Programación paralela en una técnica de optimización de tiempos de producción de una empresa

Laura Vázquez, Alicia Valdez, Gloria Campos, Raúl Campos y Rubén Hernández

Laura Vázquez, Alicia Valdez, Gloria Campos, Raúl Campos y Rubén Hernández
Facultad de Ingeniería Mecánica y Eléctrica, Universidad Autónoma de Coahuila, Barranquilla s/n colonia Guadalupe,C.P. 25700,Monclova Coahuila
lauracristina_vazquez@hotmail.com

M. Ramos., V.Aguilera., (eds.) .Ciencias de la Ingeniería y Tecnología, Handbook -©ECORFAN- Valle de Santiago, Guanajuato, 2014.

Abstract

Parallel programming is a mechanism used to solve problems in which the resources of a single machine are not enough. The aim of paralleling the genetic algorithm is to reduce the processing time by distributing tasks among the available processors. For the design of this genetic algorithm, sixty activities production times of a company were considered. As for the parallel processing, critical section where identified (those that consume greater amount of computational resources on which is necessary to work independently), it was divided dynamically between the number of available processors, set the number of iterations to perform on the assigned critical functions (stop condition) and a data set for each processor was assigned independently.

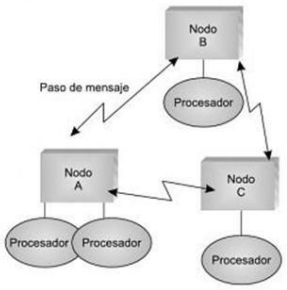
7 Introducción

Con la evolución de las computadoras, han aparecido computadores con más de un procesador o capaces de ejecutar varios procesos a la vez. Ello ha motivado la necesidad de la aparición de lenguajes capaces de aprovechar las posiblidadades de las máquinas de este tipo. Un programa paralelo es el que se ejecuta en varios procesadores de este tipo. Se pueden escribir aplicaciones paralelas usando lenguajes clásicos y bibliotecas de funciones de comunicaciones entre procesos (Vázquez, Marco, Martin, & Molinero, 2006).

La programación paralela se utiliza para resolver problemas en los que los recursos de una sola máquina no son suficientes. La finalidad de paralelizar un algoritmo es disminuir el tiempo de procesamiento mediante la distribución de tareas entre los procesadores disponibles.

Palma, Garrido, Sánchez y Quesada, mencionan que un programa paralelo es un tipo de programa concurrente diseñado para ejecutarse en un sistema multiprocesador. Además, consideran que un programa distribuido es aquel que está diseñado para ejecutarse en una red de procesadores autónomos que no comparten una memoria común.

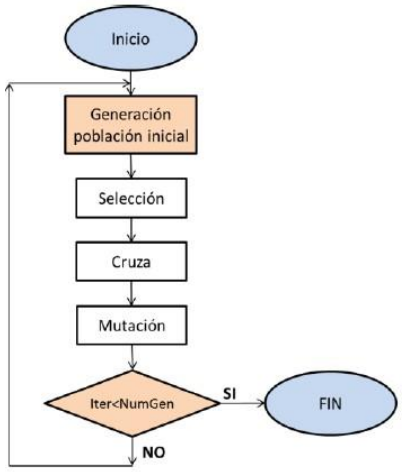
Figura 7 Programa paralelo distribuido en diferentes procesadores



Un algoritmo genético permite resolver problemas de optimización (Sivanandam & Deepa, 2008). Considerando lo anterior, es posible desarrollar y utilizar un algoritmo genético para calcular los tiempos de producción en una empresa, mediante procesos paralelos de forma práctica.

La siguiente figura muestra las partes que componen un algoritmo genético simple.

Figura 7.1 Algoritmo genético simple



7.1 Metodologia

Los algoritmos genéticos son técnicas de optimización estocástica que trabajan mediante la generación de soluciones aleatorias. La teoría de los algoritmos genéticos fue propuesta por Holland (Holland, 1975) y desarrollada posteriormente por Goldberg (Goldberg, 1989) y se basa en la evolución natural de los seres vivos (Figura).

El funcionamiento básico de un algoritmo genético se indica a continuación:

Los algoritmos genéticos para trabajar requieren que las variables de diseño (propiedades medibles del objeto) sean codificadas en cadenas de longitud finita, en cualquier alfabeto. Uno de los más usados es el código binario al ser flexible y fácilmente programable, lo importante es que puedan adaptarse al contexto del problema a ser resuelto y cumplan con los principios de los bloques de construcción (Cerrolaza & Annicchiarico, 1996).

1. Crear una función objetivo del problema.
2. Generación de la población inicial. Normalmente se genera de forma aleatoria y está formada por un conjunto de cromosomas (individuos).
3. Evaluación de los cromosomas. Se calcula la función objetivo para cada uno de los cromosomas que forman la población actual. Cerrolaza y Annicchiarico comentan que la función objetivo toma un cromosama como entrada y retorna un número o una lista de números que miden el éxito del cromosoma sobre el problema a resolver. Estas funciones juegan el mismo papel del ambiente en la evolución natural.
4. Selección. Consiste en seleccionar los cromosomas que serán cruzados en la siguiente generación. Los de mejor aptitud tienen mayor probabilidad de ser seleccionados.
5. Cruza. Es el intercambio de material genético entre dos cromosomas a la vez para generar dos descendientes.
6. Mutación. Cambio de un gen por otro causando pequeñas alteraciones en puntos determinados.

