Revista Cubana de Ciencias Informáticas
Vol. 8, No. 3, Julio-Septiembre, 2014
ISSN: 2227-1899 | RNPS: 2301
Pág. 53-68
http://rcci.uci.cu

Tipo de artículo: Artículo original
Temática: Inteligencia artificial
Recibido: 19/03/2014 | Aceptado: 5/06/2014

Uso de estrategias de paralelización en algoritmos metaheurísticos para la conformación de equipos de software

The use of parallelization strategies in metaheuristic algorithms to solve the problem of software development team conformation

Katerine Escalera Fariñas [1*], Ana Lilian Infante Abreu [1], Margarita André Ampuero [1,] Alejandro Rosete Suárez [1]

1* Facultad de Ingeniería Informática, Instituto Superior Politécnico José Antonio Echeverría. Calle 114 # 11901 e/ Ciclovía y Rotonda Marianao La Habana, Cuba. {kescalera, ainfante, mayi, rosete}@ceis.cujae.edu.cu

---

Resumen

Este trabajo se centra en el uso de estrategias de paralelización en algoritmos metaheurísticos, con el objetivo de disminuir los tiempos de respuesta de los algoritmos y mejorar la calidad de las soluciones obtenidas en la solución del problema de conformación de equipos. En el trabajo se realizan experimentos que permiten comparar el desempeño de diferentes algoritmos metaheurísticos aplicando o no estrategias de paralelización en organizaciones especialmente grandes. Los resultados obtenidos muestran mejoras en los tiempos de respuesta y la calidad de las soluciones con el uso de la estrategia de múltiples trayectorias.

Palabras clave: algoritmos metaheurísticos, conformación de equipos de software, estrategias de paralelización, paralelismo.

Abstract

This paper focuses on the use of parallelization strategies in metaheuristic algorithms. These strategies are used in order to reduce the response times of the algorithms and improve the quality of the solution for the software development team formation problem. Experiments were performed to compare the performance of different

metaheuristics algorithms with and without parallelization strategies in especially large organizations. The results show improvements in response times and quality of solutions using multiple concurrent exploration.
Keywords: *conformation forming teams of software, metaheuristics algorithms, parallelism, parallelization strategies.*

Introducción

La conformación de equipos de software responde a un problema multiobjetivo ya que implica tomar en cuenta diferentes factores, no solo aquellos que contribuyan a la asignación del personal adecuado a cada uno de los roles definidos, sino aspectos vinculados al equipo como un todo, como las relaciones entre los miembros.

Este problema resulta más complejo en organizaciones medianas y grandes, debido a la gran cantidad de combinaciones de asignaciones posibles, por lo que se hace prácticamente imposible abordar de manera eficiente sin la ayuda de modelos matemáticos y el empleo de algoritmos y métodos para su solución.

Este trabajo toma como base un modelo para la formación de equipos de proyectos de software (André, 2009; M. André, M.G. Baldoquín, *et al.* 2011) que propone optimizar cuatro funciones objetivo: maximizar las competencias de los trabajadores, minimizar las incompatibilidades entre los miembros del equipo, balancear la carga de trabajo y minimizar el costo del desarrollo a distancia (para aquellas organizaciones que incluyan este modelo de trabajo), e incluye doce tipos de restricciones.

El modelo, así como los algoritmos y métodos de solución están implementados en una herramienta de soporte a la toma de decisión denominada Teamsoft^{+}. La herramienta es configurable lo que facilita su empleo en cualquier organización de desarrollo de software, y a su vez permite realizar estudios experimentales para evaluar el desempeño de diferentes algoritmos metaheurísticos en la solución del modelo.

En (Infante, 2012) se realizaron pruebas con algoritmos multiobjetivo en la solución del modelo en escenario de organizaciones medianas, grandes y especialmente grandes, obteniéndose soluciones factibles en tiempos de respuesta aceptables (entre 5 y 12 minutos, respectivamente para los peores casos). Sin embargo en estos estudios el por ciento del espacio de soluciones explorado fue inferior al 0,01% en los escenarios evaluados. En pruebas realizadas con los algoritmos implementados en los mismos escenarios, donde se recorrió el 1% del espacio de soluciones, los tiempos de respuesta alcanzan las 12 horas aproximadamente, lo que demuestra que los tiempos se incrementan significativamente al aumentar el espacio de soluciones visitado.

En trabajos anteriores (André, 2009; M. André, M.G. Baldoquín, *et al.* 2011; Infante, 2012) se realizaron pruebas con algoritmos de trayectoria tradicionales y multiobjetivo. Sin embargo, en ninguno de los trabajos se aplicaron estrategias de paralelización a los algoritmos metaheurísticos utilizados, lo cual podría resultar beneficioso para la solución en escenarios que responden a organizaciones grandes y especialmente grande (más de 1000 personas), teniendo en cuenta que una de las principales ventajas de estas estrategias es disminuir el tiempo de respuesta. La posibilidad de disminuir el tiempo de respuesta no solo permitiría evaluar mayor cantidad de propuestas de equipo para comparar y elegir la más adecuada sino aumentar el espacio de búsqueda visitado.

