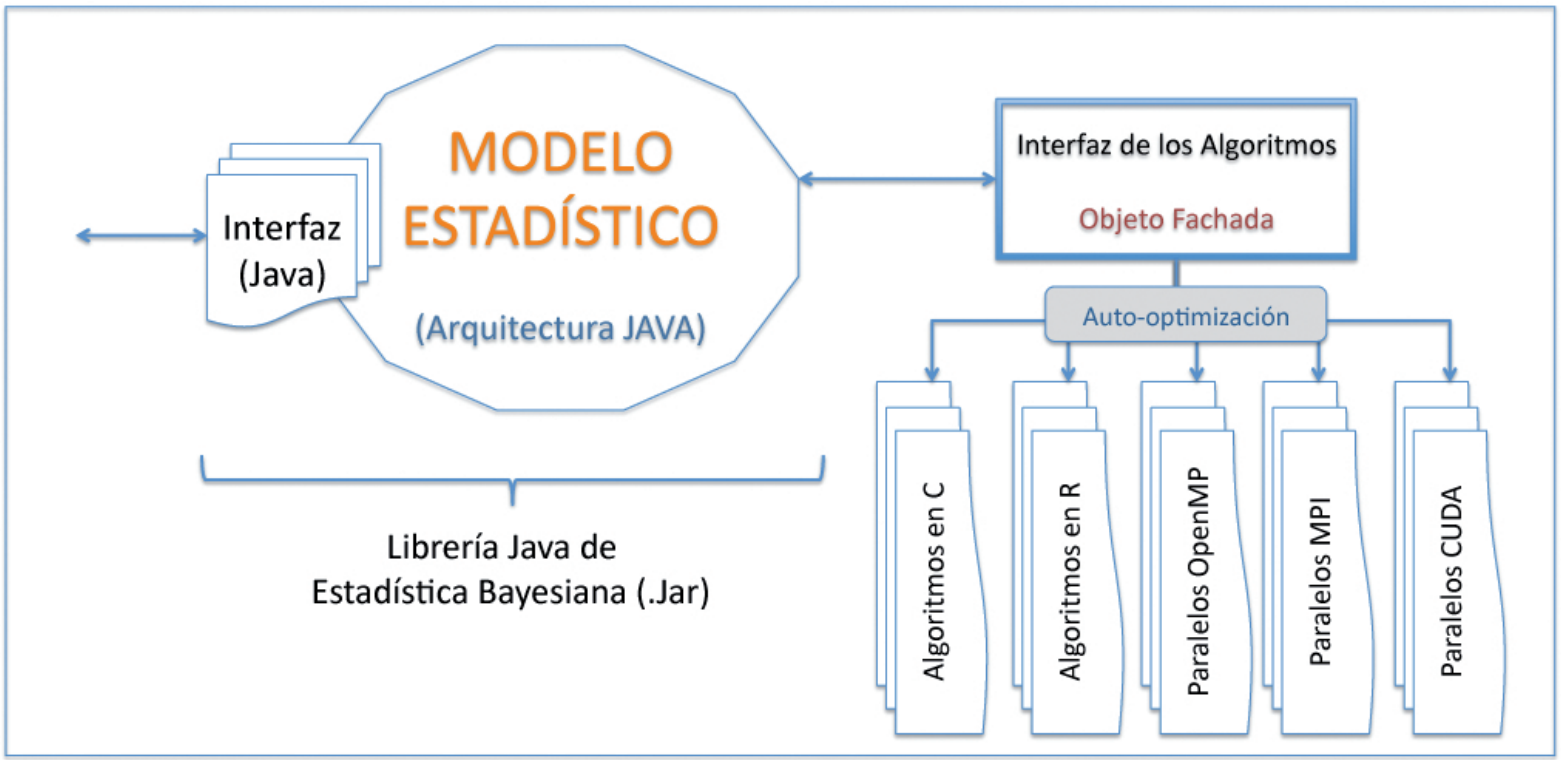
InferenceController

Otros objetos:

