

Metaheurísticas paralelas: optimización y aplicaciones

Domingo Giménez
dis.um.es/~domingo

Grupo de Computación Científica y Programación Paralela
luna.inf.um.es/grupo_investigacion

Departamento de Informática y Sistemas, Universidad de Murcia



Universidad Politécnica de Valencia, 2019

Contenidos

1 Metaheurísticas e Hiperheurísticas

- Metaheurísticas
- Metaheurísticas parametrizadas
- Hiperheurísticas

2 Metaheurísticas paralelas

- Memoria Compartida
- Autooptimización en Memoria Compartida
- Paso de Mensajes
- Autooptimización de metaheurísticas con paso de mensajes
- Metaheurísticas en manycore

3 Ejemplos de aplicaciones

- Consumo de electricidad
- *Docking* de moléculas
- Análisis Envoltorio de Datos (DEA)
- Series temporales
- Determinación de componentes por análisis de sedimentos
- Sistemas Multicuerno

Contenidos

- 1 Metaheurísticas e Hiperheurísticas
 - Metaheurísticas
 - Metaheurísticas parametrizadas
 - Hiperheurísticas
- 2 Metaheurísticas paralelas
- 3 Ejemplos de aplicaciones
- 4 Créditos

Metaheurísticas

Descripción

- Utilizadas en **problemas de optimización** donde los métodos exactos requieren de gran coste computacional (NP-hard).
- **Estrategias generales** para la **búsqueda de soluciones** cercanas a la óptima dentro del espacio de soluciones.
- Tipos:
 - **Distribuidas o basadas en población**, capacidad de exploración. Algoritmos Genéticos, Búsqueda Dispersa, Colonias de Hormigas, Optimización de Partículas...
 - **Locales o basadas en trayectoria**, capacidad de explotación. Temple Simulado, Ascensión de Colinas, Búsqueda Tabú...
 - **Híbridas**, combinan características de varias

Algoritmos Genéticos

1970 J. H. Holland

Esquema de AG

InicializarPoblación(S)

mientras (**no** CondiciónDeFin(S))

$SS =$ SeleccionarElementos(S)

$SS1 =$ CruzarElementos(SS)

$SS2 =$ Mutar($SS1$)

$S =$ IncluirMejores($SS2$)

Descripción

- Generación de población aleatoria.
- Se seleccionan los mejores o con método de **ruleta** o **torneo**.
- Se cruzan por pares para generar dos descendientes.
- Algunos de ellos se mutan para diversificar.
- Se seleccionan los mejores para la población.

Ascensión de colinas

Esquema de AC

GenerarElemento(E)

mientras (**no** CondiciónDeFin(E))

$SS = \text{GenerarVecinos}(E)$

$E1 = \text{MejorVecino}(SS)$

si ($\text{mejor}(\text{valor}(E1), \text{valor}(E))$)

$E = E1$

Descripción

- Generación aleatoria de posible solución.
- Se seleccionan una **vecindad**
- y el mejor vecino dentro de ella.
- Si ese vecino mejora a la solución actual
- se sigue la búsqueda por él.
- Se puede **reiniciar** para evitar óptimos locales.

Esquema general

Esquema general de metaheurísticas

Inicializar(S)

mientras (**no** CondiciónDeFin(S))

$SS = \text{Seleccionar}(S)$

$SS1 = \text{Combinar}(SS)$

$SS2 = \text{Mejorar}(SS1)$

$S = \text{Incluir}(SS2)$

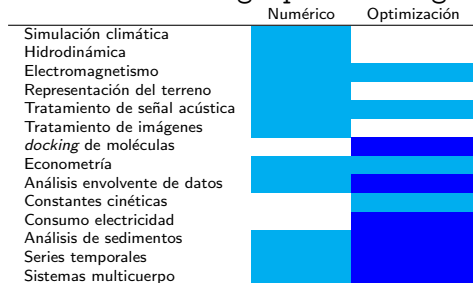
Ideas

- Se inicializa conjunto, población o elemento.
- Se seleccionan los mejores, todos, aleatorios, uno...
- Se combinan por pares, varios, se generan vecinos...
- Algunos elementos se mejoran analizando sus vecinos, mutando...
- Se incluyen los mejores, mejores y peores, el más prometedor...

Aplicaciones en el grupo de CCPP



luna.inf.um.es/grupo_investigacion



Con metaheurísticas empezamos en 2008, Tesis de Máster de José Ceferino Ortega.

José-Ceferino Ortega, Domingo Giménez, Alejandro Álvarez-Melcón, Fernando D. Quesada: Hybrid metaheuristics for the design of coupled resonator filters, Applied Artificial Intelligence, Vol 27, Issue 5, pp 323-350, 2013

Problemas en la aplicación de metaheurísticas

- Para un problema hay que probar varias metaheurísticas.
- Cada metaheurística tiene unos parámetros que hay que **tunear**.
- Problemas de programación para desarrollar las metaheurísticas
- y de tiempo de ejecución para probarlas y tunearlas.
- Necesidad de explotación de paralelismo para reducir el tiempo de ejecución.

Metaheurísticas parametrizadas:

Francisco Almeida, Domingo Giménez, Jose J. López-Espín and Melquíades Pérez-Pérez: Parameterised schemes of metaheuristics: basic ideas and applications with Genetic algorithms, Scatter Search and GRASP, IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, Vol 43, Issue 3, pp 570-586, 2013

Un esquema parametrizado de metaheurísticas

Esquema parametrizado

Inicializar(S , $ParamIni$)

mientras (**no** CondiciónDeFin(S , $ParamFin$))

$SS =$ Seleccionar(S , $ParamSel$)

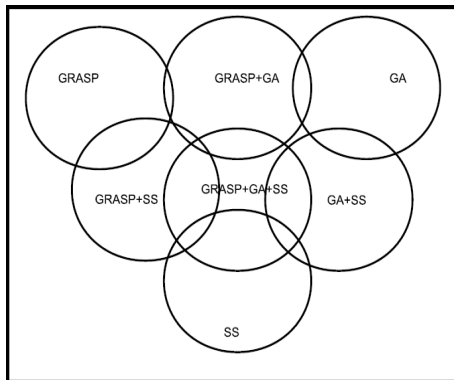
$SS1 =$ Combinar(SS , $ParamCom$)

$SS2 =$ Mejorar($SS1$, $ParamMej$)

$S =$ Incluir($SS2$, $ParamInc$)

- **Fácil selección** de diferentes **metaheurísticas** o combinaciones cambiando los parámetros $ParamX$.
- **Adaptación sencilla** de metaheurísticas a problemas específicos.
- **Paralelización simultánea** de distintas metaheurísticas si se paraleliza el esquema.
- Mismo esquema **aplicable a hiperheurísticas** para seleccionar metaheurísticas satisfactorias.

Espacio de metaheurísticas



- Varias metaheurísticas básicas
- e implementación de las funciones con parámetros
- da lugar a un espacio de metaheurísticas básicas e híbridas.

Inicialización

- Se generan aleatoriamente varias ($NIEI_{ni}$) soluciones.
GA muchas, SS (Scatter Search) menos, GRASP (Greedy randomized adaptive search procedure) varias para avance rápido.
- Pueden seleccionarse varias ($PEMI_{ni}$) para mejorarlas.
GA no se mejoran, SS y GRASP sí se mejoran.
- La mejora tiene un grado ($IMEI_{ni}$) de intensificación.
- Se selecciona un subconjunto ($NFEI_{ni}$) para las sucesivas iteraciones.
GA suele ser grande, SS menor y en búsqueda local un único elemento.

$$ParamI_{ni} = \{NIEI_{ni}, PEMI_{ni}, IMEI_{ni}, NFEI_{ni}\}$$

Condición de Fin

- Se establece un número máximo de iteraciones (*MNIFin*)
- y número máximo de iteraciones sin mejorar la mejor solución encontrada (*MIRFin*).
- Se puede establecer un tiempo máximo.

$$ParamFin = \{MNIFin, MIRFin\}$$

Seleccionar

- Se selecciona un número de los mejores elementos (*NMESel*)
- y otro de los peores (*NPESel*).
- En GA normalmente se seleccionan los mejores o por la ruleta con más probabilidad para los más prometedores.
- En SS todos, o los mejores y los más distantes de ellos (**dispersión**).

$$ParamSel = \{NMESel, NPESel\}$$

Combinar

- Se combina un número de pares de mejores elementos (*CMMCom*)
- de mejores con peores (*CMPCom*)
- y de peores (*CPPCom*).
- En GA combinaciones por pares por una posición de **cruce**. Algunas veces con elementos intermedios.
- En SS combinaciones entre todos, por pares o con más elementos.

$$ParamCom = \{CMMCom, CMPCom, CPPCom\}$$

Mejorar

- Se mejora un porcentaje ($PEMMej$) de los elementos generados
GA no se mejora, SS se mejoran todos.
- con una cierta intensificación en la mejora ($IMEMej$).
- Se puede diversificar un número de elementos ($NEDMej$)
En GA se corresponde con la mutación.
- y realizar mejora en ellos con un cierto grado de intensificación ($IMDMej$), para reducir la posibilidad de pérdida de los elementos obtenidos.

$$ParamMej = \{PEMMej, IMEMej, NEDMej, IMDMej\}$$

Incluir

- Se selecciona un número ($NMEInc$) de los mejores elementos para incluir en el conjunto de referencia para la siguiente iteración. En GA se seleccionan todos los mejores, en SS una parte.
- El resto hasta $NFEIni$ se seleccionan del resto. En SS se seleccionan los más “dispersos” respecto a los seleccionados, utilizando alguna medida de distancia.

$$ParamInc = \{NMEInc\}$$

Metaheurísticas con memoria

Algunas metaheurísticas incluyen memoria, por ejemplo Colonia de hormigas (1991, M. Dorigo) o Búsqueda tabú (1986, F. W. Glover). Incluir alguna de estas metaheurísticas en el esquema implica incluir parámetros para controlar esa memoria.

Ejemplo, Búsqueda Tabú:

- Memorias a corto plazo para la lista tabú en las tres búsquedas (determinan movimientos que no se toman):
MCPIni en la Inicialización.
MCMMej y *MCDMej* en la Mejora.
- Memoria a largo plazo para determinar elementos que se incluyen en el conjunto de referencia (se busca en zona del espacio alejada de las últimas búsquedas):
MLPInc

Ejemplos de parámetros I

Parámetros de metaheurísticas y combinaciones

		GR	TS	SS	GA	GR+TS	GR+SS	GR+GA	TS+SS	TS+GA
Ini	NEIIni	200	200	100	100	200	200	200	100	100
	NEFIni	1	1	20	100	1	20	100	20	100
	PEMIni	100	100	100	0	100	100	100	100	0
	IMEIni	50	10	50	0	50	50	50	50	0
	MCPIni	0	10	0	0	10	0	0	10	0
Sel	NEMSel	0	1	10	100	0	10	100	10	100
	NEPSel	0	0	10	0	0	25	0	10	0
Com	NMMCom	0	0	90	50	0	90	50	90	50
	NMPCom	0	0	100	0	0	100	0	100	0
	NPPCom	0	0	90	0	0	90	0	90	0
Mej	PEMMej	0	100	100	0	0	100	0	100	0
	IMEMej	0	5	5	0	0	5	0	5	0
	MCMMej	0	10	0	0	0	0	0	10	0
	PEDMej	0	0	0	10	0	0	10	0	10
	IDEMej	0	0	0	5	0	0	5	0	5
	MCDMej	0	0	0	0	0	0	0	0	10
Inc	NEMInc	0	1	10	100	0	10	100	10	100
	MLPInc	0	30	0	0	0	0	0	30	30

Ejemplos de parámetros II

Parámetros de metaheurísticas y combinaciones

	SS+GA	GR+TS+SS	GR+TS+GA	GR+SS+GA	TS+SS+GA	GR+TS+SS+GA
NEIIni	100	200	200	200	100	200
NEFIIni	50	20	100	50	50	50
PEMIIni	100	100	100	100	100	100
IMEIIni	50	50	50	50	50	50
MCPIni	0	10	10	0	10	10
NEMSel	25	10	100	25	25	25
NEPSel	25	25	0	25	25	25
NMMCom	90	90	50	90	90	90
NMPCom	100	100	0	100	100	100
NPPCom	90	90	0	90	90	90
PEMMej	100	100	0	100	100	100
IMEMej	5	5	0	5	5	5
MCMMej	0	10	0	0	10	10
PEDMej	10	0	10	10	10	10
IDEMej	5	0	5	5	5	5
MCDMej	0	0	10	0	10	10
NEMInc	25	10	100	25	25	25
MLPInc	0	30	30	0	30	30

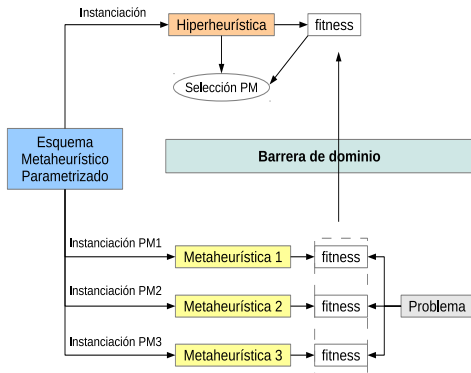
Ideas generales

- Dada una implementación de metaheurísticas y una serie de parámetros en el esquema
- ¿cómo obtener metaheurísticas (conjuntos de parámetros metaheurísticos) satisfactorias para nuestro problema?
En implementación con GA+SS+GRASP 16 parámetros,
con GA+SS+TS+GRASP 20 parámetros.