Existen numerosos trabajos donde se definen algoritmos metaheurísticos con estrategias de paralelización (Malek, Guruswamy, *et al.* 1989; Falco, R.D Balio *et al.* 1994; Lee, 1995; Cung, Martins, *et al.* 2001; Crainic and Toulouse 2002; Cruz and Moreno 2007; Luer, Venegas *et al.* 2009). Las estrategias de paralelización tienen dos objetivos fundamentales: mejorar el tiempo de respuesta de los algoritmos o mejorar la calidad de las soluciones respecto a los algoritmos secuenciales.

Este trabajo tiene como objetivo realizar un estudio del comportamiento de los algoritmos metaheurísticos haciendo uso de estrategias de paralelización para dar solución al problema de conformación de equipos de software.

Materiales y métodos

El modelo de conformación de equipos de software responde a un problema de asignación, que plantea asignar n trabajadores a m roles disponibles. El modelo tiene en cuenta los siguientes objetivos (André, Baldoquín *et al.* 2008; Rodríguez, 2008; André, 2009; André, Baldoquín *et al.* 2009):

- Maximizar las competencias de los trabajadores.
- Minimizar las incompatibilidades entre los trabajadores de un equipo de proyecto.
- Balancear la carga del personal del equipo.
- Minimizar el costo de trabajar a distancia.

El último objetivo solo es aplicable a organizaciones que enfrentan esta variante de desarrollo.

El modelo toma en cuenta un conjunto de restricciones que garantizan la asignación individual de personas a roles:

- Los roles deben ser cubiertos en función de la cantidad necesaria de personas a desempeñarlo.
- Una persona no puede desempeñar al mismo tiempo roles incompatibles.
- Restringir el número máximo de roles que puede desempeñar cualquier trabajador.
- 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.

Existen otro conjunto de restricciones asociadas a la relación que existe entre los roles de Belbin[1], los tipos psicológicos de Myers Briggs[2] y los roles a desempeñar en un equipo de proyecto de software (André, Baldoquín, *et al.* 2008; Rodríguez, 2008; André, 2009; André, Baldoquín, *et al.* 2009).

- Un conjunto de restricciones garantizan que en el equipo de desarrollo se representen las tres categorías de roles propuestas por Belbin: roles de acción, mentales y 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 Cerebro.
- La persona que desarrolla el rol Jefe de Proyecto debe ser extrovertida y planificada (subtipo E_ _J) según el test de Myers-Briggs.

Algoritmos metaheurísticos de trayectoria en el problema de conformación de equipos

En los problemas de optimización multiobjetivo existen diversas maneras de representar las preferencias del decisor (Coello, Veldhuizen, *et al.* 2007). En este sentido y basado en la información que estos posean, es posible establecer pesos entre los objetivos, prioridades entre los objetivos o no establecer ni pesos ni prioridades entre ellos. En este trabajo se emplean dos métodos (Coello, Veldhuizen, *et al.* 2007): factores ponderados y multiobjetivo puro.

El método de factores ponderados se basa en la idea de convertir el problema multiobjetivo en un problema escalar, construyendo una única función objetivo que sea la suma de las funciones objetivos iniciales, ponderadas según un peso que le asigna el decisor a cada una de ellas.

Se dice que una solución es no dominada si no existe ninguna otra solución factible que mejore algún objetivo sin empeorar al menos uno de los otros objetivos (Coello, Veldhuizen, *et al.* 2007).

[1]Test creado por el Dr. Meredith Belbin, cuya 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

Dado que el modelo de conformación de equipos de software corresponde a un problema combinatorio, se utilizaron algoritmos metaheurísticos para su solución.

Los algoritmos metaheurísticos se pueden clasificar en algoritmos basados en trayectoria (también llamados s-metaheurísticas) y algoritmos basados en poblaciones (también llamados p-metaheurísticas) (Talbi 2009).

En este trabajo se emplean algoritmos metaheurísticos de trayectoria para la solución del modelo, tomando en consideración lo planteado en el teorema No Free Lunch (Wolpert and Macready 1997) y que los algoritmos de trayectoria pueden utilizar operadores más simples y necesitan menos memoria que los poblacionales, considerándose por tanto algoritmos más sencillos.

Con el propósito de darle soporte al modelo e implementar los métodos y algoritmos de solución antes mencionados, se desarrolló una herramienta de soporte a la decisión denominada Teamsoft+.