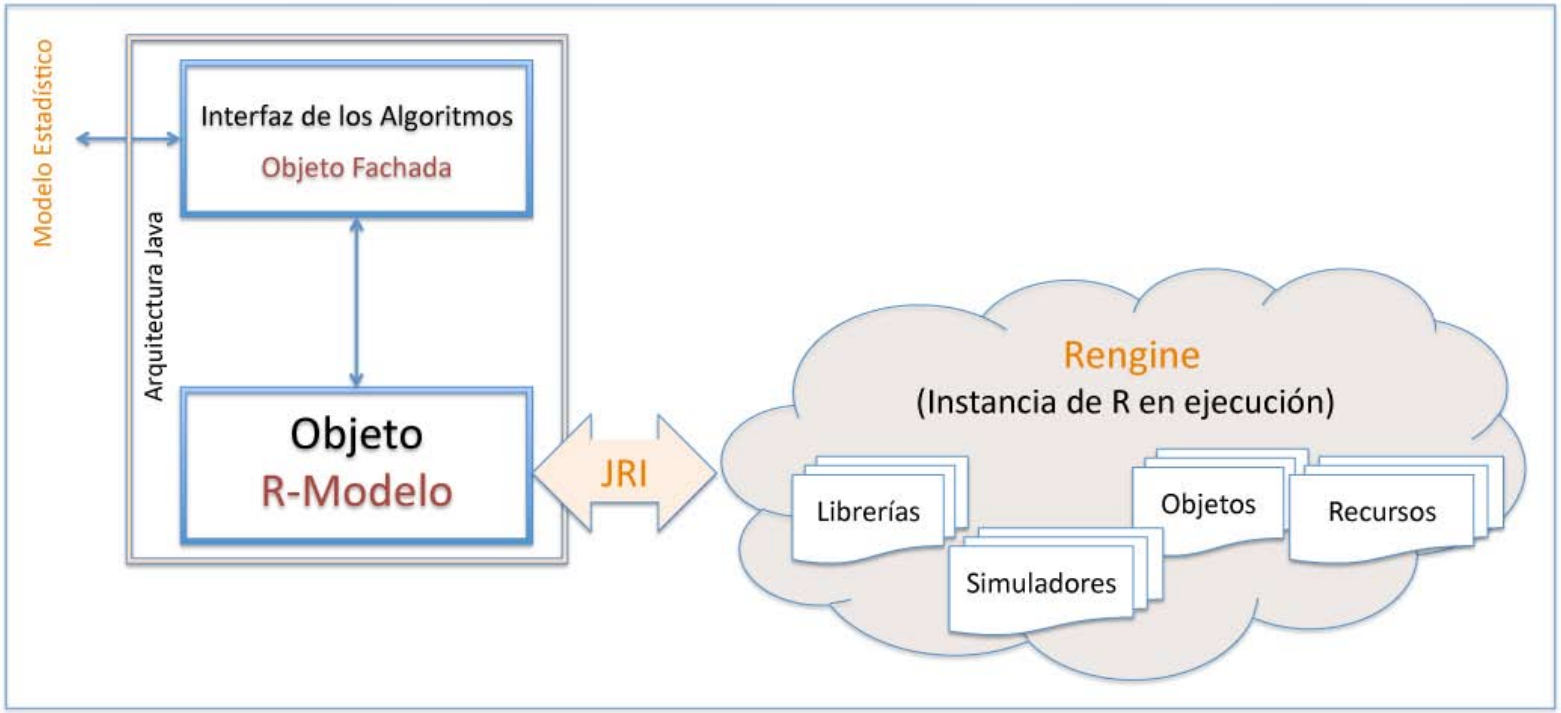
DataFramesController

GestorGraficos

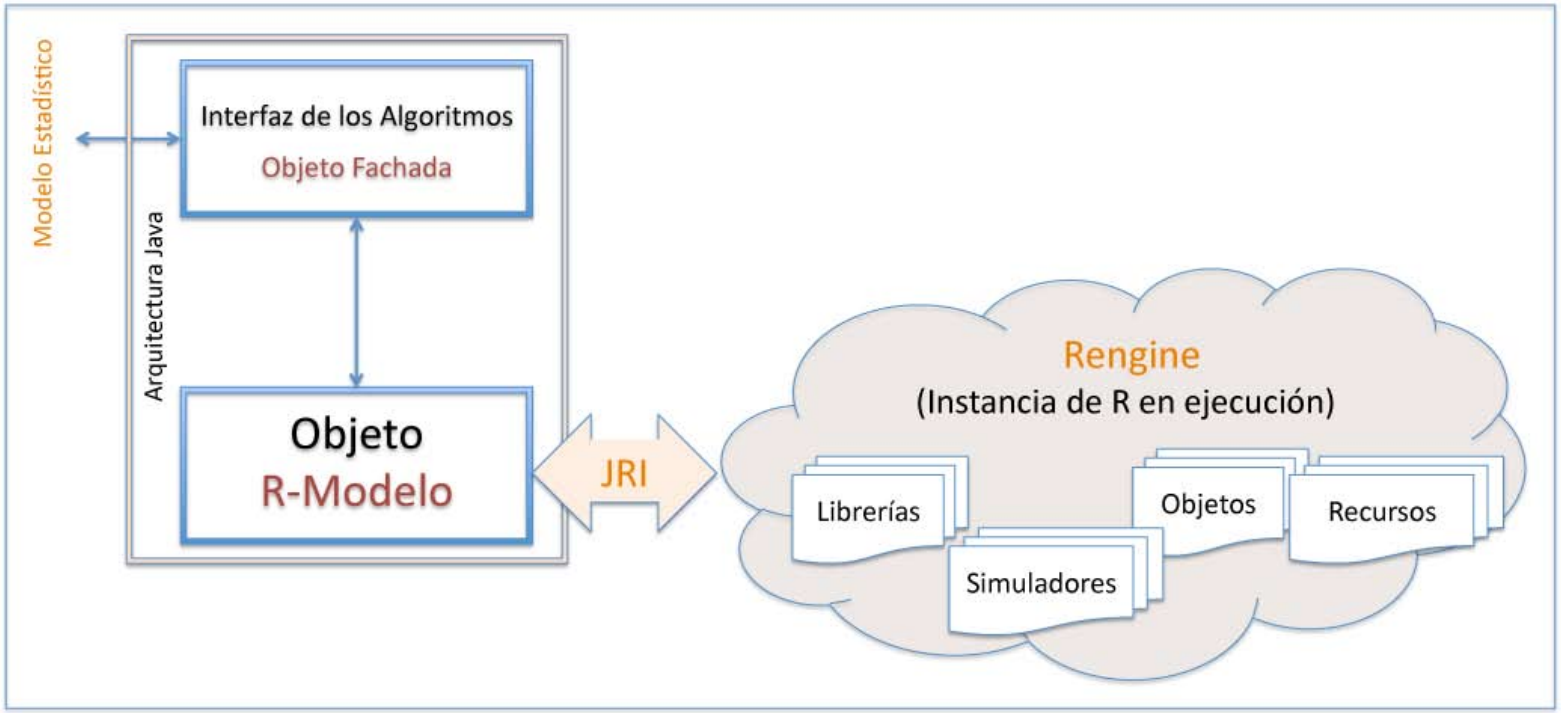
Algoritmos bayesianos. Integración de tecnologías



El R-Modelo y su integración con R



El R-Modelo y su integración con R



Logs de la aplicación: logs de la aplicación y log del R-Modelo.

