Revista Cubana de Ciencias Informáticas
Vol. 9, No. Especial CCIA 17, 2015
RNPS: 2301 | ISSN: 2227-1899
http://rcci.uci.cu
Pág. 126-140

Tipo de artículo: Artículo original
Temática: Inteligencia Artificial
Recibido: 05/01/2015 | Aceptado: 20/02/2015

Solución al problema de conformación de equipos de proyectos de software utilizando la biblioteca de clases BICIAM

Solution to the problem of forming teams software projects using BICIAM library

Ana Lilian Infante Abreu [1*], Rosalí Díaz Hernández [2], Margarita André Ampuero [3], Alejandro Rosete Suárez [4], Jenny Fajardo Calderín [5], Katerine Escalera Fariñas [6]

1* Instituto Superior Politécnico José Antonio Echeverría (ISPJAE). 114 #11901 e/Ciclovía y Rotonda, Marianao, La Habana, Cuba. {ainfante, rdiazh, mayi, rosete, jfajardo, kescalera}@ceis.cujae.edu.cu.

[*]**Autor para correspondencia**: ainfante@ceis.cujae.edu.cu

Resumen

El trabajo aborda la solución del problema de conformación de equipos haciendo uso de la biblioteca de clases de algoritmos metaheurísticos BICIAM. Se describe el modelo utilizado para representar el problema. Haciendo uso de la biblioteca de clases BICIAM se implementa el problema de conformación de equipos. Se realizan pruebas con datos que simulan organizaciones medianas, grandes y especialmente grandes, usando los algoritmos metaheurísticos multiobjetivo de trayectoria y poblacionales implementados en BICIAM. Los resultados obtenidos comprueban el buen desempeño de los algoritmos de trayectoria multiobjetivo sobre los algoritmos poblacionales, destacándose las variantes multiobjetivo del Escalador de Colinas y la Búsqueda Tabú Multiobjetivo.

Palabras clave: biblioteca de clases de algoritmos metaheurísticos, BICIAM, metaheurísticas multiobjetivo, problema de conformación de equipos.

Abstract

The paper deals with the solution of the problem of forming teams using BICIAM class library. The model used to represent the problem is described. Using BICIAM class library the problem of forming teams is deployed. Tests with simulated data of medium, large and especially large organizations, implemented using multiobjective trajectory metaheuristics and population metaheuristics algorithms are performed in BICIAM. The experimental study conducted has led to the identification of various algorithms which produced the best results for this problem. The algorithms identified are the multiobjective variants of Hill Climbing and Multiobjective Tabu Search.

Keywords: *BICIAM, class library metaheuristics algorithms, multiobjective metaheuristic; trouble forming teams.*

Introducción

A pesar del innegable avance que ha tenido la industria de software a nivel internacional, aún resulta significativo el número de proyectos que no culminan con éxito, y así lo evidencia el reporte Chaos[1] (StandishGroup, 2013) en su edición del 2013, donde se plantea que el 39% de los proyectos no culminan con éxito. Esta es una situación de la que no está exenta la industria de software cubana.

Se plantea que entre las principales causas del fracaso de los proyectos de software, se encuentran las asociadas a factores humanos (Charette, 2005; J. Verner et al., 2008; Nelson, 2007; StandishGroup, 2013; Verner and Abdullah, 2011), destacándose los problemas de comunicación en el equipo (Charette, 2005; Gulla, 2011; McManus and Wood-Harper, 2007; Nelson, 2007), la falta de competencias para desempeñar el trabajo en el proyecto (Charette, 2005; Gulla, 2011; Kappelman et al., 2006; McManus and Wood-Harper, 2007; StandishGroup, 2013) y la falta de liderazgo (Kappelman, McKeeman, et al., 2006; McManus and Wood-Harper, 2007; StandishGroup, 2013), lo que evidencia que no se realiza una correcta conformación de los equipos de proyecto. Es por esto que la conformación de equipos adquiere importancia en el marco de la industria de software y numerosos autores reconocen que es un factor crítico para el éxito de los proyectos (Acuña et al., 2008; Acuña and Juristo, 2004; B. W. Boehm, 2000; Beck, 1999; Constantine, 2001; DeMarco and Lister., 1999; Gorla and Wah, 2004; IEEE, 2004; Pressman, 2005; Pyster and Thayer, 2005).

Por otra parte, la conformación de equipos resulta un proceso complejo, en tanto debe tomar en cuenta un conjunto de factores como son: las competencias de los trabajadores, la carga de trabajo asignada, las relaciones interpersonales entre los miembros, entre otros. Muchos trabajos abordan el proceso de conformación de equipos pero solo la propuesta de (André et al., 2011) tiene en cuenta factores que permiten la asignación individual de la persona al rol y factores que contribuyen a la conformación del equipo como un todo.

El modelo de (André, Baldoquín et al., 2011) propone cuatro funciones objetivo y doce tipos de restricciones. El modelo plantea: maximizar las competencias de los trabajadores, minimizar las incompatibilidades entre los miembros, balancear la carga de trabajo y minimizar el costo de desarrollar software a distancia.

La aplicación del modelo se torna compleja en medianas y grandes empresas, teniendo en cuenta la cantidad de combinaciones de asignaciones posibles, en dimensiones relativamente significativas de roles a cubrir y empleado disponibles. El modelo responde a un problema de optimización combinatorio multiobjetivo, por lo que no es posible utilizar algoritmos exactos para su solución. En este tipo de problemas se utilizan algoritmos aproximados, entre los que se encuentran las metaheurísticas. Los algoritmos metaheurísticos no garantizan obtener el óptimo pero si buenas soluciones en tiempos razonables.