Hiperheurísticas:

- Buscan en el espacio de posibles parámetros metaheurísticos,
- lo que quiere decir de posibles metaheurísticas implementadas con el esquema unificado parametrizado.
- Y se pueden implementar a su vez como metaheurísticas siguiendo el mismo esquema.

Hiperheurísticas - esquema de funcionamiento



Descripción

- Misma estructura e implementación similar a la de las metaheurísticas pero con **mayor abstracción**: se trabaja **sobre** el espacio de metaheurísticas.
- **Selección automática** de parámetros metaheurísticos.
- Separación de dominios: sólo flujo de información sobre fitness o coste.
- **Evitan la dependencia del problema**, encontrando la mejor solución para un conjunto de instancias.
- **Tiempo de ejecución mucho mayor** (aplicación repetida de metaheurísticas).

Francisco Almeida, José-Matías Cutillas-Lozano, Domingo Giménez: Hyperheuristics based on parameterized metaheuristic schemes, Genetic and Evolutionary Computation Conference, Madrid, Spain, July 11-15, 2015

Diferencias en el uso del esquema en Metaheurísticas e Hiperheurísticas

Características

	Metaheurísticas	Hiperheurísticas
Problema de optimización	Problema dado de ciencias o ingeniería.	Selección de la metaheurística óptima para cada problema o conjunto de ellos.

Diferencias en el uso del esquema en Metaheurísticas e Hiperheurísticas

Características

	Metaheurísticas	Hiperheurísticas
Problema de optimización	Problema dado de ciencias o ingeniería.	Selección de la metaheurística óptima para cada problema o conjunto de ellos.
Individuo	Vector de enteros o reales con valores de variables específicos para cada problema de optimización.	Vector de enteros <i>MetaheurParam</i> que recoge los parámetros de cada metaheurística.

Diferencias en el uso del esquema en Metaheurísticas e Hiperheurísticas

Características

	Metaheurísticas	Hiperheurísticas
Problema de optimización	Problema dado de ciencias o ingeniería.	Selección de la metaheurística óptima para cada problema o conjunto de ellos.
Individuo	Vector de enteros o reales con valores de variables específicos para cada problema de optimización.	Vector de enteros <i>MetaheurParam</i> que recoge los parámetros de cada metaheurística.
Conjunto de Referencia	Conjunto de elementos de cada problema.	Conjunto de metaheurísticas o combinaciones aplicables.

Diferencias en el uso del esquema en Metaheurísticas e Hiperheurísticas

Características

	Metaheurísticas	Hiperheurísticas
Problema de optimización	Problema dado de ciencias o ingeniería.	Selección de la metaheurística óptima para cada problema o conjunto de ellos.
Individuo	Vector de enteros o reales con valores de variables específicos para cada problema de optimización.	Vector de enteros <i>MetaheurParam</i> que recoge los parámetros de cada metaheurística.
Conjunto de Referencia	Conjunto de elementos de cada problema.	Conjunto de metaheurísticas o combinaciones aplicables.
Fitness	Función objetivo específica del problema.	Calculado con valor de la función objetivo cuando se aplica a instancias del problema.

Diferencias en el uso del esquema en Metaheurísticas e Hiperheurísticas

Características

	Metaheurísticas	Hiperheurísticas
Problema de optimización	Problema dado de ciencias o ingeniería.	Selección de la metaheurística óptima para cada problema o conjunto de ellos.
Individuo	Vector de enteros o reales con valores de variables específicos para cada problema de optimización.	Vector de enteros <i>MetaheurParam</i> que recoge los parámetros de cada metaheurística.
Conjunto de Referencia	Conjunto de elementos de cada problema.	Conjunto de metaheurísticas o combinaciones aplicables.
Fitness	Función objetivo específica del problema.	Calculado con valor de la función objetivo cuando se aplica a instancias del problema.
Objetivo	Obtención del mejor individuo (conjunto de variables del problema científico dado).	Obtención de la combinación óptima de parámetros metaheurísticos.

Diferencias en el uso del esquema en Metaheurísticas e Hiperheurísticas

Características

	Metaheurísticas	Hiperheurísticas
Problema de optimización	Problema dado de ciencias o ingeniería.	Selección de la metaheurística óptima para cada problema o conjunto de ellos.
Individuo	Vector de enteros o reales con valores de variables específicos para cada problema de optimización.	Vector de enteros <i>MetaheurParam</i> que recoge los parámetros de cada metaheurística.
Conjunto de Referencia	Conjunto de elementos de cada problema.	Conjunto de metaheurísticas o combinaciones aplicables.
Fitness	Función objetivo específica del problema.	Calculado con valor de la función objetivo cuando se aplica a instancias del problema.
Objetivo	Obtención del mejor individuo (conjunto de variables del problema científico dado).	Obtención de la combinación óptima de parámetros metaheurísticos.
Coste Computacional	Bajo-Medio.	Alto.

Diferencias en el uso del esquema en Metaheurísticas e Hiperheurísticas

Características

	Metaheurísticas	Hiperheurísticas
Problema de optimización	Problema dado de ciencias o ingeniería.	Selección de la metaheurística óptima para cada problema o conjunto de ellos.
Individuo	Vector de enteros o reales con valores de variables específicos para cada problema de optimización.	Vector de enteros <i>MetaheurParam</i> que recoge los parámetros de cada metaheurística.
Conjunto de Referencia	Conjunto de elementos de cada problema.	Conjunto de metaheurísticas o combinaciones aplicables.
Fitness	Función objetivo específica del problema.	Calculado con valor de la función objetivo cuando se aplica a instancias del problema.
Objetivo	Obtención del mejor individuo (conjunto de variables del problema científico dado).	Obtención de la combinación óptima de parámetros metaheurísticos.
Coste Computacional	Bajo-Medio.	Alto.
Paralelismo y autooptimización	Técnica de selección automática de parámetros de paralelismo.	Misma técnica que para metaheurísticas.

Diferencias en el uso del esquema en Metaheurísticas e Hiperheurísticas

Características

	Metaheurísticas	Hiperheurísticas
Problema de optimización	Problema dado de ciencias o ingeniería.	Selección de la metaheurística óptima para cada problema o conjunto de ellos.
Individuo	Vector de enteros o reales con valores de variables específicos para cada problema de optimización.	Vector de enteros <i>MetaheurParam</i> que recoge los parámetros de cada metaheurística.
Conjunto de Referencia	Conjunto de elementos de cada problema.	Conjunto de metaheurísticas o combinaciones aplicables.
Fitness	Función objetivo específica del problema.	Calculado con valor de la función objetivo cuando se aplica a instancias del problema.
Objetivo	Obtención del mejor individuo (conjunto de variables del problema científico dado).	Obtención de la combinación óptima de parámetros metaheurísticos.
Coste Computacional	Bajo-Medio.	Alto.
Paralelismo y autooptimización	Técnica de selección automática de parámetros de paralelismo.	Misma técnica que para metaheurísticas.

Rangos de búsqueda de la hiperheurística

Se establecen límites “razonables” para los parámetros metaheurísticos a seleccionar por las hiperheurísticas. En algunos experimentos hemos considerado:

	NEIIni	NEFIIni	PEMIIni	IMEIIni	MCPIIni	NEMSel
Inferior	5	5	0	1	0	2
Superior	200	100	100	20	15	100

	NEPSel	NMMCom	NMPCom	NPPCom	PEMMej	IMEMej
Inferior	2	5	5	5	0	1
Superior	100	100	100	100	100	20

	MCMMej	PEDMej	IDEMej	MCDMej	NEMInc	MLPInc
Inferior	0	0	1	0	2	0
Superior	15	100	10	15	100	15

Parámetros metaheurísticos de las hiperheurísticas

Parámetros hiperheurísticos utilizados en algunos experimentos

	NEIIni	NEFIni	PEMIni	IMEIni	MCPIni	NEMSel
Hhi	20	20	50	5	5	10
Hre	5	5	50	3	2	3
Hge	20	20	0	0	0	20

	NEPSel	NMMCom	NMPCom	NPPCom	PEMMej	IMEMej
Hhi	10	15	20	15	50	5
Hre	2	2	3	2	50	3
Hge	0	10	0	0	0	0

	MCMMej	PEDMej	IDEMej	MCDMej	NEMInc	MLPInc
Hhi	5	10	5	5	10	5
Hre	2	10	5	2	3	5
Hge	0	10	5	0	20	0

- **Hhi**: hiperheurística híbrida de tamaño medio, con poblaciones y mejoras moderadas y no muy intensas, y con un número de elementos a combinar moderado.
- **Hre**: hiperheurística híbrida reducida. Tiene los mismos parámetros que Hhi pero con valores más bajos para muchos de ellos con el fin de reducir el tiempo de ejecución.
- **Hge**: hiperheurística con un enfoque de algoritmo genético, población pequeña y mejora solo en la diversificación. Los valores de los parámetros Tabú se establecen en cero.

Contenidos

- 1 Metaheurísticas e Hiperheurísticas
- 2 Metaheurísticas paralelas
 - Memoria Compartida
 - Autooptimización en Memoria Compartida
 - Paso de Mensajes
 - Autooptimización de metaheurísticas con paso de mensajes
 - Metaheurísticas en manycore
- 3 Ejemplos de aplicaciones
- 4 Créditos

Necesidad del paralelismo

- Experimentar con varias metaheurísticas.
- Muchos experimentos para “tunear” los parámetros
- Algunas veces necesarios muchos elementos o muchas iteraciones,
- y el coste de la función de fitness puede ser alto.
- En el caso de hiperheurísticas evaluación de muchas metaheurísticas.

⇒ es necesario **paralelismo**

Enrique Alba (editor): Parallel Metaheuristics. A New Class of Algorithms. Wiley. 2005.

Paralelización del esquema en memoria compartida

Esquema metaheurístico parametrizado en memoria compartida

Inicializar(S , $ParamIni$, $HilosIni$)

mientras (**no** CondiciónDeFin(S , $ParamFin$, $HilosFin$))

SS = Seleccionar(S , $ParamSel$, $HilosSel$)

$SS1$ = Combinar(SS , $ParamCom$, $HilosCom$)

$SS2$ = Mejorar($SS1$, $ParamMej$, $HilosMej$)

S = Incluir($SS2$, $ParamInc$, $HilosInc$)

- Paralelización independiente de las funciones con **parámetros de paralelismo** (número de hilos) para cada función. Paralelismo de **grano fino**.
- El valor óptimo de los **parámetros de paralelismo** depende de los valores de los **parámetros metaheurísticos**.

Francisco Almeida, Domingo Giménez, Jose J. López-Espín: A Parametrized Shared-Memory Scheme for Parametrized Metaheuristics, The Journal of Supercomputing, Vol 58, N 3, pp 292-301, 2011

Autooptimización en el esquema de Memoria Compartida

Funciones de un nivel de paralelismo

```
omp_set_num_threads(threads - one - level(MetaheurParam))
#pragma omp parallel for
loop in elements
  treat element
```

por ejemplo: Inicializar, Combinar...