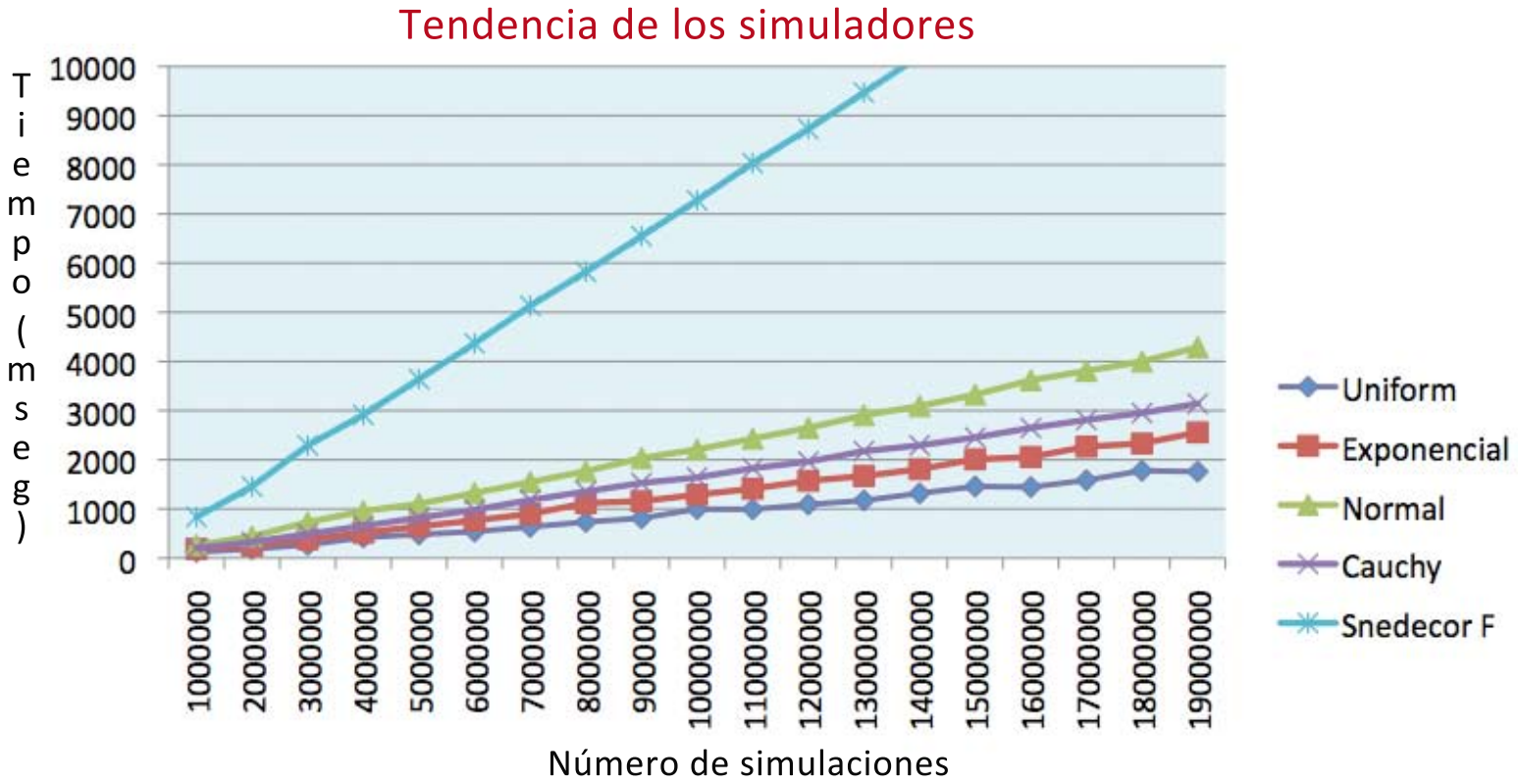
¿Qué algoritmos optimizar y paralelizar?

- Entre todos los algoritmos programados nos centramos en los **algoritmos de simulación**.
  - Consumen mayor tiempo de ejecución.
  - Punto crítico de la resolución de un análisis bayesiano.
  - Todo el análisis se basa en la simulación. Son utilizados para la inferencia de los modelos bayesianos.
- Tenemos en la aplicación un total de 27 algoritmos programados!
- **Metodología (Parte IV)**: realizamos un estudio experimental de los tiempos de ejecución de todos ellos y seleccionamos los más adecuados.