En este sentido, se han desarrollado numerosas bibliotecas de clases de algoritmos metaheurísticos que son utilizadas para solucionar problemas de optimización. El uso de bibliotecas garantiza reusar los algoritmos ya implementados y ayuda en la definición del problema a partir de proponer buenas prácticas de diseño. En este trabajo se emplea la biblioteca de clases BICIAM, cuya última versión permite modelar problemas multiobjetivo y que propone diversas variantes de algoritmos metaheurísticos.

[1] Reporte elaborado periódicamente por la compañía Standish Group y que constituye una de las investigaciones más citadas al caracterizar la situación de la industria de software.

En el trabajo se presenta la solución del problema de conformación de equipos como un problema de optimización haciendo uso de la biblioteca de clases de algoritmos metaheurísticos BICIAM. En la sección uno se presenta el problema de conformación de equipos y la propuesta realiza por (André, Baldoquín, et al., 2011), que será utilizada en el trabajo. En la sección dos se describen los algoritmos metaheurísticos utilizados para darle solución al problema, así como la biblioteca de clases BICIAM. En la sección tres se describe la representación del problema de conformación de equipos y su solución haciendo uso de la biblioteca BICIAM. En la sección cuatro se presentan los resultados obtenidos al experimentar en diferentes escenarios del problema de conformación de equipos, haciendo uso de métricas para evaluar el desempeño de algoritmos metaheurísticos y de técnicas de estadística no paramétrica para el análisis de los resultados.

Materiales y métodos

Conformación de equipos de proyectos de software

La conformación de equipos resulta un proceso crítico en el éxito de los proyectos de software (Acuña, Gómez and Juristo, 2008; Acuña and Juristo, 2004; B. W. Boehm, 2000; Beck, 1999; Cockburn, 2000; Constantine, 2001; DeMarco and Lister., 1999; Gorla and Wah, 2004; IEEE, 2004; Pressman, 2005; Pyster and Thayer, 2005), por lo que es un tema que ha sido abordado por diferentes autores. A pesar de ser reconocida su importancia son pocos los modelos que lo formalizan. La mayoría de los trabajos consultados se enfocan en la asignación de personas a tareas, y una minoría plantea la asignación de personas a roles. Todos los trabajos consideran factores que contribuyen a la asignación individual de las personas, destacándose entre estos: las competencias genéricas y técnicas de los trabajadores y la disponibilidad del personal. Sin embargo, solo la propuesta de (André, Baldoquín, et al., 2011) considera además factores que contribuyen a la conformación del equipo como un todo, por lo que es seleccionado en este trabajo para representar el problema.

El modelo formal de conformación de equipos de proyectos de software propuesto en (André, Baldoquín, et al., 2011) considera cuatro objetivos:

- Maximizar las competencias de los trabajadores en el rol o los roles asignados.
  $\max \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij}$, siendo x_{ij} = 1 si el empleado i es asignado al rol j y 0 en caso contrario, y c_{ij} la competencia neta del empleado i para desempeñar el rol j.
- Minimizar las incompatibilidades entre los miembros de un equipo de proyecto.
  $\min \sum_{h=1}^{n-1} \sum_{i>h}^{n} s_{hi} u_h u_i$, siendo s_{hi}la incompatibilidad existente entre los trabajadores h e i y u_i = 1 si el empleado i es asignado al menos a un rol y 0 en caso contrario.
- Balancear la carga del personal del equipo.
  $min \sum_{i=1}^{n} \left[\left(L_i + \sum_{j=1}^{m} b_j x_{ij} \right) - ME \right]^2$, donde $L_i = \sum_{j=1}^{m} g_{ij}$y $ME = \frac{\sum_{j=1}^{m} [(\sum_{i=1}^{n} g_{ij}) + b_j]}{n}$, g_{ij} es la carga de trabajo del empleado i en el rol j según los proyectos a los que está asignado y b_j es la carga de trabajo que implica asumir el rol j en un proyecto determinado.
- Minimizar el costo de trabajar a distancia. Este es un objetivo que no está incluido en el modelo por defecto, sino en una versión ampliada ya que solo es aplicable a organizaciones que enfrentan esta variante de desarrollo.
  $\min \sum_{i=1}^{n} \sum_{j=1}^{m} l_{ij} x_{ij}$, donde l_{ij}es el costo del empleado i según la lejanía que tenga del proyecto y el rol j que va a desempeñar.

Y doce tipos de restricciones, seis de las cuales plantean que:

- Los roles deben ser cubiertos en función de la cantidad necesaria de personas a desempeñarlo.
- Una persona no puede desempeñar al mismo tiempo roles que se consideren incompatibles entre sí.
- Restringir el número máximo de roles que puede desempeñar cualquier trabajador en el proyecto a una cantidad fijada por el usuario.
- Para que una persona desempeñe un rol debe cumplir los requisitos mínimos de nivel de competencia para desempeñar dicho rol.
- La carga de trabajo total asignada a un empleado no debe ser mayor que un valor máximo.
- Garantizar que la variable u_i tome valor 1 ó 0.

Otro conjunto de restricciones refleja la relación que existe entre los roles de Belbin[2], los tipos psicológicos de Myers Briggs[2] y los roles a desempeñar en un equipo de proyecto de software (André, 2009; André, Baldoquín, et al., 2011; Rodríguez, 2008):

- Un conjunto de restricciones garantizan que en el equipo de desarrollo se representen las tres categorías de roles propuestas por Belbin (roles de acción, roles mentales y roles sociales).
- En el equipo de trabajo la preferencia de desempeñar roles de acción debe sobrepasar la preferencia por desempeñar los roles mentales.
- En el equipo de trabajo la preferencia de desempeñar roles mentales deben sobrepasar la preferencia por desempeñar los roles sociales.
- La persona que desarrolla el rol de Jefe de Proyecto debe tener como preferido los roles de Belbin: Impulsor o Coordinador.
- En el equipo al menos una persona debe tener como preferido el rol mental Cerebro.
- La persona que desarrolla el rol Jefe de Proyecto debe ser extrovertida y planificada (subtipo E_ _J) según el test de Myers-Briggs.