Modelo de tiempos en memoria compartida. Funciones de un nivel de paralelismo

$$t_{1-nivel} = \frac{k_{s1} \cdot NE}{p} + k_p \cdot p$$

$$p_{opt.} = \sqrt{\frac{k_{s1} \cdot NE}{k_p}}$$

Constantes del sistema y parámetros metaheurísticos en el modelo

$F_{1-nivel}$	k_{s1}	NE
Gen-Ini	k_g	NEI_{ni}
Combinar	k_c	$2 \cdot (NMMCom + NMPCom + NPPCom)$
Incluir	k_i	$NEFI_{ni} + 2 \cdot (NMMCom + NMPCom + NPPCom) - NEMI_{inc}$

Autooptimización en el esquema de Memoria Compartida

Funciones con dos niveles de paralelismo

first-level(MetaheurParam):

```
omp_set_num_threads(threads - first - level(MetaheurParam))
#pragma omp parallel for
loop in elements
  second-level(MetaheurParam, threads - first - level)
```

second-level(MetaheurParam, threads - first - level):

```
omp_set_num_threads(threads - second - level(MetaheurParam, threads - first - level))
#pragma omp parallel for
loop in neighbors
  treat neighbor
```

aparecen en las mejoras

Funciones de dos niveles de paralelismo

$$t_{2-niveles} = \frac{k_{s2} \cdot Param}{p_1 \cdot p_2} + k_{p,1} \cdot p_1 + k_{p,2} \cdot p_2$$

$$p_{1,opt.} = \sqrt[3]{\frac{k_{s2} \cdot k_{p,2}}{k_{p,1}^2} \cdot Param}$$

$$p_{2,opt.} = \sqrt[3]{\frac{k_{s2} \cdot k_{p,1}}{k_{p,2}^2} \cdot Param}$$

Constantes del sistema y parámetros metaheurísticos en el modelo

$F_{2-niveles}$	k_{s2}	NE	PM	IM
Mej-Ini	k_{mi}	NEIni	PEMIni	IMEIni
Mej-Ref	k_{mr}	NEFIni	PEMMej	IMEMej
Mej-Com	k_{mc}	$2 \cdot (NMMCom + NMPCom + NPPCom)$	PEMMej	IMEMej
Div	k_d		PEDEj	IDEMej

Metaheurísticas con paso de mensajes

- El esquema más usado es el de **islas**: cada proceso trabaja con un subconjunto y cada cierto número de iteraciones (**generaciones**) se intercambian (**migran**) elementos. Paralelismo de **grano grueso**.
- Más generaciones entre intercambios implican menos comunicaciones,
- pero la convergencia puede ser peor debido a **endogamia**.
- Los sistemas pueden ser homogéneos o heterogéneos, lo que dificulta más la programación.

José-Matías Cutillas-Lozano, Domingo Giménez: Optimizing a parameterized message-passing metaheuristic scheme on a heterogeneous cluster, *Soft Computing* 21(19): 5557-5572 (2017)

Paralelización del esquema con paso de mensajes

Esquema de paso de mensajes. Modelo de Islas($S, \text{ParamPar}$)

EN PARALELO en cada proceso P_i ($i = 0, \dots, p - 1$) HACER
 Inicializar($S_i, \text{ParamIni}$)
mientras (**no** CondiciónDeFin($\text{ParamFin}, \text{NGMPar}$)) **Hacer**
 Esquema_Metaheurístico_Secuencial(S_i, NGMPar)
 Inmigrar(S_i, S_0, NEMPar)
 Si $i = 0$ **Entonces**
 Integrar Subpoblaciones(S_0)
 Fin si
 Emigrar(S_0, S_i, NEMPar)
Fin Mientras
 FIN PARALELO
 Solución: mejor $s_k \in S_0$

Esquema Metaheurístico Secuencial(S_i, NGMPar)

mientras (**no** CondiciónDeMigración(NGMPar))
Hacer
 $SS_i = \text{Seleccionar}(S_i, \text{ParamSel})$
 $SS1_i = \text{Combinar}(SS_i, \text{ParamCom})$
 $SS2_i = \text{Mejorar}(SS1_i, \text{ParamMej})$
 $S_i = \text{Incluir}(SS2_i, \text{ParamInc})$
Fin Mientras

Descripción

- Partición de datos homogénea entre procesos (islas o subpoblaciones): $|S_i| = \frac{|S|}{p}$, con $S = S_0 \cup \dots \cup S_{p-1}$.
- Nuevos parámetros:
 - p : Número de procesos.
 - NEMPar : Número de individuos a migrar, igual en inmigraciones y emigraciones.
 - NGMPar : Número de generaciones en cada isla antes de migrar.
- Migraciones entre trabajadores y maestro, no comunicación entre trabajadores. Hay otras posibilidades.

Modelo de tiempos con paso de mensajes

Ecuaciones del modelo

$$t_{total} = t_{cmp} + t_{cmc} + t_{ord}, \quad \text{con } t_{ord} \approx 0$$

$$t_{cmp} = \sum_{i=1}^2 k_i \cdot \frac{Param_i}{p} + \left(\sum_{j=1}^6 k_j \cdot \frac{Param_j}{p} \right) \cdot NGMPar$$

$$t_{cmc} = t_{rec} + t_{dif}, \quad \text{con } t_{dif} \approx 0$$

$$t_{rec} = A \cdot p^3 + B \cdot p^2 + C \cdot p + D$$

Cálculo de p_{optimo} , despejando...

$$\frac{dt_{total}}{dp} = \frac{-K_{cmp}}{p^2} + 3 \cdot A \cdot p^2 + 2 \cdot B \cdot p + C = 0, \quad \text{con}$$

$$K_{cmp} = \sum_{i=1}^2 k_i \cdot Param_i + \left(\sum_{j=1}^6 k_j \cdot Param_j \right) \cdot NGMPar$$

Constantes del sistema y parámetros metaheurísticos en el modelo

<i>func</i>	k_j	<i>Param_j</i>
Com	k_c	$2 \cdot (NMMCom + NMPCom + NPPCom)$
Mej-Ref	k_{mr}	$\frac{NEFIni \cdot PEMMej \cdot IMEMej}{100}$
Mej-Com	k_{mc}	$\frac{(NMMCom + NMPCom + NPPCom) \cdot PEMMej \cdot IMEMej}{100}$
Div-Ref	k_{dr}	$\frac{NEFIni \cdot PEDMej \cdot IDEMej}{100}$
Div-Com	k_{dc}	$\frac{(NMMCom + NMPCom + NPPCom) \cdot PEDMej \cdot IDEMej}{100}$
Inc	k_i	$NEFIni + 2 \cdot (NMMCom + NMPCom + NPPCom) - NEMInc$

GPU

- Para desarrollar esquema metaheurístico parametrizado para GPU
 - Se pueden identificar las funciones con mayor coste computacional y lanzar kernels para ejecutarlas en GPU.
 - O utilizar esquema de islas asignando subpoblaciones con tamaño distintos a elementos computacionales de distinta capacidad.

Algunos experimentos con cálculo de fitness en GPU en

Baldomero Imbernón, José M. Cecilia, Horacio Pérez Sánchez and Domingo Giménez: METADOCK: A Parallel Metaheuristic schema for Virtual Screening methods. *Journal of High Performance Computing Applications*, 32 (6) 2018

Esquema para multicore+multiGPU

```
InitializeCPU(Sini,ParamIni)
ComputefitnessGPU(Sini,ParamIni)
ImproveGPU(Sini,Sref,ParamImplni) //dos posibles niveles
mientras (no EndConditionCPU(Sref,ParamEndCon)
  SelectCPU(Sref,Ssel,ParamSel)
  CombineCPU(Ssel,Scom,ParamCom)
  ComputefitnessGPU(Scom,ParamCom)
  DiversifyCPU(Sref,Scom,Sdiv,ParamDiv)
  ComputefitnessGPU(Sdiv,ParamDiv)
  ImproveGPU(Scom,Sdiv,ParamImp) //dos niveles
  IncludeCPU(Scom,Sdiv,Sref,ParamInc)
```

Fin Mientras

Aceleración del acoplamiento de moléculas en

Baldomero Imbernón, Antonio Llanes, José-Matías Cutillas-Lozano and Domingo Giménez: HYPERDOCK: improving Virtual Screening through parallel hyperheuristics. Admitted in Journal of High Performance Computing Applications

Esquema para multicore+multiGPU, resultados

	COMT			HIVPR		
	H1	H2	H3	H1	H2	H3
	venus					
multicore	3,765	1,968	4,444	3,745	2,718	3,559
multicore+GPU	4,217	2,070	3,471	4,744	3,255	5,840
multicore/hybrid	0.89	0.95	1.28	0.79	0.84	0.61
	jupiter					
multicore	5,750	6,258	6,508	7,046	8,167	9,517
multicore+GPU	879	1,014	894	945	1,292	1,247
multicore/hybrid	6.54	6.17	7.28	7.46	6.32	7.63
	saturno					
multicore	5,162	4,642	5,784	7,109	5,918	9,846
multicore+GPU	342	299	841	557	536	645
multicore/hybrid	15.09	15.53	6.88	12.76	11.04	15.27
	heterolistic					
multicore	2,419	1,010	3,754	2,777	1,525	3,055
multicore+GPU	320	114	565	332	143	609
multicore/hybrid	7.55	8.85	6.64	8.36	10.66	5.03
	power8					
multicore	3,544	1,334	4,049	4,998	2,596	9,976
multicore+GPU	159	75	361	167	107	355
multicore/hybrid	22.27	17.65	11.21	29.82	24.18	28.06

GPU virtualizadas

- En un cluster con nodos con GPUs, rCUDA proporciona virtualización de las GPUs.
- Desde un nodo se pueden lanzar trabajos a todas las GPUs.
- Mejores prestaciones que con paso de mensajes explícito con MPI.
- Adaptación de técnicas de reparto heterogéneas, con islas de distinto tamaño según capacidad de las GPUs.

Baldomero Imbernón, Javier Prades, Domingo Giménez, José M. Cecilia and Federico Silla: Enhancing large-scale docking simulation on heterogeneous systems: an MPI vs rCUDA study. Future Generation Computer Systems. Published on line, September 2017

Xeon Phi

- MIC con 57 cores y 4 threads por core por hardware.
- Estilo de programación como en multicore: se pueden reusar los programas de OpenMP para memoria compartida.
- Paralelismo masivo, por lo que es adecuado para explotar el paralelismo multinivel.

Explotación de paralelismo en hiperheurísticas, con 2 niveles en las metaheurísticas y 2 en la hiperheurística:

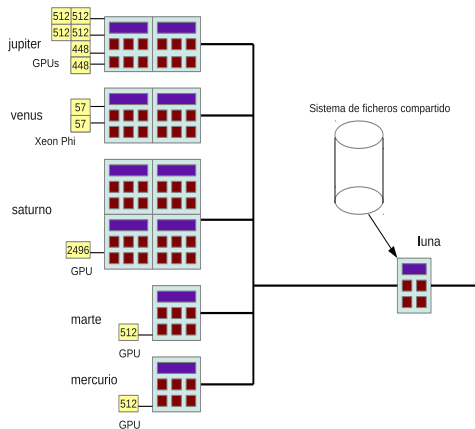
José M. Cecilia, José-Matías Cutillas-Lozano, Domingo Giménez, Baldomero Imbernón: Exploiting Multi-level Parallelism on a Many-core System for the Application of Hyperheuristics to a Molecular Docking Problem. *Journal of Supercomputing*. 74 (5) 1803-1814, 2018

Contenidos

- 1 Metaheurísticas e Hiperheurísticas
- 2 Metaheurísticas paralelas
- 3 Ejemplos de aplicaciones
 - Consumo de electricidad
 - *Docking* de moléculas
 - Análisis Envoltante de Datos (DEA)
 - Series temporales
 - Determinación de componentes por análisis de sedimentos
 - Sistemas Multicuerpo
- 4 Créditos

Sistema computacional

Cluster del grupo de CCPP de la UMU



y algunos otros

Tesis José Matías Cutillas

José Matías Cutillas Lozano: Modelado y autooptimización de metaheurísticas e hiperheurísticas parametrizadas paralelas aplicadas a problemas de optimización en ciencia e ingeniería. Tesis Doctoral, Universidad de Murcia, Septiembre 2014. luna.inf.um.es/grupo_investigacion

- Metaheurísticas parametrizadas
- Hiperheurísticas sobre el esquema parametrizado
- Implementación en memoria compartida
- Con esquema de islas en paso de mensajes
- Adaptación a cluster heterogéneo
- Autooptimización de MC y PM
- Problemas de **consumo de electricidad** y de **determinación de constantes cinéticas**

Problema de consumo de electricidad en pozos de agua

- Conjunto de pozos de agua con bombas (B) de potencia conocida, que extraen flujos de agua durante un intervalo de tiempo diario R . El caudal total es la suma de los caudales aportados por cada pozo.
- Las bombas están en funcionamiento o fuera de servicio en cada intervalo de tiempo considerado. Las bombas operan eléctricamente y la electricidad tiene un coste diario que debería ser minimizado. La función objetivo es:

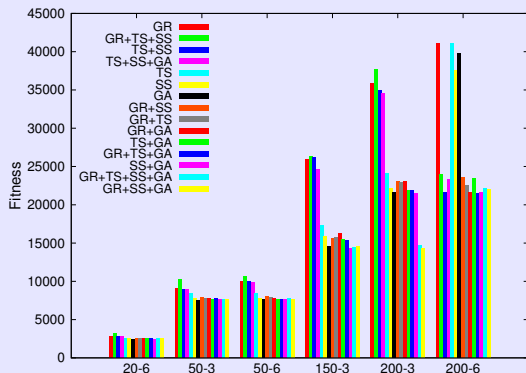
$$\text{Minimizar } C_e = \sum_{i=1}^R \sum_{j=1}^B T_i P_j N_i x_{ij}$$

con una serie de restricciones físicas

- C_e : coste de la electricidad consumida por la combinación de bombas seleccionadas en un día
- T_i : coste de la electricidad en un rango i
- P_j : potencia eléctrica consumida por la bomba j
- N_i : número de horas de bombeo en el intervalo de tiempo i
- x_{ij} : elemento binario de una matriz: 1 o 0 para bomba encendida o apagada
- Un **individuo** o elemento se representa por la **matriz binaria** x , de **tamaño** $B \cdot R$, que codifica el conjunto de **bombas** distribuidas en diferentes **rangos horarios**.