¿Qué algoritmos optimizar y paralelizar?

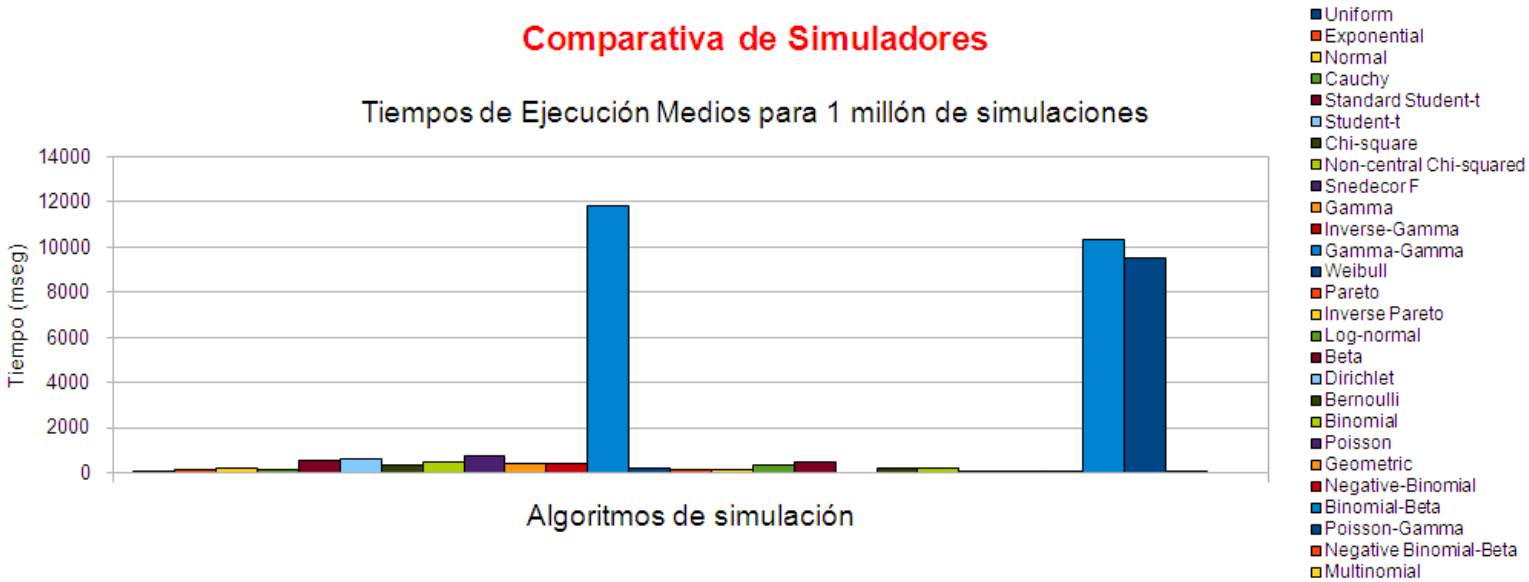
- Entre todos los algoritmos programados nos centramos en los **algoritmos de simulación**.
  - Consumen mayor tiempo de ejecución.
  - Punto crítico de la resolución de un análisis bayesiano.
  - Todo el análisis se basa en la simulación. Son utilizados para la inferencia de los modelos bayesianos.
- Tenemos en la aplicación un total de 27 algoritmos programados!
- **Metodología (Parte IV)**: realizamos un estudio experimental de los tiempos de ejecución de todos ellos y seleccionamos los más adecuados.