Metaheurísticas como solución a problemas de optimización

El modelo analizado anteriormente responde a un problema de optimización combinatorio por lo que para su solución se utilizan las metaheurísticas, las que garantizan obtener una buena solución (no necesariamente la óptima) en un tiempo razonable. Estas se puede clasificar en metaheurísticas de trayectoria y poblacionales. En la primera categoría se clasifican a los algoritmos que partiendo de un punto, buscan la vecindad y actualizan la solución actual en función de esta, formando una trayectoria de punto a punto. En la segunda categoría se clasifican los algoritmos que guían la búsqueda basado en una población de puntos.

A pesar de que el teorema No Free Lunch (Wolpert and Macready, 1997), plantea que no existe una metaheurística que sea absolutamente mejor que otra cuando se comparan en todas las funciones posibles, numerosos autores refieren la capacidad de las variantes multiobjetivo de metaheurísticas poblacionales para solucionar problemas de asignación multiobjetivo (Chica et al., 2011; Coello, 2009; Coello et al., 2007; Dasgupta et al., 2009; Jahromi et al.,

[1]Meredith Belbin es el creador del test que identifica la preferencia de las personas por desempeñar algunos de los nueve roles que define están presentes en un equipo. La metodología establece que en un equipo debe haber presencia de las tres categorías de roles (mentales, sociales y de acción) donde deben predominar los roles de acción y no debe existir una alta presencia de roles sociales ni mentales.

[2]Test que mide cuatro dimensiones diferentes de las preferencias humanas: Extroversión (E)-Introversión (I), Intuición (N)-Sentidos (S), Emoción (F)-Pensamiento (T), y Juicio (J)-Percepción (P). A partir de los valores de cada dimensión se identifica el tipo psicológico de la persona entre los 16 tipos posibles.

2012; Konstantinidis et al., 2009; Landa et al., 2003; Liefooghe, 2010; Mejía, 2008; Sahu and Tapadar, 2007). Sin embargo, de igual forma existen variantes multiobjetivo de algoritmos de trayectoria que han sido empleadas exitosamente en la solución de este tipo de problemas (Baykasoglu et al., 2002; Czyzak and Jaszkiewicz, 1998; Díaz and Rosete, 1999; Haidine and Lehnert, 2008; Hamm et al., 2009; Hapke et al., 2000; Ho et al., 2002; Kulturel-Konak et al., 2004; Smith, 2008; Suman and Kumar, 2006; Ulungu and Teghem, 1994).

En este campo existe una tendencia al uso de bibliotecas de clases de algoritmos metaheurísticos en la solución de problemas de optimización. Entre las ventajas que tiene se destaca el reuso de los algoritmos ya implementados y que permiten emplear buenas prácticas de implementación de los problemas, al separar la lógica de las metaheurísticas de la lógica del problema haciendo uso de patrones de diseño. Existen numerosas bibliotecas de clases de algoritmos metaheurísticos, algunas de estas se muestra en la Tabla 1. Un análisis comparativo de estas bibliotecas permitió seleccionar la biblioteca de clases BICIAM para solucionar el problema de conformación de equipos, teniendo en cuenta que es una de las más completas, en cuanto a que permite la solución de problemas multiobjetivo e incorpora variantes de algoritmos multiobjetivo de trayectoria y poblacionales. Además, emplea un conjunto de patrones de diseño que garantizan su fácil utilización.

BICIAM fue desarrollada en la facultad de Ingeniería Informática de la CUJAE y su implementación se basa en un modelo unificado de algoritmos metaheurísticos, que representa la secuencia de pasos en común entre las metaheurísticas. Actualmente incorpora algoritmos para solucionar problemas monobjetivo tales como: Búsqueda Tabú, Escalador de Colinas, Búsqueda Aleatoria, Recocido Simulado, Algoritmo genético, Algoritmo de Estimación de Distribuciones y Estrategias Evolutivas; así como algoritmos para solucionar problemas multiobjetivo: Escalador de Colinas Estocástico Multiobjetivo, Escalador de Colinas Estocástico Multiobjetivo con Reinicio, Escalador de Colinas Estocástico Multiobjetivo por Mayor Distancia, Recocido Simulado de Ulungu, Recocido Simulado Multicaso, Búsqueda Tabú Multiobjetivo y Algoritmo Genético de Ordenación No-Dominada Elitista. Incorpora además la técnica de solución de problemas multiobjetivo factores ponderados (Caballero and Hernández, 2003; Coello, Lamont, et al., 2007).

Este trabajo considera un conjunto de metaheurísticas de trayectoria multiobjetivo y una variante de metaheurísticas poblacional multiobjetivo para darle solución al problema de conformación de equipos, considerando los algoritmos que implementa la biblioteca de clases BICIAM. Estos algoritmos tienen en común que mantienen una lista de soluciones no dominadas en la que se almacena las soluciones no dominadas encontradas durante la ejecución del algoritmo.

Entre las variantes multiobjetivo del algoritmo Escalador de Colinas se encuentra el Escalador de Colinas Estocástico Multiobjetivo (ECEMO) (Díaz, 2001). Este algoritmo mantiene una lista de soluciones no dominadas durante la búsqueda comparando la dominancia entre la solución actual y la solución candidata, y si esta última no es dominada es comparada con la lista de soluciones no dominadas.

Tabla 1. Comparación de bibliotecas de clases de algoritmos metaheurísticos.