Al incorporar diferentes métodos y algoritmos, la herramienta permite la realización de estudios experimentales. En trabajos anteriores (André 2009; Infante 2012) se realizaron experimentos con datos que simulan organizaciones medianas (51-250 trabajadores), grandes (251-999 trabajadores) y especialmente grandes (más de 1000 trabajadores), evaluando la calidad de las soluciones y el tiempo de respuesta al aplicar algunas variantes de algoritmos metaheurísticos de trayectoria tales como Escalador de Colinas de Mejor Ascenso con Reinicio (Juels and Watenberg 1994; Jones 1995), Recocido Simulado (Kirkpatrick, Gelatt *et al.* 1983) , Búsqueda Tabú (Glover 1986), Híbridos de GRASP (Resende and Ribeiro 2003) como: GRASP y Escalador de Colinas de Mejor Ascenso, GRASP y Recocido Simulado, GRASP con Tabú, Escalador de Colinas Estocástico Multiobjetivo (Díaz 2001), Escalador de Colinas Estocástico Multiobjetivo con Reinicio (Díaz 2001), Escalador de Colinas Estocástico Multiobjetivo por mayor distancia (Díaz 2001), Búsqueda Tabú Multiobjetivo (Baykasoglu, Ozbaku *et al.* 2002), Recocido Simulado Multiobjetivo Multicaso (Haidine and Lehnert 2008) y Recocido Simulado Multiobjetivo de Ulungu (Ulungu and Teghem 1994).

Estrategias de paralelización en algoritmos metaheurísticos

Las estrategias de paralelización se pueden clasificar en métodos de un único camino y métodos de múltiples caminos (Verhoeven and Aarts 1995; Cung, Martins *et al.* 2001).

Las estrategias de paralelización en algoritmos metaheurísticos se utilizan fundamentalmente para aumentar la velocidad del recorrido secuencial (método de un único camino) o para permitir explorar una mayor porción del espacio de soluciones (método de múltiples caminos) .

En el método de un único camino la tarea cuya ejecución se paraleliza puede ser la evaluación de la función objetivo, la exploración del entorno de la solución actual o cualquier otra que requiera un uso intensivo de recursos computacionales (Verhoeven and Aarts 1995; Cung, Martins *et al.* 2001).

En el método de múltiples caminos se puede ejecutar en paralelo el mismo algoritmo, el mismo algoritmo con diferentes parámetros o diferentes algoritmos. (Verhoeven and Aarts 1995; Cung, Martins *et al.* 2001)

A continuación se describen algunos de los esquemas de paralelización que han sido aplicados a algoritmos metaheurísticos (Baños 2006):

El modelo de vecindario paralelo divide el vecindario en partes que se exploran de forma simultánea. Esto resulta particularmente interesante cuando la evaluación de las soluciones es muy costosa y/o el tamaño del vecindario es grande.

El modelo multiarranque consiste en ejecutar en paralelo varios buscadores locales sin intercambio de información. Este modelo intenta mejorar la calidad de la búsqueda aprovechando la diversidad ofrecida por cada ejecución independiente, y haciendo uso de diferentes parámetros.

El modelo maestro-esclavo permite mantener la secuencialidad del algoritmo original. Un procesador maestro centraliza la población y gestiona la selección y los reemplazos de individuos. Este también se encarga del envío de sub-poblaciones a los esclavos, los cuales ejecutan tareas de evaluación, y aplicación de ciertos operadores. Tras esto, los esclavos devuelven las soluciones evaluadas al maestro. Esta aproximación es especialmente utilizada cuando el costo computacional de generar y evaluar nuevas soluciones es elevado.

El modelo de islas divide la población original en un conjunto de sub-poblaciones distribuidas entre diferentes procesadores. Cada procesador es responsable de la gestión de la sub-población asignada, de forma que ejecuta todos los pasos de la metaheurística y ocasionalmente, migran individuos entre islas.

En la literatura se pueden encontrar varias propuestas exitosas de algoritmos metaheurísticos con estrategias de paralelización, entre las que se encuentran:

En (Luer, Venegas *et al.* 2009) se describe la implementación paralela del algoritmo genético con poblaciones generadas por búsqueda en vecindarios variables, donde cada uno de los procesadores crea una población a partir de las soluciones generadas. Se propone una aproximación híbrida MPI/OpenMP basada en el modelo maestro-esclavo para paralelizar una versión del algoritmo NSGA-II (Fast Nondominated Sorting Genetic Algrithm) adaptada al problema de la inferencia filogenética.

En (Kliewer and Tsch¨oke 2000) se propone una biblioteca de programas para el Recocido Simulado que brinda entre sus métodos de paralelización una combinación de métodos, de un único camino con métodos de múltiples caminos.

Cada procesador comienza evaluando su propio espacio de búsqueda y se comunican entre sí después de que cada nivel de temperatura alcanza su valor final. Esta estrategia es aplicada al problema de asignación de flota.

En (Luer, Venegas *et al.* 2009) se describe la implementación paralela del algoritmo genético con poblaciones generadas por búsqueda en vecindarios variables, donde cada uno de los procesadores crea una población a partir de las soluciones generadas. Se propone una aproximación híbrida MPI/OpenMP basada en el modelo maestro-esclavo para paralelizar una versión del algoritmo NSGA-II (Fast Nondominated Sorting Genetic Algrithm) adaptada al problema de la inferencia filogenética.