Experimento 1: Tendencia del crecimiento



Tendencia lineal en el tiempo de ejecución a medida que aumentamos el **número de simulaciones.**

Experimento 2: Comparativa de los simuladores



Se observan dos tipos de simuladores: **Simuladores Simples** y **Simuladores Compuestos.**

Estructura de los Simuladores Compuestos

- Invocamos una vez a una **función simple** de tamaño X.
- Invocamos X veces a otra función simple (**función cadena**) con parámetros extraídos de la función anterior.

```
1  rgamma.gamma = function(nsim, alpha, beta, nu) {
2    theta=rgamma(nsim, alpha, beta);
3    x=vector(length=nsim);
4    for(i in 1:nsim) {
5            theta[i]=rgamma(1, nu, theta[i]);
6    };
7
8    return(theta);
9  }
```

Código 1. Algoritmos de simulación de la función compuesta *Gamma-Gamma*.

- Los experimentos indican que la **función cadena consume un 90 % del tiempo total de ejecución**.

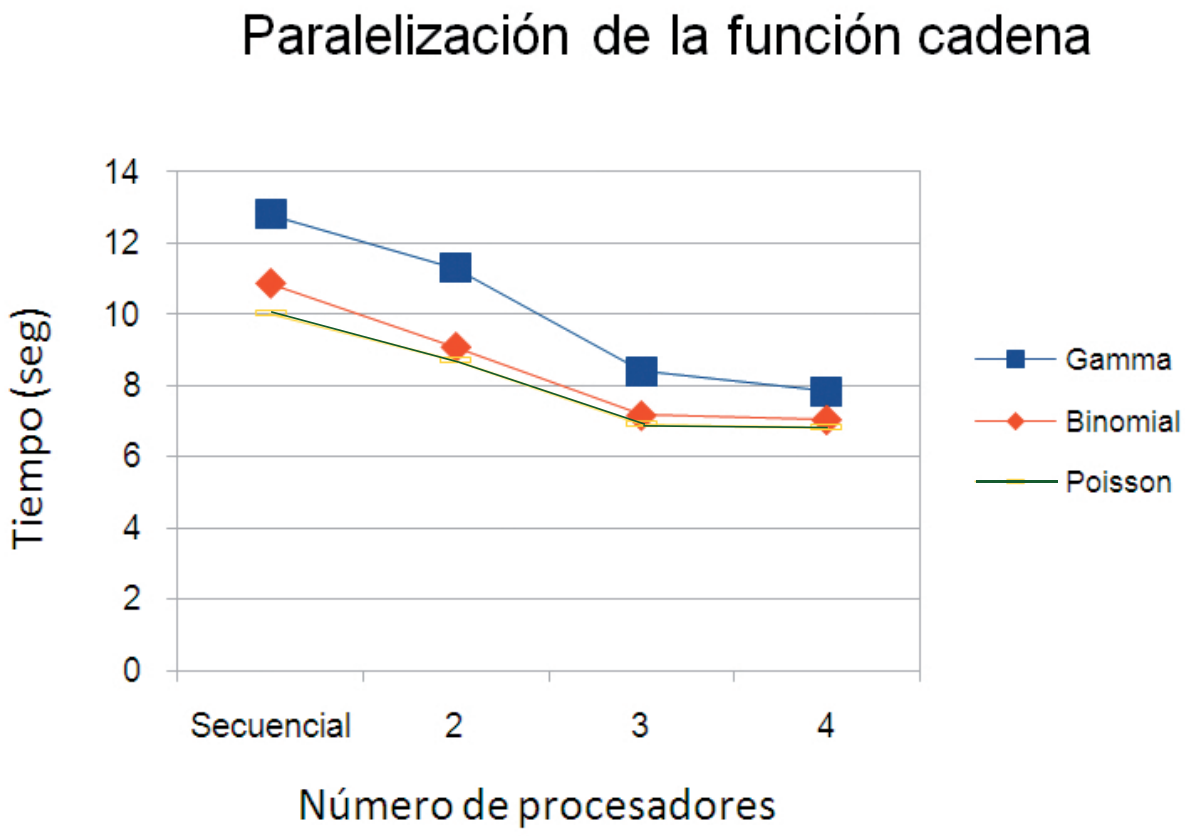
Paralelizamos la función cadena con R paralelo.