Biblioteca/Parámetros	Lenguaje de implementación	Algoritmos que implementa	¿Resuelve problemas multiobjetivo?	Técnica de solución que implementa	Licencia	Extensibilidad	Aplicaciones
EJC (Luke, 2010)	Java	Estrategias evolutivas, Algoritmo genético, Variantes de Programación Genética, Evolución diferencial, NSGA-II, SPEA2	Si	No	Libre	Si	Se reportan 30 ó 40 publicaciones donde se aplica
JGAP(Rostan, 2012)	Java	Algoritmo genético, Programación genética.	Si	No		Si	No se reportan
JMetal(Durillo et al., 2010)	Java	NSGA-II, SPEA2, PAES, PESA-II, OMOPSO, MOEA/D.	Si	No		Si	Ha sido descargado 1600 veces
MALLBA (Alba and Cotta, 2002)	C++	Algoritmo genético, Recocido Simulado, Estrategias Evolutivas, Colonia de Hormigas, Híbrido de Recocido Simulado con Algoritmo Genético, Búsqueda Local Cooperativa, Optimización de enjambre de partículas.	No	No		Si	Problema de asignación de frecuencias, ajuste de pesos en redes neuronales artificiales, y otros problemas paralelos.
MOMHLib++ (MOMHTeam, 2007)	C++	Recocido Simulado de Pareto, Búsqueda local genética multiobjetivo, Recocido Simulado Multiobjetivo de Serafini, Recocido Simulado Multiobjetivo de Ulungu, Algoritmo Memético de Pareto, entre otros.	Si	No		Si	No se reportan
OPT4J (Lukasiewycz et al., 2011)	Java	Algoritmos evolutivos (SPEA2, NSGA), Optimización de enjambre de partículas multiobjetivo, Recocido Simulado monobjetivo con esquema de enfriamiento predefinido, Evolución diferencial multiobjetivo.	Si	No		Si	Problema del viajante de comercio. Incluye los problemas de prueba ZDT, DTLZ y WFG
BICIAM (Díaz, 2014)	Java	Búsqueda Tabú, Escalador de Colinas, Búsqueda Aleatoria, Recocido Simulado, Algoritmo genético, Algoritmo de Estimación de Distribuciones, Estrategias Evolutivas. Variantes multiobjetivo de Escalcador de Colinas, Recocido Simulado, Búsqueda Tabú y Algoritmo genético.	No	Si, factores ponderados y multiobjetivo puro		Si	Integración de modelos de minería de datos, obtención de predicados difusos a partir de datos, problema del viajante, problema de asignación cuadrática, etc.

Otra variante es el Escalador de Colinas Estocástico Multiobjetivo con Reinicio (ECEMOR) (Díaz, 2001), cuyo funcionamiento es similar al anterior con la diferencia de que cuando la solución candidata no es aceptada para sustituir a la solución actual, el algoritmo verifica si ya se han generado todos los vecinos posibles de la solución actual. De ser así se sustituye la solución actual por otra obtenida aleatoriamente.

El algoritmo Escalador de Colinas Estocástico Multiobjetivo por mayor distancia (ECEMOD) (Díaz, 2001) difiere de los anteriores en la forma de seleccionar la solución actual. Cada vez que se elimina o agrega un elemento a la lista de soluciones no dominadas, se calcula para cada elemento, su distancia respecto a los demás sumando sus distancias individuales. Como nueva solución se toma la solución con mayor distancia.

Entre las variantes multiobjetivo del algoritmo Recocido Simulado se encuentra el Recocido Simulado Multiobjetivo Multicaso (RSMOMC) (Haidine and Lehnert, 2008). Este algoritmo propone una regla de aceptación que se basa en la comparación de la solución actual con la nueva solución, tomando en cuenta el caso en que las dos soluciones sean indiferentes (hay mejoras en algunos objetivos y deterioros en otros). En esta situación se derivan tres subcasos. El

algoritmo permite aceptar soluciones durante el proceso de búsqueda que no están tan lejos del conjunto aproximado actual y evitar regiones que son dominadas por una población densa de soluciones de Pareto. Otra variante es el Método de Ulungu y Teghem (RSMOU) (Coello, Lamont et al., 2007) que mantiene una lista de soluciones no dominadas y calcula la probabilidad de aceptación dando pesos a los objetivos.

La variante de Búsqueda Tabú denominada Búsqueda Tabú Multiobjetivo (BTMO) (Baykasoglu, Ozbaku and Dereli, 2002) mantiene una lista de soluciones no dominadas durante el proceso de búsqueda y acepta soluciones peores solo si no son parte de la lista tabú. Por último el Algoritmo Genético de Ordenación No-Dominada Elitista (NSGAII) (Deb, 2001) como variante del algoritmo genético implanta la idea de un método de selección basado en clases de dominancia para todas las soluciones. Construye una población de individuos, asigna un rango y clasifica cada individuo según los niveles de no dominancia, aplica operaciones evolutivas para crear una nueva agrupación de descendencias, y entonces combina a los padres y a sus hijos antes de dividir los grupos en partes frontales. El NSGAII conduce la operación de nicho con la adición de una distancia calculada a cada miembro que mantiene la diversificación y ayuda al algoritmo a explorar la población.

Solución del modelo de conformación de equipos de proyectos de software utilizando BICIAM

Dado que el problema consiste en asignar trabajadores a los roles definidos en un equipo de proyecto dado y que se puede necesitar que se asignen varios trabajadores a un mismo rol, se define como solución, una lista de elementos, donde cada elemento está compuesto por un rol y la lista de trabajadores que juegan ese rol.

El espacio de soluciones está determinado por la instancia del problema que se analiza. Para una instancia de 60 trabajadores y 6 roles, suponiendo que un trabajador solo puede desempeñar un rol y que el rol Jefe de Proyecto se asigna al inicio y permanece fijo durante la asignación, el espacio de soluciones es 6×10^8.