En (Kliewer and Tsch¨oke 2000) se propone una biblioteca de programas para el Recocido Simulado que brinda entre sus métodos de paralelización una combinación de métodos, de un único camino con métodos de múltiples caminos. Cada procesador comienza evaluando su propio espacio de búsqueda y se comunican entre sí después de que cada nivel de temperatura alcanza su valor final. Esta estrategia es aplicada al problema de asignación de flota.

En (Badeau, Guertin *et al.* 1997) se presentan estrategias de un único camino y múltiples caminos para darle solución al problema de enrutamiento de vehículos. Cada uno de los procesadores con una solución actual ejecuta la variante secuencial del algoritmo.

En (Salto 2009) se presentan metaheurísticas paralelas heterogéneas donde cada uno de los hilos de búsqueda utiliza un procedimiento de búsqueda distinto, ya sea a nivel algorítmico o a nivel de configuración de parámetros.

En (Fajardo 2009) se presenta la paralelización de PSO (*Particle Swarm Optimization*) y AGEDA (*Adaptive Gibbs sampling Estimation Density Algorithm*) para reducir su tiempo de cómputo e intentar mejorar su desempeño aplicando diferentes formas de paralelización, entre las que se encuentran la paralelización de funciones y procesos paralelos con intercambio de soluciones.

En (Luer, Venegas *et al.* 2009) se describe la implementación paralela del algoritmo genético con poblaciones generadas por búsqueda en vecindarios variables reducida, donde cada uno de los procesadores crea una población a partir de las soluciones generadas.

En (Bouthillier and Crainic 2001) se realizan estudios con estrategias que combinan diferentes metaheurísticas.

En (Porto and Ribeiro 1995) se propone la descomposición de la vecindad para ejecutar el algoritmo Búsqueda Tabú en diferentes procesadores.

Tecnologías para dar soporte a la paralelización

En la Tabla 1 se realiza una comparación de las tecnologías para dar soporte a la paralelización. Los criterios de comparación tienen en cuenta: el soporte a la programación en memoria compartida, el soporte a la programación por paso de mensaje, el diseño escalable y el lenguaje de programación que emplea.

Tabla 1: Comparación de las tecnologías para dar soporte a la paralelización

Tecnología	Memoria compartida	Memoria distribuida	Diseño escalable	Lenguaje de programación
OpenMP	Si	No	Si	C, C++,Fortran
MPI	No	Si	Si	C, Fortran
JP	Si	Si	Si	Java
JOMP	Si	No	No	Java
CUDA	Si	No	Si	C

Todas las tecnologías analizadas soportan la programación en memoria compartida, pero solo MPI y JP soportan la programación por paso de mensaje. JOMP no permite realizar un diseño escalable, ya que presentan dificultades en el método que devuelve el número de hilos dentro de una región paralela.

Se decide el uso de la tecnología JP ya que soporta la programación en memoria compartida en Java y permite realizar un diseño escalable en la aplicación, es decir la aplicación se adapta a las características de la computadora en la que se esté corriendo.

Implementación de algoritmos metaheurísticos de trayectoria con estrategias de paralelización como solución al modelo para la conformación de equipos

Analizando los esquemas de paralelización que han sido aplicados a algoritmos metaheurísticos en la bibliografía, se concluye que:

- No es factible dividir el vecindario en partes, ya que se dejarían de contemplar combinaciones de equipo que pueden ser buenas soluciones.
- El modelo maestro-esclavo y el modelo de islas está diseñado fundamentalmente para aquellos problemas cuyo costo computacional de generar y evaluar nuevas soluciones es elevado. Sin embargo, en el problema que se trata no resulta costoso la evaluación de las funciones objetivos. Algunos experimentos realizados permiten corroborar que solo se emplea como promedio del 2 al 10% del tiempo total de ejecución del algoritmo en la evaluación de las funciones objetivos.
- El modelo multiarranque es aplicable al problema, ya que puede permitir en algunos casos mejorar la calidad de las soluciones, a través de la diversificación de la búsqueda y en otros, mejorar los tiempos de respuesta.

Teniendo en cuenta esto, se decidió implementar el método de múltiples caminos. Se aplicaron dos estrategias que consistieron en ejecutar el mismo algoritmo en varios hilos de ejecución, la primera, sin ampliar el espacio de búsqueda, con el objetivo de disminuir el tiempo de respuesta y la segunda, ampliando el espacio de búsqueda, con el

objetivo de mejorar la calidad de la solución. A continuación se describen los experimentos realizados y los principales resultados alcanzados.

Resultados y discusión

Para la realización de los experimentos, se generaron varios juegos de datos que incluyeron toda la información relacionada con los trabajadores: competencias genéricas y técnicas, carga de trabajo, incompatibilidades con otros trabajadores y características psicológicas teniendo en cuenta los test de Myers-Briggs y Belbin. Las pruebas realizadas permitieron evaluar el desempeño de los algoritmos en organizaciones especialmente grandes, teniendo en cuenta la clasificación dada en (Hunter 2004), donde se considera como grandes aquellas organizaciones que poseen más de 250 empleados. En este trabajo se consideran como organizaciones muy grandes a aquellas organizaciones que tienen más de 1000 empleados.