Resultados experimentales -fitness

Fitness alcanzado para diferentes combinaciones de metaheurísticas y tamaños del problema PCEPA

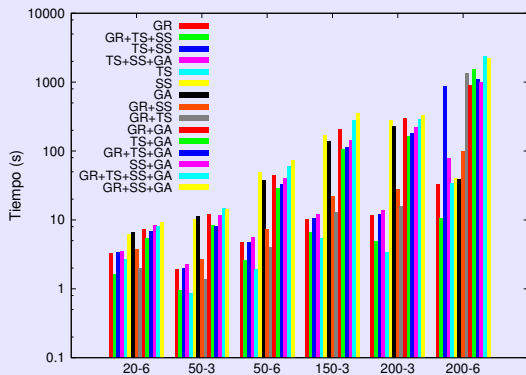


Análisis resultados

- Mejores fitness (más bajos): GA y combinaciones que incluyen SS+GA.
- Pobres resultados con GRASP que mejoran al hibridarse con otras.
- TS arroja resultados intermedios, su hibridación con SS no da buenos resultados.
- La hibridación de las cuatro metaheurísticas es satisfactoria aunque no la mejor.

Resultados experimentales - tiempo

Tiempos de ejecución

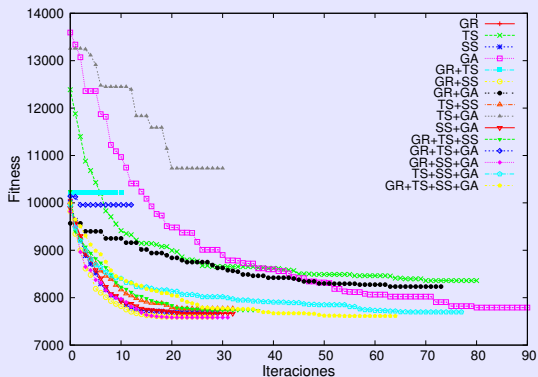


Análisis resultados

Mejores resultados de tiempo de ejecución: TS, GRASP+TS y GRASP+TS+SS.

Resultados experimentales - convergencia

Evolución del fitness en iteraciones sucesivas para PCEPA 50-6



Análisis resultados

- SS alcanza muy buenos resultados en muy pocas iteraciones.
- Combinaciones incluyendo SS+GA: idóneas para PCEPA 50-6 desde el punto de vista del fitness.

Parámetros metaheurísticos obtenidos al aplicar hiperheurísticas a PCEPA

Aplicación de Hhi, Hre y Hge a instancias de PCEPA 10-3

	NEINi	NEFINi	PEMINi	IMEINi	MCPINi	NEMSel
HhiFit1P1E	8	8	50	16	14	8
HhiFitVPVE2	39	37	54	12	9	33
HreFit1P1E	111	60	6	16	8	2
HreFitVPVE2	106	39	26	11	10	19
HgeFit1P1E	122	43	44	13	7	43
HgeFitVPVE2	82	49	57	6	9	26

	NEPSel	NMMCom	NMPCom	NPPCom	PEMMej	IMEMej
HhiFit1P1E	0	37	0	0	16	7
HhiFitVPVE2	4	33	22	5	27	12
HreFit1P1E	58	30	69	94	94	10
HreFitVPVE2	19	44	23	31	72	13
HgeFit1P1E	0	99	0	0	40	11
HgeFitVPVE2	23	94	21	29	36	10

	MCMMej	PEDMej	IDEMej	MCDMej	NEMInc	MLPInc
HhiFit1P1E	4	60	4	9	8	9
HhiFitVPVE2	5	50	6	9	19	12
HreFit1P1E	13	49	2	8	15	2
HreFitVPVE2	7	71	4	7	22	2
HgeFit1P1E	1	97	9	8	43	5
HgeFitVPVE2	8	89	7	5	49	6

Discusión parámetros

- Valores intermedios del parámetro *NEINi*.
- Porcentaje de mejora moderado en muchos casos, con una intensidad de la mejora con valores medios o altos en el rango considerado.
- Se ha dado prioridad a la combinación de los mejores elementos.
- La diversificación ha resultado importante para este problema, y el número de mejores elementos a incluir en el conjunto de referencia es alto.

Aplicación hiperheurísticas a PCEPA

Aplicación directa hiperheurísticas a PCEPA 10-3 (Fit1P1E)

	Hhi	Hre	Hge
fitness (€)	2047.81	2047.81	2047.81
tiempo (s)	49705	5212	1517
IC	9.82	93.69	321.9

Aplicación de las mejores metaheurísticas obtenidas con tres configuraciones hiperheurísticas a PCEPA 20-6, 40-3 y 40-6 (valores medios)

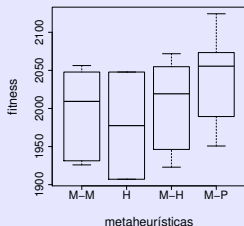
		Hhi	Hre	Hge	Media
Fit1P1E	(f)	4065.60	4190.89	4322.45	4192.98
	(t)	2.80	25.76	22.91	17.16
	(IC)	8.78	0.93	1.01	3.57
FitVPVE2	(f)	4117.35	4174.70	4161.14	4151.06
	(t)	11.26	22.23	20.42	17.97
	(IC)	2.16	1.08	1.18	1.47
FitVP1E	(f)	4062.84	4118.14	3966.74	4049.24
	(t)	23.25	41.53	18.06	27.61
	(IC)	1.06	0.58	1.40	1.01
Media	(f)	4081.93	4161.24	4150.11	
	(t)	12.44	29.84	20.46	
	(IC)	4.00	0.86	1.19	

Discusión resultados

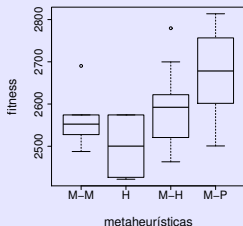
- Hre y Hge mucho más rápidas que Hhi y fitness iguales en la aplicación directa de hiperheurísticas.
- Las metaheurísticas obtenidas con Hhi mejoran, de media, a Hre y Hge en fitness e IC.
- El fitness obtenido con FitVP1E mejora los obtenidos con Fit1P1E y FitVPVE2.
- El mejor IC viene de Fit1P1E (metaheurística menos efectiva con tiempo de ejecución más reducido).

Aplicación hiperheurísticas a PCEPA. Resumen estadístico

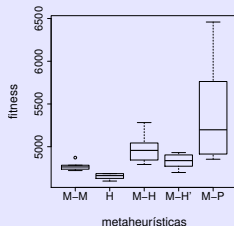
Resumen estadístico del fitness



PCEPA 10-x



PCEPA 20-x



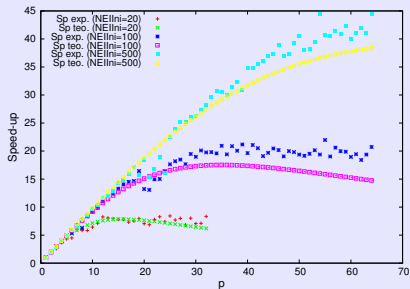
PCEPA 40-x

Conjuntos de algoritmos

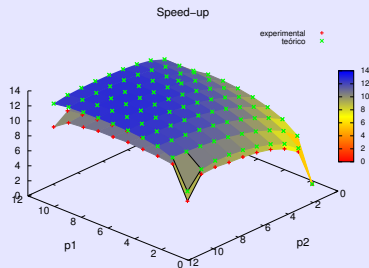
- M-M: mejores metaheurísticas no seleccionadas automáticamente.
- H: aplicación directa de Hhi, Hre y Hge.
- M-H y M-H': conjunto de metaheurísticas obtenidas a partir de las hiperheurísticas, con computación del fitness agrupado en Fit1P1E y FitVPVE2 (M-H) y FitVP1E (M-H').
- M-P: conjunto de las cuatro metaheurísticas puras.

Resultados experimentales. Modelado particular funciones

Speed-ups teórico y experimental variando el número de hilos del primer (y segundo nivel) de paralelismo para rutinas de uno y dos niveles de paralelismo en *Ben* (MC 128 cores)



EM, rutina un-nivel



HEM, rutina dos-niveles

Análisis resultados

El modelo de tiempos en el **esquema metaheurístico (EM)** predice bien el comportamiento de la función de generación inicial de elementos para varios valores de $NElNi$, así como en las **hiperheurísticas basadas en el esquema metaheurístico (HEM)** para la función de mejora de la población inicial con $NElNi = 50$, $PElNi = 50$ y $IMElNi = 1$.

Resultados experimentales. Modelado particular funciones

Hilos y speed-ups experimentales y modelados. Aplicación del EM a PCEPA en *Ben*

<i>NEIIn</i>	hilos		speed-up		
	exp	mod	exp	mod	exp-auto
100	48	55	27	27	25
500	121	122	77	61	75

EM, rutina un-nivel

<i>NEIIn</i>	<i>PEMIn</i>	<i>IMEIn</i>	hilos		speed-up		
			exp	mod	exp	mod	exp-auto
100	50	10	89	67	35	17	21
500	100	5	128	150	78	51	78

EM, rutina dos-niveles

Aplicación del HEM a PCEPA en *Ben*. Combinaciones de parámetros *NEIIn*, *PEMIn* y *IMEIn* (*c1*: 50,50,1; *c2*: 100,50,1)

<i>NEIIn</i>	hilos		speed-up		
	exp	mod	exp	mod	exp-auto
20	22	15	11	8	8
100	24	34	12	17	12

HEM, rutina un-nivel

Param	hilos		speed-up		
	exp	mod	exp	mod	exp-auto
<i>c1</i>	9×8	6×7	14	15	11
<i>c2</i>	9×4	8×9	15	24	14

HEM, rutina dos-niveles

Variables

- **exp**: speed-up experimental más alto.
- **mod**: speed-up modelado.
- **exp-auto**: speed-up experimental con el número de hilos seleccionado con la metodología de autooptimización.

Análisis resultados

Al aplicar EM y HEM, el número de hilos y el speed-up seleccionados con la técnica de autooptimización cercanos a los mejores experimentales.

Resultados experimentales. Modelado particular funciones

Aplicación del EM a PCEPA en *Ben*. Combinaciones de parámetros *NEIIni*, *PEMI* y *IMEI* (c1: 100,50,10; c2: 500,100,5)

<i>NEIIni</i>	hilos				speed-up			
	max	$\frac{max}{2}$	exp	mod	max	$\frac{max}{2}$	exp	exp-auto
100	128	64	48	55	20	23	27	25
500	128	64	121	122	73	49	77	75

EM, rutina un-nivel

Param	hilos				speed-up			
	max	$\frac{max}{2}$	exp	mod	max	$\frac{max}{2}$	exp	exp-auto
c1	128	64	89	67	27	15	35	21
c2	128	64	128	150	78	52	78	78

EM, rutina dos-niveles

Variables

- **max**: speed-up obtenido con el número máximo de hilos disponible.
- $\frac{max}{2}$: speed-up obtenido con la mitad del número máximo de hilos disponible.
- **exp**: speed-up experimental más alto.
- **exp-auto**: speed-up experimental con el número de hilos seleccionado (**mod**) con la metodología de autooptimización.

Resultados experimentales. Modelado particular funciones

Aplicación del HEM a PCEPA en *Ben*. Combinaciones de parámetros NEI_{ini} , PEM_{ini} y IME_{ini} (c1: 50,50,1; c2: 100,50,1)

NEI_{ini}	hilos				speed-up			
	max	$\frac{max}{2}$	exp	mod	max	$\frac{max}{2}$	exp	exp-auto
20	128	64	22	15	11	11	11	8
100	128	64	24	34	11	11	12	12

HEM, rutina un-nivel

Param	hilos				speed-up			
	\sqrt{max}	$\sqrt{\frac{max}{2}}$	exp	mod	\sqrt{max}	$\sqrt{\frac{max}{2}}$	exp	exp-auto
c1	11×11	8×8	9×8	6×7	9	11	14	11
c2	11×11	8×8	9×4	8×9	11	12	15	14

HEM, rutina dos-niveles

Análisis resultados

El speed-up alcanzado con el modelo es casi siempre mejor que el obtenido al ejecutar de manera no óptima el máximo número de hilos disponible o su mitad, y se acerca al valor óptimo experimental en muchos casos.