Para la representación del problema en BICIAM se tuvo en cuenta su estructura. El diseño realizado para solucionar el problema se muestra en la Figura 1. Se encuentra resaltado en azul las clases que se implementan para modelar el problema. Los elementos modelados son:

- Se define una clase para cada una de las funciones objetivo del problema (*MinimizeIncompatibilities*, *MinimizeCostDistance*, *MaximizeCompetency* y *MinimizeWorkload*), que heredan de la clase *ObjectiveFunction* de BICIAM. Se implementan tres operadores que heredan de la clase *Operator* de BICIAM.
- Se implementa una clase por cada una de las restricciones del problema. Estas heredan de la clase Constrain que representa una restricción.
- La clase *Solution* representa la condificación del problema, hereda de la clase *Codification* de BICIAM, y es quien manipula las restricciones del mismo.
- La clase *ProblemX* hereda de Problem y es la encarga de manipular el problema.
- La clase *RoleWorkerState* representa una solución y hereda de la clase *State*.

Revista Cubana de Ciencias Informáticas
Vol. 9, No. Especial CCIA 17, 2015
RNPS: 2301 | ISSN: 2227-1899
http://rcci.uci.cu
Pág. 126-140

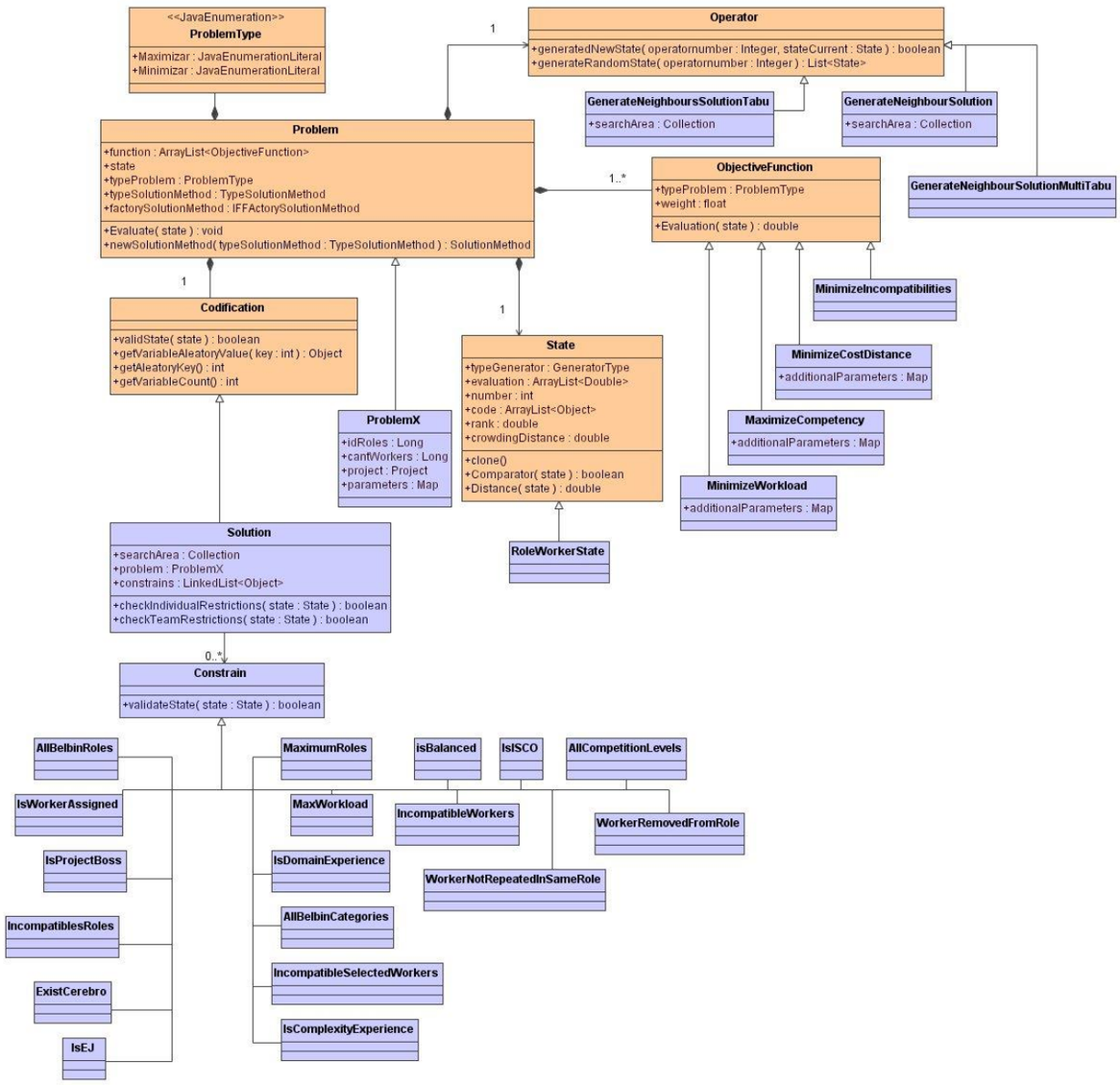

Figura 1. Clases para implementar el problema de conformación de equipos usando BICIAM

Resultados y discusión

Para validar la solución del problema de conformación de equipos usando la biblioteca de clases BICIAM se generaron tres escenarios de prueba, que simulan organizaciones medianas (entre 60 y 250 trabajadores), grandes (más de 250 trabajadores) y especialmente grandes (más de 1000 trabajadores): **Escenario 1** - 100 trabajadores y 6 roles a cubrir, **Escenario 2** - 500 trabajadores y 10 roles a cubrir y **Escenario 3** - 1500 trabajadores y 10 roles a cubrir.