Paralelización para memoria compartida (SnowFall)

```
1   # Calculariamos con la funcion simple los parametros
2   # alpha y beta que usaremos en la funcion rgamma.
3   ... ... ...
4   library(snowfall)
5
6   # 1. Inicializamos snowfall
7   sfInit(parallel=TRUE, cpus=1, type="SOCK")
8
9   # 2. Cargariamos bancos de datos que queremos que
10  # sean leidos por todos los procesadores
11  # require(mvna)
12  # data(sir.adm)
13
14  # 3. Definimos el wrapper, el cual va a ser paralelizado.
15  wrapper <- function(idx) { return(rgamma(1,mu,theta)); }
16
17  # 4. Exportariamos los datos y paquetes que queremos
18  # que sean leidos por todos los procesadores
19  # sfExport("sir.adm")
20  # sfLibrary(cmprsk)
21
22  # 5. Inicializamos el generador paralelo de numeros aleatorios
23  sfClusterSetupRNG()
24
25  # 6. Distribuimos los calculos
26  result <- sfLapply(1:tamSimulaciones, wrapper);
27
28  # 7. Detenemos snowfall
29  sfStop()
30  ... ... ...
31  ## Devolvemos el resultado de la simulacion
```

Código 2. Algoritmo paralelo de la función cadena del simulador *Gamma-Gamma*.

Experimento 3: Resultado de la paralelización



Pese a la ganancia, no se alcanza el límite teórico p.

Conclusiones

- *OBANSoft* ha permitido **cubrir el vacío** dentro de las aplicaciones de estadística bayesiana. Ha sido **presentado** en el congreso *Valencia International Meeting on Bayesian Statistics*, **2010 ISBA World Meeting**.
- Por su diseño modular y la integración de tecnologías, la librería y la aplicación **servirán de base** para integrar otros resultados obtenidos en la continuación de esta línea de investigación.
- Basándonos en el estudio experimental de los algoritmos paralelos, parece que las **funciones compuestas pueden ser paralelizadas** para conseguir beneficios en el tiempo de ejecución. La utilización de **SnowFall** ha **limitado las conclusiones**.

Trabajos futuros

- Continuar con el **estudio** del comportamiento de la **librería SnowFall**. Podríamos aplicar la metodología seguida en el proyecto de Virginio García López.
- Buscar los códigos de los **simuladores** programados **en C**, y compararlos con las funciones en R.
- *Paralelizar* los algoritmos programados en C con **OpenMP** y comprarlos con SnowFall.
- **Integrar** otros modelos que implican la simulación con algoritmos basados en **cadenas de Markov**.
- **Ampliar los módulos** de OBANSoft con nueva funcionalidad.
- Adaptar el **modelo estadístico** a una web para explotarlo como **Cloud Computing**.

OBANSoft: aplicación para el análisis bayesiano objetivo y subjetivo. Estudio de su optimización y paralelización

Manuel Quesada Martínez

12 de julio de 2010