Resultados experimentales. Modelado global del esquema

Constantes del modelo para todas las funciones del esquema. Aplicación del EM a PCEPA en *Saturno*

	Rutinas paralelas de un nivel			Rutinas paralelas de dos niveles		
	Ini	Com	Inc	Mej-Ini	Mej	Div
$k_S \cdot 10^4$	4.56	5.72	5.60	6.05	6.01	56.8
$k_{p,1} \cdot 10^4$	0.482	1.76	12.2	3.16	2.31	12.2
$k_{p,2} \cdot 10^4$	-	-	-	-0.05	37.1	-38.9

Constantes del modelo para todas las funciones del esquema. Aplicación del HEM a PCEPA en *Saturno*

	Rutinas paralelas de un nivel			Rutinas paralelas de dos niveles		
	Ini	Com	Inc	Mej-Ini	Mej	Div
$k_S \cdot 10^2$	1.47	2.95	0.302	26.2	26.2	26.6
$k_{p,1} \cdot 10^2$	0.096	0.593	0.465	3.27	2.43	2.07
$k_{p,2} \cdot 10^2$	-	-	-	2.74	1.68	3.56

Resultados experimentales. Modelado global del esquema

Valores de los parámetros metaheurísticos usados para los experimentos de autooptimización al aplicar el EM (metaheurísticas m1 y m2) y el HEM (hiperheurísticas h1 y h2) a PCEPA en *Saturno*

	NEIIni	NEFIni	PEMIni	IMEIni	MCPIni	NEMSel
m1	75	50	75	20	-	10
m2	150	100	25	15	-	25
h1	10	10	100	1	7	5
h2	20	20	50	3	7	10

	NEPSel	NMMCom	NMPCom	NPPCom	PEMMej	IMEMej
m1	10	50	45	50	100	20
m2	25	100	90	100	40	10
h1	5	20	5	10	100	1
h2	10	50	10	5	20	3

	MCMMej	PEDMej	IDEMej	MCDMej	NEMInc	MLPInc
m1	-	15	15	-	10	-
m2	-	15	10	-	50	-
h1	7	100	1	7	10	7
h2	7	20	3	7	10	7

Justificación parámetros

Valores de los parámetros que producen metaheurísticas e hiperheurísticas de tamaño intermedio, permitiendo un estudio exhaustivo en un tiempo relativamente reducido.

Resultados experimentales. Autooptimización global del esquema

Parámetros de paralelismo seleccionados para cada función por el modelo con autooptimización para PCEPA en el sistema *Saturno*.

	Rutinas paralelas de un nivel			Rutinas paralelas de dos niveles		
	HGEIni	HCPCom	HIEInc	HM1Ini	HM1Mej	HD1Mej
m1	27	22	12	46	51	23
m2	38	31	17	33	32	26

nivel		Rutinas paralelas de un nivel			Rutinas paralelas de dos niveles		
		HGEIni	HCPCom	HIEInc	HM_Ini	HM_Mej	HD_Mej
h1	p_1	12	13	7	5	4	6
	p_2	-	-	-	4	6	4
h2	p_1	17	18	10	6	4	6
	p_2	-	-	-	7	6	4

Análisis resultados parámetros paralelismo

- **EM:** $p_2 = 1$. p_2 trabajan en la mejora del fitness pero no para reducir el tiempo de ejecución.
- En la práctica, si $p_1(\times p_2)_{modelo} > NumCores_{Saturno} \Rightarrow p_1(\times p_2)_{modelo} = NumCores_{Saturno}$.

Comparativa speed-up global autooptimización (exp-auto) - selección no óptima (\max y $\frac{\max}{2}$) en *Saturno*.

	max	speed-up	
		$\frac{\max}{2}$	exp-auto
m1	10	8	10
m2	14	10	16
h1	6	5	6
h2	6	6	7

Análisis resultados speed-up

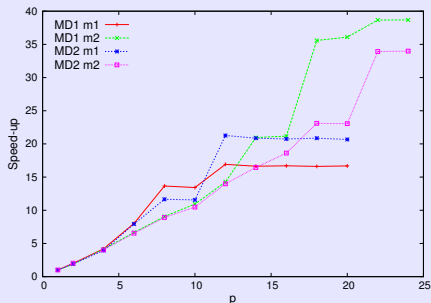
- **EM:** exp-auto mejora o iguala \max y $\frac{\max}{2}$.
- **HEM:** speed-up más parecidos porque:
 - Reducido tamaño de las hiperheurísticas.
 - Sistema *Saturno* con un número de cores del orden de los tamaños poblacionales considerados.

Aplicación a sistemas homogéneos. Comparativa MD - MC

Comparativa de fitness y speed-up con la aplicación del esquema de paso de mensajes y en memoria compartida a PCEPA 50-6 en *Saturno*

	Procesos MPI		Speed-up			MC	Fitness MD1	MD2
	MD1	MD2	MC	MD1	MD2			
m1	12	12	7(-)	17(-)	21(+)	7704.39(+)	7735.39(~)	7842.02(-)
m2	22	22	9(-)	39(+)	34(-)	8139.18(-)	8050.83(-)	7903.52(+)

Speed-up vs p para paso de mensajes

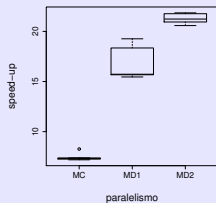


Análisis resultados

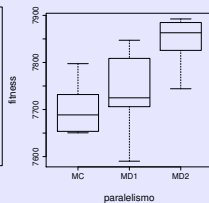
- Diferencias mínimas en algunas funciones básicas entre MD1 y MD2.
- MD mejora claramente a MC en speed-up (paralelismo de más alto nivel).
- Valores de fitness más próximos en todos los casos entre MC y MD.
- Test de Wilcoxon: MD2 presenta fitness **significativamente mejor al aplicar m2**.
- Speed-up superlineal para m2: mejor uso de la memoria y naturaleza cuadrática de algunas funciones.

Aplicación a sistemas homogéneos. Comparativa MD - MC

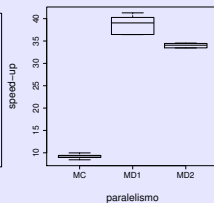
Resumen estadístico del speed-up y del fitness con la aplicación del esquema de paso de mensajes y en memoria compartida PCEPA 50-6 en *Saturno*



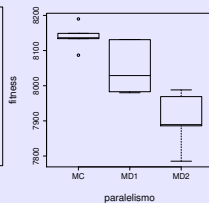
m1 - Speed-up



m1 - Fitness



m2 - Speed-up

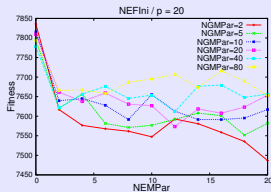


m2 - Fitness

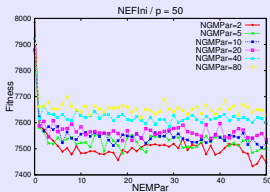
Diferencias significativas en las medias de speed-up y fitness para m1 y m2.

Aplicación a sistemas homogéneos. Influencia de *NEMPar* y *NGMPar*

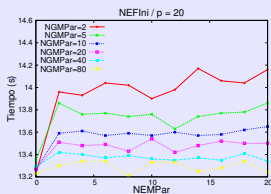
Fitness y tiempo de ejecución vs *NEMPar* (para varios *NGMPar*) en *Saturno*



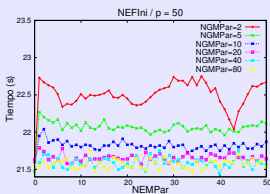
m8



m9



m8



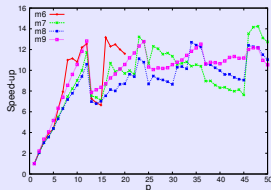
m9

Discusión resultados

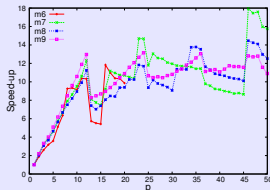
- Fitness ↓ hasta el 20 - 25 % de tasa de migración donde queda plano.
- *NGMPar* ↓ ⇒ ↓ Fitness y ↑ tiempo.
- Poca variación del tiempo vs *NEMPar*: coste envío elementos << coste inicio comunicaciones y coste computacional.
- Óptimos: *NEMPar* 20 - 25 % de migración; *NGMPar* en torno a 5 - 10.

Aplicación a sistemas homogéneos. Influencia de p

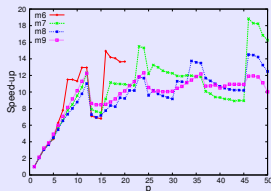
Speed-up vs p en el clúster *Marte + Mercurio*



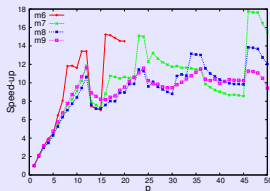
PCEPA 20-6



PCEPA 50-3



PCEPA 50-6



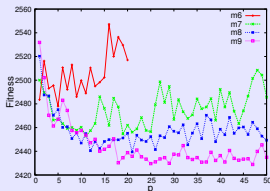
PCEPA 150-3

Discusión resultados

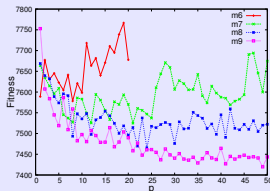
- *Marte + Mercurio* clúster con mayor coste comunicaciones que *Saturno*.
- Incremento lineal del speed-up hasta $p = 12$ (cores del clúster).
- Para $p > 12$, speed-up fluctúa debido balanceo progresivo de la carga.

Aplicación a sistemas homogéneos. Influencia de p

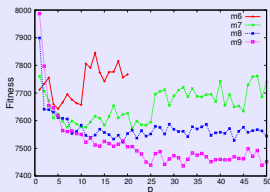
Fitness vs p en el clúster *Marte + Mercurio*



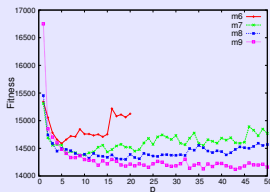
PCEPA 20-6



PCEPA 50-3



PCEPA 50-6



PCEPA 150-3

Discusión resultados

- $\exists \min(\text{fitness})$ en $p \approx \frac{NEFlni}{2}$ (se aprecia en series m6 y m7).
- Si $p > \frac{NEFlni}{2} \Rightarrow \text{fitness} \uparrow$ por pérdida de diversidad en subpoblaciones (islas con solo un individuo).
- En general, aconsejable p altos (poca pérdida de calidad de la solución y disminución notable del tiempo).

Aplicación del esquema en sistemas heterogéneos. Mapeo de procesos

Número de procesos lanzados en el sistema heterogéneo *Saturno*(sat) + *Marte*(mar) + *Mercurio*(mer) + *Luna*(lun)

		CSS	SSB	STB	0.1·STB	0.5·STB
m3	sat	24	30	31	3	12
	mar	6	8	7	1	3
	mer	6	7	7	1	3
	lun	4	5	5	1	2
	total	40	50	50	5	25
m4	sat	24	60	62	6	31
	mar	6	15	14	1	7
	mer	6	15	14	1	7
	lun	4	10	10	1	5
	total	40	100	100	10	50
m5	sat	24	120	122	12	61
	mar	6	30	28	3	14
	mer	6	30	28	3	14
	lun	4	20	22	2	11
	total	40	200	200	20	100

Mapeo de procesos

- *Cores Sin Sobrecarga (CSS)*: $p_{nodo,i} = NCores_{nodo,i}$.
- *Sobrecarga Sin Balancear (SSB)*: $p_{nodo,i} \propto NCores_{nodo,i}$, con $p_{total} = NEFlni$.
- *Sobrecarga Totalmente Balanceada (STB)*: $p_{nodo,i} \propto Velocidad_{nodo,i}$, con $p_{total} = NEFlni$.
- *ySTB*: $p_{nodo,i} \propto Velocidad_{nodo,i}$, con $p_{nodo,i} \times y$ (factor de escalado para reducir la sobrecarga).