Los parámetros de los algoritmos fueron fijados como se describe a continuación:

- Todos los algoritmos: 45000 iteraciones y 20 ejecuciones por algoritmo
- Búsqueda Tabú Multiobjetivo: 20 elementos en la lista tabú, 91 iteraciones para el escenario 1, 10 iteraciones para el escenario 2 y 3 iteraciones para el escenario 3.
- Recocido Simulado Multiobjetivo de Ulungu: 100 iteraciones con la misma temperatura, alpha de 0.9, temperatura inicial de 20 y temperatura final de 0.
- Escalador de Colinas Estocástico Multiobjetivo por Mayor Distancia/ Escalador de Colinas Estocástico Multiobjetivo con Reinicio: tamaño de la vecindad a explorar de 2.

- NSGAII: 9000 iteraciones, tamaño de la población de 5, selección por torneo como operador de selección, cruzamiento uniforme como operador de cruzamiento, mutación en un punto como operador de mutación, probabilidad de mutación de 0.9 y probabilidad de cruzamiento de 0.5.

Métricas para evaluar el desempeño de algoritmos multiobjetivo

Para comparar el rendimiento de diferentes algoritmos multiobjetivo se utilizan varias métricas, las cuales persiguen medir tres aspectos (Van-Veldhuizen, 1999; Zitzler, 1999): la cantidad de elementos del conjunto de óptimos de Pareto generados, la distancia del frente de Pareto producido por el algoritmo diseñado con respecto al frente verdadero (suponiendo que el frente de Pareto verdadero es conocido) y la distribución de soluciones obtenidas, de manera que se pueda tener una distribución de vectores lo más uniforme posible.

En este trabajo se utilizarán tres métricas que evalúan cada uno de los aspectos mencionados anteriormente: tasa de error, distancia generacional y dispersión.

- La tasa de error (Veldhuizen, 1999) indica el porciento de soluciones que no son miembros del frente de Pareto verdadero. El comportamiento ideal del algoritmo es que la tasa de error sea 0, puesto que indica que todos los vectores generados por el algoritmo pertenecen al frente de Pareto verdadero. Sin embargo, esta métrica requiere conocer los elementos del frente de Pareto verdadero, lo cual puede resultar difícil de obtener en problemas reales.
- La métrica distancia generacional (Veldhuizen, 1999) indica qué tan lejos están los elementos del frente de Pareto actual respecto al frente de Pareto verdadero. Si la distancia generacional es 0 indica que todos los elementos generados están en el frente de Pareto verdadero. Cualquier otro valor indica que tan lejos se está del frente de Pareto verdadero.
- La métrica dispersión(Schott, 1995) mide la varianza de la distancia de cada miembro del conjunto de óptimos de Pareto (encontrados hasta el momento) con respecto a su vecino más cercano. Un valor de 0 de esta métrica indica que todos los miembros del frente de Pareto generado están equidistantes.

Test estadísticos no paramétricos

Las técnicas estadísticas no paramétricas se aplican cuando no se puede determinar la distribución original ni la distribución de los estadísticos, por lo que en realidad no se tienen parámetros a estimar sino distribuciones que comparar (Guillen, 2013; Wasserman, 2006). Para el análisis de los resultados se formula una prueba de hipótesis con el objetivo de determinar si existen o no diferencias entre el comportamiento de los algoritmos en general, donde la hipótesis nula plantea que no existen diferencias entre los algoritmos. Todas las pruebas de hipótesis se realizan con un nivel de significancia de $\alpha = 0{,}05$.

En este trabajo para probar las hipótesis sobre el comportamiento general de los algoritmos se utilizará la prueba de Friedman, la cual calcula el *ranking* de los resultados observados por algoritmo para cada problema, asignando al mejor de ellos el *ranking* 1, y al peor el *ranking* k. Se realiza además, un análisis *post hoc* con Holm y Finner.

Resultados experimentales

El análisis de los resultados obtenidos por los diferentes algoritmos se realizó para cada métrica por escenario y general para todos los escenarios. La Tabla 2 muestran los *rankings* de Friedman para la métrica Tasa de error en cada

uno de los escenarios y general de todos los escenarios, además del valor del p-value que indica si existen o no diferencias significativas entre los resultados de los algoritmos. Como se observa, en todos los escenarios existen diferencias significativas entre los resultados ya que el p-value es menor que 0.05. Aparece resaltado en negrita el algoritmo que obtuvo como promedio mejores resultados. Se destacan por sus buenos resultados los algoritmos BTMO en el escenario 1, los algoritmos ECEMOR y RSMOMC en el escenario 2 y el algoritmo ECEMOR en el escenario 3. El análisis general de todos los escenarios evidencia que ECEMOR es el algoritmo que mejores resultados obtiene.

Tabla 2. Resultados de *rankings* y p-value del test de Friedman por escenario y general. Métrica tasa de error.

Algoritmo	Escenario 1	Escenario 2	Escenario 3	Todos
ECEMO	4.425	2.65	3.4	3.4917
ECEMOR	2.975	**2.15**	**2.775**	**2.6333**
ECEMOD	4.4	3.725	3	3.7083
BTMO	**2.275**	5.775	5.175	4.4083
RSMOU	5.35	5.775	5.175	5.4333
RSMOMC	3.575	**2.15**	3.3	3.0083
NSGAII	5	5.775	5.175	5.3167
p-value	$2x10^{-5}$	0	$1.6x10^{-5}$	$8.08x10^{-11}$

Haciendo un análisis *post hoc* de los resultados obtenidos por Friedman en el escenario 1, se comparan los resultados entre el mejor algoritmo (BTMO) y el resto de los algoritmos. Los resultados obtenidos por ECEMOR y RSMOMC no difieren de los obtenidos por BTMO para este escenario, tal como se muestra en la Tabla 3.