Por otra parte, en los escenarios que representan organizaciones especialmente grandes fue donde menor por ciento del espacio de soluciones fue explorado en los experimentos anteriores y donde los tiempos de respuesta exceden los 20 minutos con algunos algoritmos, por lo que se decide generar los casos de prueba en función de este escenario.

Escenario 1: 1500 trabajadores y 10 roles a cubrir.

Los algoritmos utilizados para aplicar las estrategias fueron Búsqueda Tabú y Recocido Simulado Multiobjetivo Multicaso.

Los métodos de solución empleados y los valores de pesos establecidos fueron:

- Método de factores ponderados: Se utilizaron como pesos de los objetivos los obtenidos en (André 2009) al aplicar encuestas a expertos: maximizar competencias (0,36), balancear carga de trabajo (0,31) y minimizar incompatibilidades (0,33) .
- Multiobjetivo puro.

Los algoritmos fueron implementados en Java bajo el ambiente de desarrollo Eclipse 3.2, compilados con el JDK 1.6.0. Las ejecuciones fueron realizadas en un Intel Core 2 Duo con 2.66 GHz y 2 Gb de RAM.

Se realizaron 20 ejecuciones de cada uno de los algoritmos implementados.

Los casos de prueba ejecutados tuvieron en cuenta las tres funciones objetivos del modelo por defecto: maximizar competencias, minimizar incompatibilidades y balancear carga de trabajo, teniendo en cuenta las restricciones de que un trabajador solo puede desempeñar un rol y las competencias mínimas establecidas para cada rol.

Además, se tomaron en cuenta las restricciones asociadas con la sinergia del equipo: considerar la presencia de todas las categorías de roles de Belbin, elegir la presencia de al menos una persona con el rol cerebro y que exista un

balance entre las categorías de roles de Belbin (los roles de acción deben ser mayores que los roles mentales y los roles mentales deben ser mayores que los roles sociales).

Los parámetros utilizados para la ejecución de los diferentes algoritmos se muestran en la Tabla 2. Todos los algoritmos utilizados toman como condición de parada un número máximo de evaluaciones de las funciones objetivo.

Tabla 2: Parámetros de algoritmos

Algoritmos/Parámetros	Escenario 1
Búsqueda Tabú	
Tamaño de la lista tabú	20
Cantidad de iteraciones	7
Recocido Simulado	
Temperatura inicial	10
Temperatura final	0
Esquema de enfriamiento	Esquema de Cauchy (Tk = T0 /(1 + k)), donde k = # iteración y T0 es la temperatura inicial
Cantidad de iteraciones	90000
Recocido Simulado Multiobjetivo Multicaso	
Cantidad de iteraciones	90000

Métricas

Para evaluar el comportamiento de los diferentes algoritmos se evaluaron un conjunto de métricas que permiten determinar la calidad de las soluciones y el tiempo de respuesta.

En todos los casos de prueba se evaluará la métrica tiempo (tiempo máximo, tiempo mínimo, tiempo promedio), para determinar este factor.

Para evaluar cómo mejora el tiempo con las variantes de algoritmos que incluyen estrategias de paralelización en relación a las variantes secuenciales, se utilizará la métrica aceleración. Esta métrica busca que la reducción del tiempo sea proporcional de forma lineal con respeto al número de procesadores empleados (León, Framiñán *et al.* 2004). El cálculo de la métrica se muestra a continuación.

$$\text{aceleración } (p) = \frac{\text{tiempo}_{\text{algoritmo secuencial}}}{\text{tiempo}_{\text{algoritmo paralelo (p)}}}$$

El valor ideal de esta métrica es que la aceleración sea igual al número de procesadores. Si las pruebas son realizadas en una computadora con dos procesadores, el valor ideal de la aceleración es 2.

Métricas de rendimiento para el método de factores ponderados

La calidad de las soluciones se obtiene a partir de la evaluación máxima, mínima y promedio de la función objetivo resultante de aplicar el método. Todas las funciones objetivo fueron transformadas en funciones de maximización, por lo que mientras mayor es el valor de la métrica, mejor es la calidad de las soluciones.

Métricas de rendimiento para el método multiobjetivo puro

Revista Cubana de Ciencias Informáticas
Vol. 8, No. 3, Julio-Septiembre, 2014
ISSN: 2227-1899 | RNPS: 2301
Pág. 53-68
http://rcci.uci.cu

La calidad de las soluciones será medida a través de las métricas tasa de error y dispersión. La tasa de error (Veldhuizen and Lamont 2000) indica el porciento de soluciones encontradas que no son miembros del frente de Pareto verdadero. El valor ideal de la métrica es 0, indicando que todos los vectores son parte del frente de Pareto verdadero.