Sistemas heterogéneos. Resultados experimentales

Speed-up, fitness e $IC = \frac{10^6}{F \cdot t}$ alcanzado en el clúster *Saturno + Marte + Mercurio + Luna*

	CSS	SSB	STB
m3	34	41	41
m4	44	54	56
m5	38	61	59
Media	39	52	52

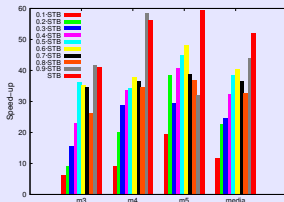
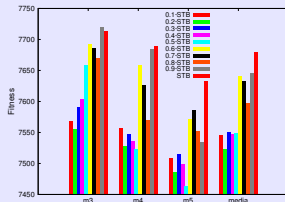
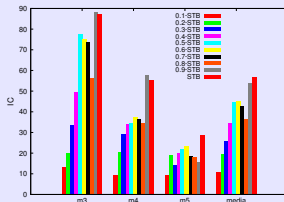
Speed-up

	CSS	SSB	STB
m3	7674.64	7719.67	7713.48
m4	7529.90	7646.13	7690.01
m5	7483.48	7611.97	7633.67
Media	7562.67	7659.26	7679.06

Fitness

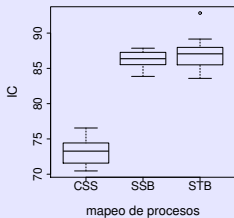
	CSS	SSB	STB
m3	73(-)	86(~)	87(+)
m4	44(-)	53(-)	55(+)
m5	18(-)	29(+)	29(~)
Media	45	56	57

IC

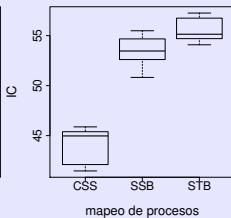
Speed-up - y_{STB} Fitness - y_{STB} IC - y_{STB}

Sistemas heterogéneos. Resultados experimentales

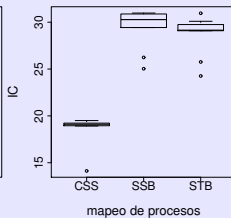
Resumen estadístico para el IC



m3



m4



m5

Análisis resultados

- Mismo speed-up medio de 52 (el más alto) para *SSB* y *STB*.
- $NCores_{totales} \uparrow \Rightarrow \uparrow$ Speed-up.
- El fitness no varía demasiado con el tipo de mapeo, pero sí con *y* en *STB*.
- Diferencias estadísticas significativas en todas las medias del *IC*.
- Test de Wilcoxon para el *IC*: *STB* mejor para m3 y m4; *SSB* mejor para m5.
- Mejor opción: *STB*, con $p_{total} \approx NEFIn_i$.

Autooptimización. Resultados experimentales

Constantes del sistema y parámetros metaheurísticos en *Saturno*

<i>func</i>	k_j	$Param_j$
Com	$3.98 \cdot 10^{-5}$	$2 \cdot (45 + 50 + 45)$
Mej-Ref	$3.36 \cdot 10^{-4}$	$\frac{50 \cdot 75 \cdot 5}{100}$
Mej-Com	$6.72 \cdot 10^{-4}$	$\frac{(45+50+45) \cdot 75 \cdot 5}{100}$
Div-Ref	$3.53 \cdot 10^{-4}$	$\frac{50 \cdot 10 \cdot 5}{100}$
Div-Com	$7.06 \cdot 10^{-4}$	$\frac{(45+50+45) \cdot 10 \cdot 5}{100}$
Inc	$1.50 \cdot 10^{-5}$	$50 + 2 \cdot (45 + 50 + 45) - 25$

Speed-up en *Saturno*

	Procesos MPI		Speed-up		
	exp	mod	exp	mod	exp-auto
m6	16	20	15	11	12
m8	24	24	17	16	17

Discusión resultados

- Parámetros instalación: m7.
- $NGMPar = 1$ (mayor contribución de las comunicaciones, optimiza el fitness).
- p_{mod} y $speed-up_{mod} \approx p_{exp}$ y $speed-up_{exp}$.

Autooptimización. Resultados experimentales

Constantes del sistema y parámetros metaheurísticos en *Marte + Mercurio*

<i>func</i>	k_j	<i>Param_j</i>
Com	$4.54 \cdot 10^{-5}$	$2 \cdot (45 + 50 + 45)$
Mej-Ref	$3.83 \cdot 10^{-4}$	$\frac{50 \cdot 75 \cdot 5}{100}$
Mej-Com	$7.66 \cdot 10^{-4}$	$\frac{(45+50+45) \cdot 75 \cdot 5}{100}$
Div-Ref	$3.92 \cdot 10^{-4}$	$\frac{50 \cdot 10 \cdot 5}{100}$
Div-Com	$7.84 \cdot 10^{-4}$	$\frac{(45+50+45) \cdot 10 \cdot 5}{100}$
Inc	$9.38 \cdot 10^{-6}$	$50 + 2 \cdot (45 + 50 + 45) - 25$

Speed-up en *Marte + Mercurio*

	Procesos MPI		exp	Speed-up	
	exp	mod		mod	exp-auto
m6	11	12	10	7	9
m8	12	12	8	9	8

Discusión resultados

- Constantes del sistema diferentes a las de *Saturno*. Necesidad de autooptimización.
- p_{mod} y $speed-up_{mod} \approx p_{exp}$ y $speed-up_{exp}$.
- Necesidad de más de 12 cores para m6 y m8 (mucho más peso de la computación que de las comunicaciones).

Autooptimización. Resultados experimentales

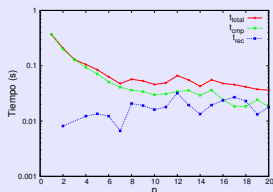
Speed-up *Júpiter + Luna + Saturno + Marte + Mercurio*

	Procesos MPI		Speed-up		
	exp	mod	exp	mod	exp-auto
m6	20	18	10	7	9
m8	38	26	18	13	11
m9	40	37	24	23	20

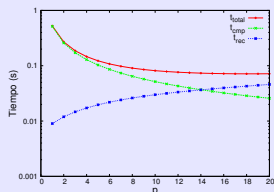
Discusión resultados

- Constantes del sistema: las de *Marte + Mercurio* (nodos más lentos).
- Orden de asignación de procesos a nodos: *Júpiter + Luna + Saturno + Marte + Mercurio* (1^o los más rápidos).
- p_{mod} y $speed-up_{mod} \approx p_{exp}$ y $speed-up_{exp}$.
- Contribución comunicaciones en m8 mayor que en m6.
- Los resultados indican que no se requieren más cores (existencia de mínimos en t_{total}).

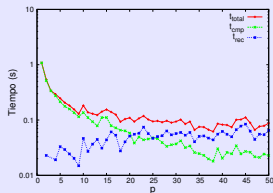
Tiempos experimentales y modelados



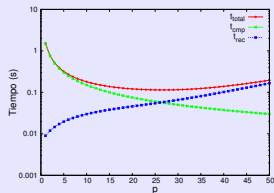
m6 — tiempo experimental



m6 — tiempo modelado



m8 — tiempo experimental

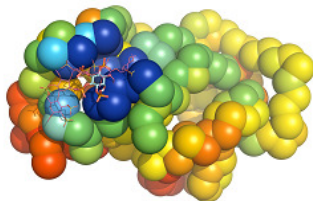


m8 — tiempo modelado

Problema

Estructura paralela

- Varios puntos en el receptor (*spots*) donde se pueden acoplar los ligandos de forma independiente.
- Aplicación de la misma metaheurística en los distintos puntos.



Sistemas computacionales

- Explotación de GPUs en la parte de mayor computación, el cálculo de la función de fitness.
- Cluster heterogéneo, incluyendo rCUDA.

Aplicación de metaheurísticas

- Aplicación de varias metaheurísticas al mismo tiempo.
- Exploración de varias metaheurísticas con un esquema paralelo, con GA, SS y GRASP como metaheurísticas básicas.
- Número de parámetros y significado ligeramente diferente a los vistos anteriormente.
- Generación de poblaciones cerca de cada spot.
- Cada elemento dado por su posición y giro en el espacio, y flexibilidad en algunos puntos de la molécula.
- Función de fitness comparando todas las moléculas del receptor y el ligando (alto coste computacional).
- Combinación generando posiciones intermedias entre dos dadas.
- Vecinos generados en posiciones cercanas y con giros y flexiones pequeños.

Paralelización

Se utiliza OpenMP y se genera un hilo para asignar trabajo a cada GPU.

Computación de la función de Scoring en multicore+multiGPU:

```
omp_set_num_threads(number_GPUs)
#pragma omp parallel for
for i=1 to number_GPUs do
    Select_device(Devices[i].id)
    Host_To_GPU(S,Stmp)
    Conformations=Devices[i].conformations
    threads=Devices[i].Threadsblock
    Calculate_scoring<Conformations/threads,threads>(Stmp)
    GPU_To_Host(S,Stmp)
end for
```

Explotación de la heterogeneidad

- Se puede asignar el mismo número de posibles soluciones a cada GPU.
- Pero las GPUs de un nodo pueden ser heterogéneas.
- Se ejecuta una fase de *Warm Phase* para estimar la capacidad de cada elemento computacional.
- División del trabajo proporcionalmente a la capacidad:

$$Percent = \frac{Ex.time_{actualGPU}}{Ex.time_{slowestGPU}}$$

Explotación de la heterogeneidad

- Se puede asignar el mismo número de posibles soluciones a cada GPU.
- Pero las GPUs de un nodo pueden ser heterogéneas.
- Se ejecuta una fase de *Warm Phase* para estimar la capacidad de cada elemento computacional.
- División del trabajo proporcionalmente a la capacidad:

$$\text{Percent} = \frac{\text{Ex.time}_{\text{actualGPU}}}{\text{Ex.time}_{\text{slowestGPU}}}$$

Benchmarks y Datasets

Benchmarks

Se utilizan cuatro metaheurísticas en los experimentos:

- GA, Algoritmo Genético.
- SS, Scatter Search.
- Ss, Scatter Search con búsqueda local menos intensiva.
- NS, Búsqueda en entorno.

GA, SS y Ss usan conjuntos de 64 elementos en cada spot, y NS de 1024.

Datasets

Número de átomos de los componentes de los benchmarks, de la base de datos PDB.

Componentes	Átomos	Componentes	átomos
2BSM Receptor	3264	2BXG Receptor	8609
2BSM Ligando	45	2BXG Ligando	32

Resultados experimentales

Tiempo de ejecución (en segundos) al aplicar las metaheurísticas a la proteína PDB:2BXG en Jupiter: 12 cores + 4 GeForce GTX 590 + 2 Tesla C2075.

Metaheurística	OpenMP	Sistema Heterogéneo			SPEED-UP Computación heterogénea vs OpenMP
		Computación Homogénea	Computación Heterogénea	porcentaje de reducción	
GA	1402.63	16.96	16.77	1.12	82.70
SS	2272.71	26.57	25.43	4.29	85.53
Ss	711.01	8.72	8.46	2.98	81.53
NS	70505.22	764.131	757.32	0.89	92.26

Tiempo de ejecución (en segundos) al aplicar las metaheurísticas a la proteína PDB:2BXG en Ucam: 4 cores + 1 Tesla K40c + 1 GeForce GTX 580.

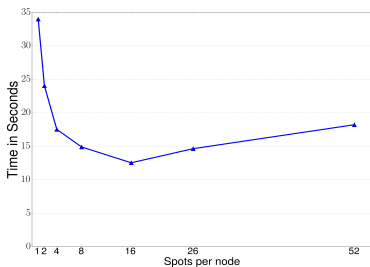
Metaheurística	OpenMP	Sistema Heterogéneo			SPEED-UP Computación heterogénea vs OpenMP
		Computación Homogénea	Computación Heterogénea	porcentaje de reducción	
GA	2327.60	33.92	22.82	32.62	101.96
SS	3908.46	55.56	41.58	25.16	93.98
Ss	1336.40	18.13	13.64	24.67	97.96
NS	150958.75	1735.73	1253.64	27.67	120.41

En cluster

- Se usa MPI para asignar varios *spots* a cada nodo.
- Asignación **Estática**: Se usa *Warm Phase* para evaluar la capacidad computacional.
- Asignación **Dinámica**: Se asigna un número de *spots* por nodo, y cuando un nodo acaba pide otro grupo.

Resultados en 4 nodos con 2 GeForce GTX 480 y 1 Tesla K20c:

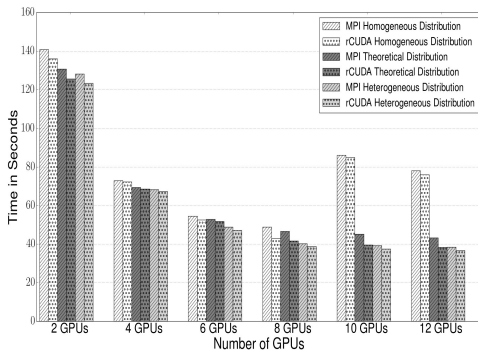
- **Estática**: 15.24 segundos.
- **Dinámica**: Número óptimo de *spots* por grupo 16, con 12.53 segundos.