En el análisis *post hoc* de los resultados obtenidos por Friedman en el escenario 2, se comparan los resultados entre ECEMOR (el algoritmo de mejor *ranking*) y el resto de los algoritmos. Los resultados obtenidos por ECEMO, ECEMOD y RSMOMC no son diferentes a los obtenidos por ECEMOR para este escenario. En el análisis *post hoc* de los resultados obtenidos por Friedman en el escenario 3, se observa que no existen diferencias significativas entre los resultados de los algoritmos ECEMO, RSMOMC y ECEMOD y el algoritmo ECEMOR (algoritmo de mejor *ranking*).

Tabla 3. Resultados de los post hoc por escenario de la métrica tasa de error comparando el mejor algoritmo con el resto. (R-rechaza la hipótesis, NR-no rechaza la hipótesis)

Algoritmo	Escenario 1				Escenario 2				Escenario 3			
	p-value Holm		p-value Finner		p-value Holm		p-value Finner		p-value Holm		p-value Finner	
ECEMO	0.006592	R	0.003293	R	0.9284	**NR**	0.527	**NR**	1.080722	**NR**	0.48829	**NR**
ECEMOR	0.305507	**NR**	0.305507	**NR**								
ECEMOD	0.006592	R	0.003293	R	0.0634	**NR**	0.0315	R	1.080722	**NR**	0.74188	**NR**
BTMO					0.000001	R	1E-06	R	0.002656	R	0.00265	R
RSMOU	0.000041	R	0.000041	R	0.000001	R	1E-06	R	0.002656	R	0.00265	R
RSMOMC	0.11408	**NR**	0.068051	**NR**	1	**NR**	1	**NR**	1.080722	**NR**	0.50364	**NR**
NSGAII	0.000332	R	0.000199	R	0.000001	R	1E-06	R	0.002656	R	0.00265	R

El análisis general de la métrica Tasa de error en los tres escenarios permite concluir que los mejores algoritmos son ECEMO, ECEMOR, ECEMOD, BTMO y RSMOMC, ya que no existen diferencias significativas entre estos.

La Tabla 4 muestran los *rankings* de Friedman para la métrica dispersión en cada uno de los escenarios y general de todos los escenarios, además del valor del p-value que indica si existen o no diferencias significativas entre los resultados de los algoritmos. Como se observa en todos los escenarios existen diferencias significativas entre los resultados ya que el p-value es menor que 0.05. Aparece resaltado en negrita el algoritmo que obtuvo como promedio mejores resultados. Se destacan por sus buenos resultados los algoritmos BTMO en los escenarios 1 y 2, y NSGAII en

el escenario 3. El análisis general de todos los escenarios evidencia que BTMO es el algoritmo que mejores resultados obtiene.

El análisis *post hoc* del test de Friedman para la métrica dispersión se muestra en la Tabla 5. En todos los escenarios se evidencia que no existen diferencias entre los resultados de los algoritmos BTMO y NSGAII.

Tabla 4. Resultados de *rankings* y p-value del test de Friedman por escenario y general. Métrica dispersión.

Algoritmo	Escenario 1	Escenario 2	Escenario 3	Todos
ECEMO	4.225	4.55	4.35	4.375
ECEMOR	3.775	3.2	4.4	3.7917
ECEMOD	4.55	5.8	4.5	4.95
BTMO	**1.925**	**1.75**	2.375	**2.0167**
RSMOU	5.4	6.35	6.2	5.9833
RSMOMC	4.9	4.3	4.4	4.5333
NSGAII	3.225	2.05	**1.775**	2.35
p-value	$5x10^{-6}$	0	0	$6.8x10^{-11}$

Tabla 5. Resultados de los *post hoc* por escenario de la métrica dispersión comparando el mejor algoritmo con el resto. (R-rechaza la hipótesis, NR-no rechaza la hipótesis)

Algoritmo	Escenario 1				Escenario 2				Escenario 3			
	p-value Holm		p-value Finner		p-value Holm		p-value Finner		p-value Holm		p-value Finner	
ECEMO	0.002281	R	0.00114	R	0.000166	R	0.000083	R	0.000487	R	0.000243	R
ECEMOR	0.013533	R	0.008114	R	0.067578	**NR**	0.040408	R	0.000487	R	0.000243	R
ECEMOD	0.000487	R	0.000243	R	0	R	0	R	0.000332	R	0.000199	R
BTMO									0.379775	**NR**	0.379775	**NR**
RSMOU	0.000002	R	0.000002	R	0	R	0	R	0	R	0	R
RSMOMC	0.000067	R	0.00004	R	0.000568	R	0.000284	R	0.000487	R	0.000243	R
NSGAII	0.05704	**NR**	0.05704	**NR**	0.660549	**NR**	0.660549	**NR**				

La Tabla 6 muestran los *rankings* de Friedman para la métrica distancia generacional en cada uno de los escenarios y general de todos los escenarios, además del valor del p-value que indica si existen o no diferencias significativas entre los resultados de los algoritmos. Como se observa en todos los escenarios existen diferencias significativas entre los resultados ya que el p-value es menor que 0.05. Aparece resaltado en negrita el algoritmo que obtuvo como promedio mejores resultados. Se destaca por sus buenos resultados en cada uno de los escenarios y general de todos los escenarios el algoritmo ECEMOR.

El análisis *post hoc* del test de Friedman para la métrica distancia generacional se muestra en la Tabla 7. En el escenario 1 no existen diferencias entre los algoritmos ECEMOR, ECEMO, ECEMOD, BTMO y RSMOMC. En el escenario 2 no existen diferencias entre ECEMOR y RSMOMC. En el escenario 3 no existen diferencias entre ECEMOR, ECEMO y RSMOMC.

Tabla 6. Resultados de *rankings* y p-value del test de Friedman por escenario y general. Métrica distancia generacional.