La dispersión mide la distribución de las soluciones encontradas en el frente de Pareto, y se calcula a través de la varianza de la distancia de cada miembro del conjunto de óptimos de Pareto encontrados con respecto a su vecino más cercano, tal como se muestra en la siguiente ecuación (Schott 1995).

$$ESS = \sqrt{\frac{1}{e-1}\sum_{i=1}^{e}(\bar{d} - d)^2}, d_i = \min_j\{|f_1^i - f_1^j| + |f_2^i - f_2^j|\} \quad (2)$$

donde i, j = 1,…,n y $\bar{d}$ se refiere a la media de todas las d_i y e es el número de elementos del conjunto de Pareto obtenidos hasta el momento.

El valor ideal de la métrica es 0, indicando que el algoritmo ha encontrado la distribución ideal de vectores no dominados.

Para la determinación del frente de Pareto verdadero se tomaron aquellas soluciones no dominadas de todas las obtenidas (por los diferentes algoritmos), lo cual es una aproximación al frente de Pareto real que se desconoce.

Análisis de los resultados obtenidos

Los experimentos realizados perseguían dos objetivos fundamentales: mejorar la calidad de las soluciones y disminuir los tiempos de respuesta. En la Tabla 3 se muestran los parámetros de los diferentes casos de prueba diseñados.

Tabla 3: Parámetros de los casos de prueba

Casos	Método de solución	Algoritmo	Estrategia
1 y 3	Factores ponderados	Búsqueda Tabú	Investigacion en paralelo de multiples trayectorias
2 y 4	Multiobjetivo puro	Recocido Simulado Multiobjetivo Multicaso	

A continuación se exponen los resultados obtenidos en los diferentes casos de prueba diseñados.

1. Casos de prueba para cumplir con el objetivo de mejorar la calidad de las soluciones

El objetivo de estas pruebas fue mejorar la calidad de las soluciones, explorando un mayor por ciento del espacio de búsqueda (duplicando el número de iteraciones).

Caso de prueba 1

Como se observa en la Tabla 4, tanto en el peor de los casos, como en los casos mejor y promedio se evidencia una ligera mejora en la calidad de las soluciones obtenidas con el algoritmo que implementa la investigación en paralelo

Revista Cubana de Ciencias Informáticas
Vol. 8, No. 3, Julio-Septiembre, 2014
ISSN: 2227-1899 | RNPS: 2301
Pág. 53-68
http://rcci.uci.cu

de múltiples trayectorias respecto al secuencial. Los tiempos se comportan similares en ambas variantes, incluso muestra una ligera mejoría con la variante que utiliza la estrategia de paralelización.

Tabla 4: Comparación de los algoritmos Búsqueda Tabú secuencial y paralelo utilizando el método de solución de factores ponderados

Métrica		Búsqueda Tabú Secuencial	Búsqueda Tabú con la estrategia de múltiples trayectorias
Evaluación	Mínimo	0,891	0,893
	Máximo	0,89	0,90
	Promedio	0,895	0,898
Tiempo (segundos)	Mínimo	55,23	51,12
	Máximo	68,21	65,94
	Promedio	61,33	55,67

Caso de prueba 2

Como se observa en la Tabla 5, la calidad de las soluciones respecto a la métrica tasa de error no mejora en la variante del algoritmo que emplea investigación en paralelo de múltiples trayectorias y la dispersión se mantiene con valores ideales en ambas variantes. Los tiempos se mantienen constantes en ambas variantes.

Tabla 5: Comparación de los algoritmos Recocido Simulado secuencial y paralelo utilizando método de solución multiobjetivo puro

Métrica		Recocido Simulado Multiobjetivo Multicaso Secuencial	Recocido Simulado Multiobjetivo Multicaso con la estrategia de múltiples trayectorias
Tasa de error	Mínimo	0,0	0,24
	Máximo	0,666	0,757
	Promedio	0,236	0,496
Dispersión	Mínimo	0,0	0,0
	Máximo	0,0	0,0
	Promedio	0,0	0,0
Tiempo (segundos)	Mínimo	89,20	89,20
	Máximo	140,62	159,86
	Promedio	110,95	116,18

2. Casos de prueba para cumplir con el objetivo de disminuir los tiempos de respuesta

Para evaluar la métrica de tiempo, se evaluaron en el primer escenario 45000 iteraciones en el algoritmo secuencial y 22500 en el algoritmo con estrategia de paralelización de múltiples trayectorias. De igual forma para el segundo escenario, se realizaron 90000 iteraciones del algoritmo secuencial y 45000 en el algoritmo con estrategia de paralelización de múltiples trayectorias.

Caso de prueba 3

Revista Cubana de Ciencias Informáticas
Vol. 8, No. 3, Julio-Septiembre, 2014
ISSN: 2227-1899 | RNPS: 2301
Pág. 53-68
http://rcci.uci.cu

Como se observa en la Tabla 6, se evidencia una mejora en cuanto al tiempo del algoritmo que implementa la investigación en paralelo de múltiples trayectorias respecto al secuencial, lográndose como promedio una disminución del tiempo del 26% (aceleración de 1.36).