En cluster con rCUDA

- En un nodo OpenMP, y se asignan varios *spots* a cada GPU (como en MPI).
- Asignación **Homogénea**: mismo volumen de trabajo a cada GPU.
- Asignación **Teórica**: teniendo en cuenta la capacidad computacional de las GPU.
- Asignación **Heterogénea**: con fase de warm-up.

Resultados en 5 nodos: 2 con K40m+K20m, 1 con 4 K20m, 2 con 2 GTX590



Explotación de cuatro niveles de paralelismo en Xeon Phi

ThH_ThT_x representa *threads en la hiperheurística_número total de threads_experimento*.
El resto de threads se asignan a las metaheurísticas.

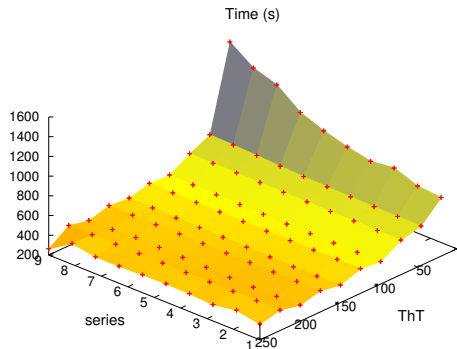
		Rutinas de un nivel				Rutinas de dos niveles			
		TGEIni	TCPCom	TIEInc	TI_Ini	TR_Imp	TC_Imp	TR_Div	TC_Div
ThH_20.1	p_1	1	1	1	1	1	1	1	1
	p_2	-	-	-	2	2	2	1	1
ThH_20.2	p_1	2	2	2	1	1	1	2	2
	p_2	-	-	-	2	2	2	1	1
ThH_20.3	p_1	5	5	5	2	2	2	5	5
	p_2	-	-	-	2	2	2	1	1
ThH_20.4	p_1	5	5	5	1	1	1	1	1
	p_2	-	-	-	5	5	5	2	2
ThH_20.5	p_1	5	5	5	1	1	1	2	2
	p_2	-	-	-	5	5	5	2	2
ThH_20.6	p_1	10	10	10	2	2	2	5	5
	p_2	-	-	-	5	5	5	2	2
ThH_20.7	p_1	10	10	10	1	1	1	1	1
	p_2	-	-	-	10	10	10	5	5
ThH_20.8	p_1	10	10	10	1	1	1	2	2
	p_2	-	-	-	10	10	10	5	5
ThH_20.9	p_1	20	20	20	2	2	2	4	4
	p_2	-	-	-	10	10	10	5	5

Resultados experimentales

Tiempo de ejecución de una hiperheurística tipo GA para varias combinaciones de threads.

Tiempo secuencial en Xeon Phi 7983 segundos.

threads	número total de threads (ThT)					
	20	50	100	150	200	250
ThH_ThT_1	795	428	346	367	366	361
ThH_ThT_2	782	508	388	296	327	422
ThH_ThT_3	940	495	354	283	282	338
ThH_ThT_4	765	519	359	339	324	393
ThH_ThT_5	950	469	343	375	390	397
ThH_ThT_6	1021	530	396	401	345	365
ThH_ThT_7	1449	696	398	331	477	357
ThH_ThT_8	1436	734	407	407	308	442
ThH_ThT_9	1512	910	459	416	318	265



Problema

- Análisis de eficiencia de varias unidades de decisión (DMU)
- Cada unidad k varias entradas x y salidas y
- Problema de optimización para cada unidad
- Disponibles métodos exactos, tipo Branch and Bound, y librerías (CPLEX)
- alto coste de ejecución
- necesidad de paralelismo y métodos aproximados (metaheurísticas)
- y operaciones matriciales en la evaluación de las restricciones y cálculo del fitness
- Posibilidad de métodos híbridos: exactos con metaheurísticas

Martín González, Jose J. López-Espín, Juan Aparicio, Domingo Giménez, Jesús T. Pastor: Using Genetic Algorithms for Maximizing Technical Efficiency in Data Envelopment Analysis, The International Conference on Computational Science, Reykjavík, Iceland, 1-3 June, 2015

Modelo de Programación Matemática

$$\max \quad \beta_k - \frac{1}{m} \sum_{i=1}^m \frac{t_{ik}^-}{x_{ik}}$$

s.t.

$$\beta_k + \frac{1}{s} \sum_{r=1}^s \frac{t_{rk}^+}{y_{rk}} = 1 \quad (c.1)$$

$$-\beta_k x_{ik} + \sum_{j=1}^n \alpha_{jk} x_{ij} + t_{ik}^- = 0 \quad \forall i \quad (c.2)$$

$$-\beta_k y_{rk} + \sum_{j=1}^n \alpha_{jk} y_{rj} - t_{rk}^+ = 0 \quad \forall r \quad (c.3)$$

$$-\sum_{i=1}^m \nu_{ik} x_{ij} + \sum_{r=1}^s \mu_{rk} y_{rj} + d_{jk} = 0 \quad \forall j \quad (c.4)$$

$$\nu_{ik} \geq 1 \quad \forall i \quad (c.5)$$

$$\mu_{rk} \geq 1 \quad \forall r \quad (c.6)$$

$$d_{jk} \leq M b_{jk} \quad \forall j \quad (c.7)$$

$$\alpha_{jk} \leq M(1 - b_{jk}) \quad \forall j \quad (c.8)$$

$$b_{jk} = 0, 1 \quad \forall j \quad (c.9)$$

$$\beta_k \geq 0 \quad (c.10)$$

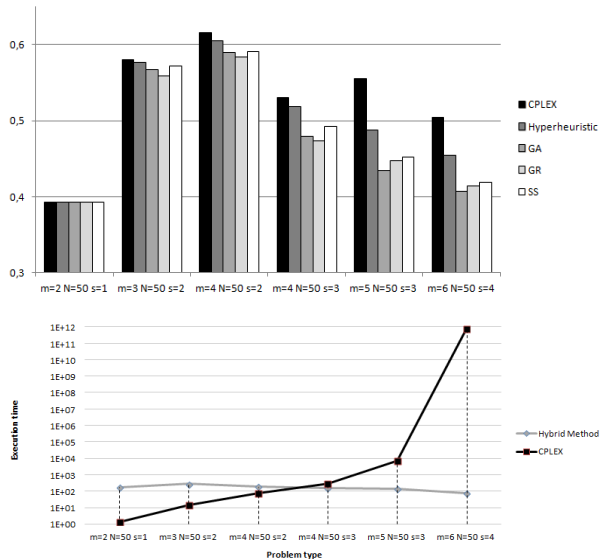
$$t_{ik}^- \geq 0 \quad \forall i \quad (c.11)$$

$$t_{rk}^+ \geq 0 \quad \forall r \quad (c.12)$$

$$d_{jk} \geq 0 \quad \forall j \quad (c.13)$$

$$\alpha_{jk} \geq 0 \quad \forall j \quad (c.14)$$

Resultados



Combinación con métodos numéricos de optimización e hiperheurísticas

Schemes	Hyperheuristic	Metaheuristic
$m = 2, n = 50, s=1$	0.393	0.393
$m = 3, n = 50, s=2$	0.576	0.579
$m = 4, n = 50, s=2$	0.606	0.609
$m = 4, n = 50, s=3$	0.518	0.520
$m = 5, n = 50, s=3$	0.488	0.502
$m = 6, n = 50, s=4$	0.455	0.430

Martin Gonzalez-Rodriguez, Jose-Juan López-Espín, Juan Aparicio, Domingo Giménez, El-Ghazali Talbi: A parameterized scheme of metaheuristics with exact methods for determining the Principle of Least Action in Data Envelopment Analysis. CEC 2017: 588-595

Series Temporales Multivariantes

- Con **Big Data**, cada vez más datos de diversidad de campos.
- En algunos casos, tienen evolución temporal: **series temporales**.
- Método tradicional de obtención de modelos por expertos es cada vez de más difícil aplicación.
- **Series Temporales Multivariantes:**
Parámetros de los que se toman valores periódicamente, determinar la evolución temporal de los parámetros, el valor de un parámetro en un instante puede depender de sus valores en instantes anteriores y de los de otros parámetros. Puede haber parámetros externos, que influyen en los internos pero no influidos por estos.

Datos de la serie temporal

- d parámetros, t instantes de tiempo
Serie temporal es $Y \in R^{t \times d}$:

$$\begin{pmatrix} y_1^{(1)} & \dots & y_d^{(1)} \\ \vdots & \ddots & \vdots \\ y_1^{(t)} & \dots & y_d^{(t)} \end{pmatrix}$$

- Dependencias temporales con i instantes de tiempo anteriores.
- Datos externos al modelo, z , que influyen en los valores de y pero no se ven influenciados:

$$\begin{pmatrix} z_1^{(1)} & \dots & z_e^{(1)} \\ \vdots & \ddots & \vdots \\ z_1^{(t)} & \dots & z_e^{(t)} \end{pmatrix}$$

Formulación matricial

Dependencia de valores de un instante j en función de los valores anteriores:

$$y^{(j)} \approx y^{(j-1)}A_1 + y^{(j-2)}A_2 + \dots + y^{(j-i)}A_i + z^{(j-1)}B_1 + z^{(j-2)}B_2 + \dots + z^{(j-k)}B_k + a$$

A_l , $1 \leq l \leq i$, de dimensión $d \times d$, representa la dependencia de los datos con los valores de l instantes anteriores;

B_l , $1 \leq l \leq k$, de dimensión $e \times d$, dependencias de factores externos.

En forma matricial:

$$Y(i+1:t, :) \approx Y(i:t-1, :)A_1 + Y(i-1:t-2, :)A_2 + \dots + Y(1:t-i, :)A_i + Z(i:t-1, :)B_1 + Z(i-1:t-2, :)B_2 + \dots + Z(i-k+1:t-k, :)B_k + A_0 =$$

$$[Y(i:t-1, :) | \dots | Y(1:t-i, :) | Z(i:t-1, :) | \dots | Z(i-k+1:t-k, :) | 1] \begin{bmatrix} A_1 \\ \vdots \\ A_i \\ B_1 \\ \vdots \\ B_k \\ a \end{bmatrix}$$

Problema de mínimos cuadrados

- Obtener el modelo consiste en calcular A_I , B_I y a que minimizan la diferencia.
- Problema de mínimos cuadrados con matriz de incógnitas de dimensión $(d \cdot i + e \cdot k + 1) \times d$.
- \hat{Y} es $Y(i + 1 : t, :)$, \hat{X} la matriz formada por submatrices de Y , Z y el vector 1 , y \hat{A} la formada por las matrices A_I , B_I y el vector a :

$$\min_{\hat{A}} \left\| \hat{Y} - \hat{X} \hat{A} \right\|$$

Tipos:

- Sin restricciones en las entradas de las matrices.
- Puede que los valores estén en unos ciertos rangos,
- o que sean de un conjunto.
- Puede que los valores de un parámetro no influyan en los de otro.

Problemas matriciales

- Rutinas y librerías para estos tipos de problemas (Matlab, LAPACK).
- Métodos basados en descomposiciones matriciales y métodos iterativos.
- Iterativos: partir de valores proporcionados por los expertos en el campo de los datos, o resolver ecuaciones parciales y comenzar la búsqueda desde ellas.
- Explotación de la estructura tipo Toeplitz por vectores.
- Optimización del cálculo del fitness (métodos numéricos):
 $X * (A + D) = X * A + X * D$, $X * A$ precalculado y $X * D$ coste lineal.

Problemática estadística

- Normalmente se pretende minimizar un estimador estadístico.
- Puede incluir el determinante de la matriz de covarianzas (Akaike).
- La solución del problema de mínimos cuadrados se puede usar como punto de partida para optimizar el estimador con búsqueda local,
- o se puede optimizar directamente con el estimador usando metaheurísticas.

Diversidad de problemas

- Identificación de variables externas.
- Determinación de la amplitud de las dependencias temporales.
- Intervalo de realización de medias para evitar influencia de fluctuaciones.
- Variación del modelo con el tiempo: series temporales de modelos de series temporales.

Campos de aplicación

- Datos meteorológicos.
- Datos médicos, proyecto de Jose Juan López Espín.
- Datos económicos

Por ahora, con datos económicos aproximaciones satisfactorias, con datos médicos peores predicciones.

Alfonso L. Castaño, Javier Cuenca, Jose Matias Cutillas Lozano, Domingo Giménez, Jose-Juan López-Espín, Alberto Pérez-Bernabeu: Parallelism on Hybrid Metaheuristics for Vector Autoregression Models. HPCS 2018: 828-835

Problema de determinación de componentes

- Problema del Grupo de Polímeros del Departamento de Química-Física de la Universidad de Murcia.
<http://leonardo.inf.um.es/macromol/>
- Análisis de sedimentación de una solución compuesta, a partir de planteamiento de José García de la Torre. Analysis of Sedimentation experiments, 2014.
- Determinar la concentración de cada componente en el compuesto. Cada componente dado por su masa, M_k , y coeficiente de fricción, f_k .