Algoritmo	Escenario 1	Escenario 2	Escenario 3	Todos
ECEMO	3.7	3	2.25	2.9833
ECEMOR	**2.525**	**1.35**	**1.75**	**1.875**
ECEMOD	3.6	3.475	3.3	3.4583
BTMO	3.5	4.65	5.5	4.55
RSMOU	4.95	5.95	5.6	5.5
RSMOMC	3.75	2.575	2.7	3.0083
NSGAII	5.975	7	6.9	6.625
p-value	$1.4x10^{-5}$	0	0	$1.24x10^{-10}$

Tabla 7. Resultados de los *post hoc* por escenario de la métrica distancia generacional comparando el mejor algoritmo con el resto. (R-rechaza la hipótesis, NR-no rechaza la hipótesis)

Algoritmo	Escenario 1				Escenario 2				Escenario 3			
	p-value Holm		p-value Finner		p-value Holm		p-value Finner		p-value Holm		p-value Finner	
ECEMO	0.291753	**NR**	0.140557	**NR**	0.03144	R	0.018834	R	0.464214	**NR**	0.464214	**NR**

ECEMOR												
ECEMOD	0.291753	**NR**	0.140557	**NR**	0.0056	R	0.000003	R	0.069811	**NR**	0.034702	R
BTMO	0.291753	**NR**	0.153507	**NR**	0.000005	R	0.000003	R	0	R	0	R
RSMOU	0.001927	R	0.001156	R	0	R	0	R	0	R	0	R
RSMOMC	0.291753	**NR**	0.140557	**NR**	0.072938	**NR**	0.072938	**NR**	0.328659	**NR**	0.193801	**NR**
NSGAII	0.000003	R	0.000003	R	0	R	0	R	0	R	0	R

El análisis global de los resultados permite concluir que los algoritmos de trayectoria multiobjetivo se comportan mejor que los algoritmos poblacionales, teniendo en cuenta las métricas tasa de error y distancia generacional. Se destacan las variantes multiobjetivo del algoritmo Escalador de Colinas: ECEMO, ECEMOR y ECEMOD y el Recocido Simulado Multiobjetivo Multicaso por sus buenos resultados en las métricas distancia generacional y tasa de error, incluso en escenarios muy grandes (escenario 3). El algoritmo BTMO obtiene buenos resultados en escenarios de organizaciones medianas (escenario 1) para estas métricas. En cuanto a la métrica dispersión obtienen mejores desempeños los algoritmos BTMO y NSGAII.

Conclusiones

El problema de conformación de equipos de proyectos de software es modelado como un problema de optimización multiobjetivo, siguiendo un modelo formal que propone optimizar cuatro funciones objetivo y doce tipos de restricciones. Se utiliza la biblioteca de clases BICIAM para solucionar el problema haciendo uso de algoritmos metaheurísticos multiobjetivo, lo que permite el uso de buenas prácticas en la implementación del problema.

Los resultados experimentales de aplicar los algoritmos a tres escenarios de prueba permiten concluir que los algoritmos de trayectoria multiobjetivo se comportan en general mejor que el algoritmo poblacional multiobjetivo utilizado, destacándose las tres variantes multiobjetivo del Escalador de Colinas utilizadas y el Recocido Simulado Multiobjetivo Multicaso. En el caso del algoritmo Búsqueda Tabú Multiobjetivo obtiene buenos resultados en escenarios medianos fundamentalmente. Es de desatacar cómo variantes tan sencillas de algoritmos, como las variantes del Escalador de Colinas obtienen buenos resultados en este problema, a pesar de ser muy poco utilizadas en la bibliografía en la solución de problemas de optimización.

Referencias

Ashenden, Peter J. The VHDL Cookbook. Departamento de Ciencias de la Computación, Universidad de Adelaide, Australia. Julio, 1990.

Batard Lorenzo, David & Martínez Hernando, Víctor J. “Simulación de NEP en Java”, Evento UCIENCIA 2012, La Habana, Cuba, 2012.

Implementación sobre FPGA de una Red de Procesadores Evolutivos (NEP) para solucionar el Problema de los Tres Colores. I Convención Internacional de Ciencias Técnicas y VII

Conferencia Internacional de Ingeniería Eléctrica, Universidad de Oriente, Santiago de Cuba, Cuba, junio 2014. ISBN: 978-959-207-529-0

Implementación en Hardware Reconfigurable de una Red de Procesadores Evolutivos para la Solución a un Problema NP-Completo. I Conferencia Internacional UCIENCIA, Universidad de las Ciencias Informáticas, La Habana, Cuba, abril 2014. ISBN 978-959-286-026-1

Díaz Martínez, Miguel Ángel. “Redes de Procesadores Evolutivos con Filtros en las Conexiones”. Tesis Doctoral, Universidad Politécnica de Madrid, España, 2008.

Digilent. Información obtenida en fecha 26 de junio de 2014. Información en http://www.digilentinc.com/Products/ATLYS.

Freire Rubio, Miguel Ángel. "Manual de Introducción al lenguaje VHDL". Universidad Politécnica de Madrid, España. Año 2011.

Martínez Hernando, Víctor. "Desarrollo de Sistemas Físicos para Implantar Modelos de Computación con Membranas". Tesis Doctoral, Universidad Politécnica de Madrid, España (2008).

Păun, Gh., Rozenberg, G., & Salomaa, A., "DNA Computing. New Computing Paradigms", Berlin, Springer, 1998.

Ramírez Despaine, Maikel. "Controlador Lógico Programable basado en Hardware Reconfigurable". Tesis de Maestría, CUJAE, La Habana, Cuba, 2011.

Xilinx. Información obtenida en http: //www.Xilinx.com. Información Publicada el 20 de febrero de 2013, consultada el 15 de mayo de 2015.