Tabla 6: Comparación de los algoritmos Búsqueda Tabú secuencial y paralelo utilizando el método de solución de factores ponderados

Métrica		Búsqueda Tabú Secuencial	Búsqueda Tabú con la estrategia de múltiples trayectorias	Aceleración
Tiempo (segundos)	Mínimo	55,23	39,3	1,4
	Máximo	68,21	52,18	1,3
	Promedio	61,33	45,08	1,36
Evaluación	Mínimo	0,891	0,871	-
	Máximo	0,89	0,887	-
	Promedio	0,895	0,874	-

Caso de prueba 4

Como se observa en la Tabla 7, se evidencia una mejora en cuanto al tiempo del algoritmo que implementa la investigación en paralelo de múltiples trayectorias respecto al secuencial, lográndose como promedio una disminución del tiempo del 19% (aceleración de 1.23). La calidad de las soluciones es similar como promedio en ambas variantes.

Tabla 7: Comparación de los algoritmos Recocido Simulado multiobjetivo multicaso secuencial y paralelo utilizando método de solución multiobjetivo puro

Métrica		Recocido Simulado Multiobjetivo Multicaso Secuencial	Recocido Simulado Multiobjetivo Multicaso con la estrategia de múltiples trayectorias	Aceleración
Tiempo (segundos)	Mínimo	89,20	69,49	1,3
	Máximo	140,62	109,30	1,28
	Promedio	110,95	89,75	1,23
Tasa de error	Mínimo	0,0	0,24	-
	Máximo	0,666	0,757	-
	Promedio	0,236	0,296	-
Dispersión	Mínimo	0,0	0,.0	-
	Máximo	0,0	0,.0	-
	Promedio	0,0	0,0	-

Al analizar los resultados obtenidos, se observa que al aplicar la estrategia investigación en paralelo de múltiples trayectorias se logra una disminución considerable del tiempo en todos los casos de prueba analizados, destacándose el Recocido Simulado Multiobjetivo Multicaso donde se reduce en un 42% el tiempo de respuesta con una aceleración de 1.7. Resulta importante señalar que a pesar del resultado alcanzado la aceleración no logra el valor ideal.

Adicionalmente, aplicando la estrategia investigación en paralelo de múltiples trayectorias se logra una mejora en la calidad de las soluciones en la mayoría de los casos, exceptuando cuando se aplica el método multiobjetivo puro con el algoritmo Recocido Simulado Multiobjetivo Multicaso.

Conclusiones

A partir de los resultados obtenidos se puede concluir que la propuesta de utilizar la estrategia de investigación en paralelo de múltiples trayectorias, logra resultados ligeramente mejores en cuanto a calidad de las soluciones, con una reducción del tiempo de entre 19 y 42 porciento respecto a los algoritmos secuenciales, por lo que se considera factible su utilización para solucionar el modelo de conformación de equipos de software.

Referencias

- ANDRÉ, M. Un modelo para la asignación de recursos humanos a equipos de proyectos de software. Tesis de doctorado, ISPJAE, Ciudad de La Habana, Cuba, 2009
- ANDRÉ, M., M. BALDOQUÍN, *et al.* Gestión del conocimiento para la identificación de patrones útiles en la formación de equipos de proyectos de software. En: XII Taller Internacional de Gestión Tecnológica e Innovación y su Aplicación en las organizaciones. 2009
- ANDRÉ, M., M. G. BALDOQUÍN, *et al.* A formalized model for the assignment of human resources to software projects. XIV Latin Ibero-American Congress on Operations Research CLAIO 2008, Colombia.
- BADEAU, P., F. GUERTIN, *et al.* A parallel tabu search heuristic for the vehicle routing problem with time windows. Transportation Research-C 5, 1997: 109-122.
- BAÑOS, R. Meta-heurísticas Híbridas para Optimización Mono-objetivo y Multi-objetivo. Paralelización y Aplicaciones. Universidad de Almería, 2006
- BAYKASOGLU, A., L. OZBAKU, *et al.* Multiple dispatching rule based heuristic for multiobjective scheduling of job using Tabu Search 5th International Conference on Managing Innovations in Manufacturing, Milwaukee, Wisconsin, USA.
- BOUTHILLIER, A. L. and T. G. CRAINIC. Parallel Co-operative Multi-thread Meta-heuristic for the Vehicle Routing Problem with Time Window Constraints. Publication, Centre de recherche sur les transports, 2001.
- COELLO, C. A. C., D. A. V. VELDHUIZEN, *et al.* Evolutionary Algorithms for Solving Multi-Objective Problems. New York, Springer, 2007. 800
- CRAINIC, T. G. and M. TOULOUSE. Parallelstrategiesformeta-Heuristics. 2002.
- CRUZ, C. and J. M. MORENO. Estrategias cooperativas paralelas en la solución de problemas de optimización. Revista Iberoamericana de Inteligencia Artificial, 2007 **11**(34): 16.
- CUNG, V., S. L. MARTINS, *et al.* Strategies for the parallel implementation of Metaheuristics. Essays and Surveys in Metaheuristics 2001: 33.