Experimento

- Se centrifuga el compuesto. Medidas en distintos instantes de tiempo, t_j , $1 \leq j \leq n_t$, de una señal óptica en posiciones con distancias r_i , $1 \leq i \leq n_r$.
- La señal es función de la posición y del tiempo: $z(r, t)$.
- En instante inicial, con la solución en reposo, la señal tiene el mismo valor z_0 en todas las posiciones r , $z(r, 0) = z_0$.
- Solución con n_k componentes, cuya contribución a la señal es z_k , con $1 \leq k \leq n_k$.
- La señal z es aditiva en los componentes: $z = \sum_{k=1}^{n_k} z_k$.
- Cada z_k es de la posición r y el tiempo t .
- Para cada componente tenemos el peso molecular, M_k , y el coeficiente de fricción, f_k
- y queremos obtener la concentración del componente en la solución, y_k .

Determinación de la señal

- Se realizan varios experimentos, $1 \leq l \leq n_{exp}$
- que dependen de parámetros experimentales, e_l , y entre los que está la velocidad de rotación, ω_l .
- La señal de un experimento para una posición y un tiempo determinado es $z_{exp,l}(r_i, t_j; e_l)$, o simplificando $z_{exp,l}(r_i, t_j)$,
- y se obtiene de forma aditiva a partir de las señales de los componentes:

$$z_{exp,l}(r_i, t_j; e_l) = \sum_{k=1}^{n_k} z_k(r_i, t_j; M_k, f_k, y_k; e_l)$$

- y como la contribución a la señal de cada componente es lineal:

$$z_{exp,l}(r_i, t_j; e_l) = \sum_{k=1}^{n_k} (y_k \cdot z_k(r_i, t_j; M_k, f_k; e_l))$$

Simulación

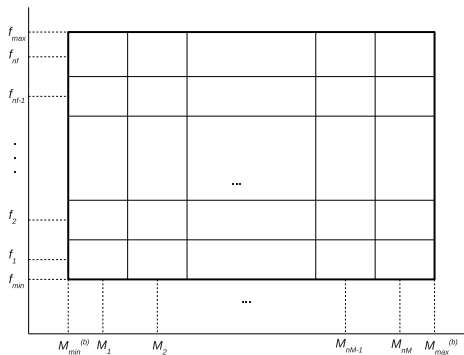
- Se dispone de procedimientos teóricos con los que estimar los valores de señal para cada experimento, en cada momento de tiempo y en cada posición, $z_{cal,l}(r_i, t_j; e_l)$.
- Obtener los valores de y_k con los que la diferencia entre los $z_{exp,l}$ y $z_{cal,l}$ se minimicen. Problema de mínimos cuadrados:

$$\Delta^2(p) = \frac{1}{n_{exp}} \frac{1}{n_r} \frac{1}{n_t} \sum_{l=1}^{n_{exp}} \sum_{i=1}^{n_r} \sum_{j=1}^{n_t} \left(z_{exp,l}(r_i, t_j; e_l) - \sum_{k=1}^{n_k} y_k \cdot z_{k,cal,l}(r_i, t_j; e_l) \right)^2$$

- Sin conocer los z_{cal} ni los z_{exp} , podemos estudiar métodos de determinación de los y_k suponiendo valores de $z_{cal,l}$ determinados y valores de y_k , se generan con ellos los $z_{exp,l}$ que se obtendrían, se perturban, y se aplica el método de resolución del problema de mínimos cuadrados para determinar cuanto de lejos está la solución obtenida con la utilizada en la generación de los datos experimentales.

Malla de valores

- Se considera una distribución uniforme de los parámetros peso molecular y coeficiente de fricción.
- Los M en rango $[M_{min}, M_{max}]$, y los de f en $[f_{min}, f_{max}]$.
- Se discretizan con n_M y n_f valores:

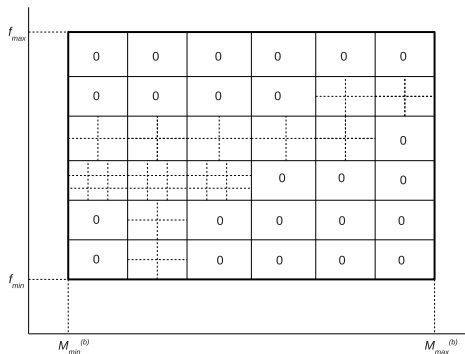


Parámetros del problema

- Típicamente datos de entre 1 y 5 experimentos,
- alrededor de 200 posiciones
- y 200 instantes de tiempo.
- Número de ecuaciones entre $4 \cdot 10^4$ y $2 \cdot 10^5$.
- Si se toman 20 valores de peso molecular y otros 20 para el coeficiente de fricción, el número de incógnitas es 200.

Refinamiento de la malla

- Se puede resolver con esos valores y descartar combinaciones con aportaciones al compuesto muy bajas (valores de y_k por debajo de un umbral)
- y resolver nuevos problemas definiendo mallas más finas en zonas del mallado donde la aportación sea mayor:



Solución numérica

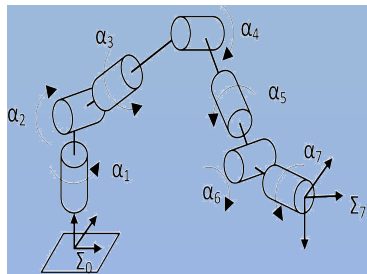
- Se forma la matriz A obtenida con la aplicación de la ecuación para los $n = n_{exp} \cdot n_r \cdot n_t$ experimentos, posiciones en la celda que rota e instantes de tiempo.
- El número de incógnitas es $m = n_M \cdot n_f$.
- Tenemos $A \in R^{n \times m}$,
- y se plantea el problema a optimizar $\min_y \|Ay - b\|$, con b es el vector de señales obtenidas experimentalmente.
- El problema tenemos la restricción de que $0 \leq y_k \leq 1, \forall k / 1 \leq k \leq n_k$, y $\sum_{k=1}^{n_k} y_k = 1$.
- Es un problema de mínimos cuadrados no negativos, o NNLS (Non-Negative Least Squares), con variables acotadas y con una restricción lineal.
- En Matlab está la rutina `lsqlin`.
- Necesaria realización de rutinas paralelas y/o utilización de librerías paralelas

Aproximación metaheurística

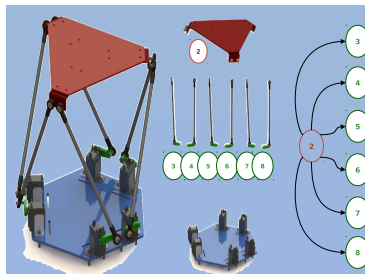
- Se buscan valores de y_k para el problema de mínimos cuadrados.
- Se generan individuos (valores de los y_k con $\sum_{i=1}^k y_k = 1$)
- la función de fitness es la diferencia entre los z_{exp} y los z_{cal}
- y se utilizan técnicas de combinación y mejora de elementos para ir mejorando el conjunto de individuos:
Algoritmos Genéticos, Scatter Search, Ascensión de colinas, Colonia de hormigas...
- Necesaria implementación de metaheurísticas paralelas y/o utilización de librerías paralelas de optimización

Estructura con grupos de ecuaciones

Sistema multicuerpo



Plataforma de Stewart



José-Carlos Cano, Javier Cuenca, Domingo Giménez, Mariano Saura-Sánchez, Pablo Segado-Cabezas: A parallel simulator for multibody systems based on group equations. *The Journal of Supercomputing* 75(3): 1368-1381 (2019)

Simulador

High Performance Kinetic Simulator: Model Builder

Open model | New model | Delete model | Check | Save | Execute | DB | About | Model 1: Stewart

Groups (8) | Scenarios (2) | Routines (5) | Scripts (2) | Training scenario | Config

Select a Group: G4 | New Group | New Final Group | Generate trees

Group name: G4 | Delete Group | Clone group | Add a child

Routine: tres

Successors

Group Name	Delete	↑	↓
G8			

G1 | Add

F(x, ...)	Parameter	Matrix Name	Index
SOLVESYS	inMatrix	M1	3
SOLVESYS	inTrtlindep	M2	3

Delete unused | Delete all

Matrix Name
M1
M2
M3
M4
M5
M6

Auto layout | Rotate | Zoom to Fit | Zoom in | Zoom out | Show scenarios

```

graph TD
    G1[G1] --> G2[G2 SOLVESYS]
    G1 --> G3[G3 SOLVESYS]
    G1 --> G4[G4 SOLVESYS]
    G1 --> G5[G5 SOLVESYS]
    G1 --> G6[G6 SOLVESYS]
    G1 --> G7[G7 SOLVESYS]
    G2 --> G8[G8 MATMUL, MATADD, MATSUB, MATCHG, MATTRN, NORMA, RESHAP]
    G3 --> G8
    G4 --> G8
    G5 --> G8
    G6 --> G8
    G7 --> G8
  
```

- Utilización del simulador para optimización de rutinas de álgebra lineal en entornos heterogéneos.
- Planteamiento de un **problema de optimización**: determinar posición inicial y parámetros de impulso para obtener una trayectoria deseada. El cálculo del fitness consiste en simular la trayectoria (métodos numéricos) y comparar con la deseada.

Contenidos

- 1 Metaheurísticas e Hiperheurísticas
- 2 Metaheurísticas paralelas
- 3 Ejemplos de aplicaciones
- 4 Créditos

En orden de aparición

José-Ceferino Ortega, Domingo Giménez, Alejandro Álvarez-Melcón, Fernando D. Quesada: Hybrid metaheuristics for the design of coupled resonator filters, Applied Artificial Intelligence, 27(5), 323-350, 2013

Francisco Almeida, Domingo Giménez, Jose J. López-Espín, Melquíades Pérez-Pérez: Parameterised schemes of metaheuristics: basic ideas and applications with Genetic algorithms, Scatter Search and GRASP, IEEE SMC, 43(3), 570-586, 2013

Francisco Almeida, José-Matías Cutillas-Lozano, Domingo Giménez: Hyperheuristics based on parameterized metaheuristic schemes, GECCO, 2015

Francisco Almeida, Domingo Giménez, Jose J. López-Espín: A Parametrized Shared-Memory Scheme for Parametrized Metaheuristics, The Journal of Supercomputing, 58(3), 292-301, 2011

José-Matías Cutillas-Lozano, Domingo Giménez: Optimizing a parameterized message-passing metaheuristic scheme on a heterogeneous cluster, Soft Computing, 2016.

Baldomero Imbernón, José M. Cecilia, Horacio Pérez Sánchez and Domingo Giménez: META-DOCK: A Parallel Metaheuristic schema for Virtual Screening methods. Journal of High Performance Computing Applications, 32 (6) 2018

Baldomero Imbernón, Antonio Llanes, José-Matías Cutillas-Lozano and Domingo Giménez: HYPERDOCK: improving Virtual Screening through parallel hyperheuristics. Admitted in Journal of High Performance Computing Applications

José-Matías Cutillas-Lozano, Domingo Giménez, [Luis-Pedro García](#): Optimizing Metaheuristics and Hyperheuristics through Multi-level Parallelism on a Many-core System, PCO, 2016

José Matías Cutillas Lozano: Modelado y autooptimización de metaheurísticas e hiperheurísticas parametrizadas paralelas aplicadas a problemas de optimización en ciencia e ingeniería. Tesis Doctoral, UM, 2014. [Aproximadamente una cuarta parte de la presentación de aquí](#)

[Martín González](#), Jose J. López-Espín, [Juan Aparicio](#), Domingo Giménez, [Jesús T. Pastor](#): Using Genetic Algorithms for Maximizing Technical Efficiency in Data Envelopment Analysis, The International Conference on Computational Science, Reykjavík, Iceland, 1-3 June, 2015

Martin Gonzalez-Rodriguez, Jose-Juan López-Espín, Juan Aparicio, Domingo Giménez, [El-Ghazali Talbi](#): A parameterized scheme of metaheuristics with exact methods for determining the Principle of Least Action in Data Envelopment Analysis. CEC 2017: 588-595

[Alfonso L. Castaño](#), [Javier Cuenca](#), Jose Matias Cutillas Lozano, Domingo Giménez, Jose-Juan López-Espín, [Alberto Pérez-Bernabeu](#): Parallelism on Hybrid Metaheuristics for Vector Autoregression Models. HPCS 2018: 828-835

[José García de la Torre](#). Analysis of Sedimentation experiments, 2014

[José-Carlos Cano](#), Javier Cuenca, Domingo Giménez, [Mariano Saura-Sánchez](#), [Pablo Segado-Cabezos](#): A parallel simulator for multibody systems based on group equations. The Journal of Supercomputing 75(3): 1368-1381 (2019)