- DÍAZ, R. Estudio de la capacidad del algoritmo Escalador de Colinas Estocástico para enfrentar problemas multiobjetivo. Master en Informática Aplicada, Instituto Superior Politécnico "José Antonio Echeverría", Ciudad de la Habana, 2001
- EIBEN, A. E., R. HINTERDING, *et al.* Parameter control in evolutionary algorithms. Evolutionary Computation, IEEE Transactions on, 2002.
- FAJARDO, J. Algoritmo Multigenerador de soluciones para la competencia y colaboración de generadores metaheurísticos. Tesis de Maestría, CUJAE, 2009
- FALCO, I. D., R.D BALIO, *et al.* Improving search by incorporating evolution principles in parallel tabu search. 1994.
- GLOVER, F. Future Paths for Integer Programming and Links To Artificial Intelligence. Computers & Operations Research, 1986 **13**(5): 17.
- HAIDINE, A. and R. LEHNERT. Multi-Case Multi-Objective Simulated Annealing (MC-MOSA): New Approach to Adapt Simulated Annealing to Multi-objective Optimization. International Journal of Information Technology, 2008 **4**(3): 9.
- HUNTER, G. M. Information Systems & Small Business: Research Issues. Journal of Global Information Management, 2004 **14**(4): 1-5.
- INFANTE, A. Algoritmos de trayectoria multiobjetivo aplicados al problema de asignación de recursos humanos a equipos de proyecto de software. Instituto Superior Politécnico José Antonio Echeverría, 2012
- JONES, T. C. Evolutionary Algorithms, Fitness Landscapes and Search. University of New Mexico, Alburquerque, 1995
- JUELS, A. and M. WATENBERG (1994). Stochastic Hill-Climbing as a Baseline Method for Evaluating Genetic Algorithms, University of California at Berkeley.
- KIRKPATRICK, S., C. D. GELATT, *et al.* Optimization by Simulated Annealing. Science, 1983 **220**: 10.
- KLIEWER, G. and S. TSCH¨OKE. A General Parallel Simulated Annealing Library and its Application in Airline Industry. In Proceedings of the 14th International Parallel and Distributed Processing Symposium, 2000: 55-61.
- LEE, F. A. Parallel Simulates Annealing on a Message-Passing Multi-Computer. UTAH STATE UNIVERSITY, 1995
- LEÓN, J. M., J. M. FRAMIÑÁN, *et al.* (2004). Implementación de meta-heurísticas en paralelo mediante LAM-MPI: Situación actual y perspectivas de futuro*. VIII Congreso de Ingeniería de Organización.
- LUER, A., B. VENEGAS, *et al.* Estrategias de Paralelización de Metaheurísticas aplicadas a problemas de localización de instalaciones Ingeniería Industrial, 2009 **8**: 75 - 90.
- M. ANDRÉ, M.G. BALDOQUÍN, *et al.* Formal model for assigning human resources to teams in software projects. Information and Software Technology, 2011 **53**(3): 16.
- MALEK, M., M. GURUSWAMY, *et al.* Serial And Parallel Simulated Annealing And Tabu Search Algorithms For The Traveling Salesman Problem. Annals of Operations Research, 1989 **21**: 59-84.
- PORTO, S. C. and C. C. RIBEIRO. Parallel tabu search message-passing synchronous strategies for task scheduling under precedence constraints. Journal of Heuristics, 1995 **1**(2): 207-223.

- RESENDE, M. G. C. and C. S. RIBEIRO. Chapter 8: Greedy Randomized Adaptive Search Procedure. Handbook of Metaheuristics. F. Glover and G. A. Kochenberber, 2003, Kluwer Academics**:** 221-249.
- RODRÍGUEZ, L. Modelación Formal del problema de asignación de recursos humanos a proyectos de softwar. Instituto Superior Politécnico "José Antonio Echeverría", La Habana, 2008
- SALTO, C. Metaheuristicas Híbridas Paralelas para Problemas Industriales De Corte, Empaquetado Y Otros Relacionados. Doctoral, Universidad Nacional De San Luis, San Luis, Argentina, 2009
- SCHOTT, J. R. Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Maestría, 1995
- TALBI, E.-G. Metaheuristics : From Design to Implementation. New Jersey, John Wiley & Sons, Inc., 2009. 593
- ULUNGU, E. L. and J. TEGHEM. Multi-objective combinatorial optimization problems: A survey. Journal of Multi-Criteria Decision Analysis, 1994 **3**: 22.
- VELDHUIZEN, D. A. V. and G. B. LAMONT. Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. Evolutionary Computation, 2000 **8**(2): 23.
- VERHOEVEN, M. and E. AARTS. Parallel local search. Journal of Heuristics, 1995 **1**: 43-65.
- WOLPERT, D. H. and W. G. MACREADY. No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1997 **1**(1): 67–82.