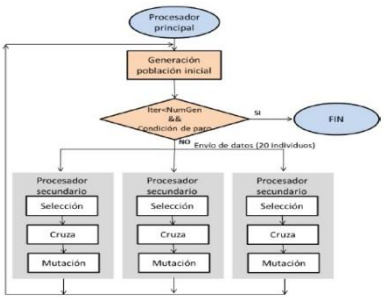
Antes de empezar a aplicar un algoritmo genético, se necesita diseñar una codificación para las soluciones, siendo éste uno de los aspectos fundamentales del diseño, y por tanto de la posterior eficiencia del algoritmo genético. Es decir, es necesario definir cómo representar de forma apropiada cada una de las posibles soluciones al problema (Maroto, Alcaraz, & Ruiz, 2002).

Para el diseño de este algoritmo se consideraron los tiempos de producción de sesenta actividades de una empresa de manufactura.

Los pasos utilizados en la metodología mediante la programación paralela partiendo de un algoritmo genético simple, se muestran en la figura, y a continuación se explican.

- Primeramente se identifican las secciones críticas, es decir, aquellas que consuman mayor cantidad de recursos computacionales, en las que es necesario trabajar de forma independiente.
- En seguida se dividen de forma dinámica los procesos entre el número total de procesadores disponibles (en esta investigación se utilizaron tres).
- Se copia el programa a cada procesador, indicándole las funciones que debe ejecutar.
- Después se determina el número de iteraciones a realizar en las funciones críticas asignadas (condiciones de paro).
- Por último se asigna un conjunto de datos para cada procesador que trabajará de forma independiente.

Grafico 7 Diseño del algoritmo genético con programación paralela



El algoritmo genético fue diseñado y estructurado en el lenguaje de programación C++, considerando que la programación paralela necesita manejar la asignación de las tareas de selección, cruza y mutación en los tres procesadores del sistema, teniendo la oportunidad de variar el número de procesadores disponibles, según sea necesario. Hay cambios profundos en la manera en que el sistema operativo adquiere recursos del sistema y los administra dentro de un proceso para ejecutar eficientemente los programas paralelos; estos cambios sólo son visibles en la programación.

7.2 Resultados y discusión

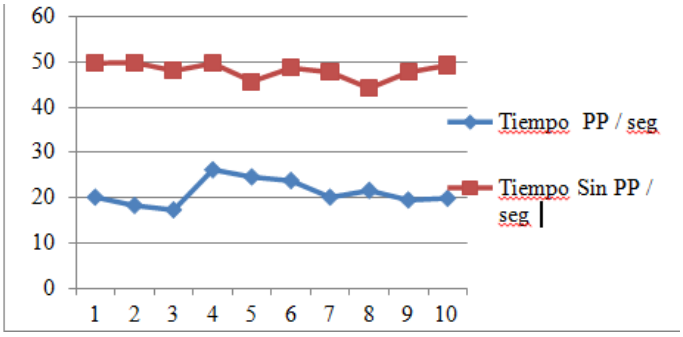
La reducción en los tiempos de cómputo obtenidos al momento de utilizar la programación paralela, mostró una diferencia significativa con respecto a la ejecución del algoritmo simple (ver Tabla 1). El tiempo promedio de ejecución se redujo aproximadamente en un 56%.

Tabla 7 Resultados obtenidos durante las diez ejecuciones

No. ejecución	Tiempos			Tiempo PP / seg	Tiempo Sin PP / seg	Diferencia seg	% reducción seg
	Proc 1	Proc 2	Proc 3				
1	18.9	21.4	20.1	20.1	49.8	29.7	59.64
2	17.9	18.2	18.9	18.3	49.9	31.6	63.33
3	16.8	18	17.1	17.3	48.1	30.8	64.03
4	28.4	24.5	25.8	26.2	49.7	23.5	47.28
5	24.9	23.9	24.9	24.6	45.7	21.1	46.17
6	23.8	23.9	23.7	23.8	48.7	24.9	51.13
7	18.2	22.1	20.1	20.1	47.8	27.7	57.95
8	19.8	21.3	23.9	21.6	44.2	22.6	51.13
9	17.9	19.8	20.8	19.5	47.8	28.3	59.21
10	18.3	21.3	20.1	19.9	49.3	29.4	59.63

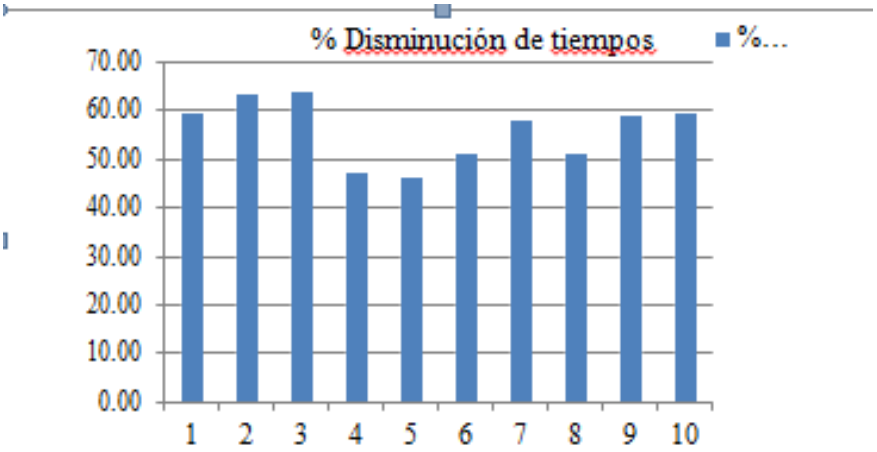
La siguiente figura muestra las diez ejecuciones y los respectivos tiempos utilizados al momento de utilizar la programación paralela en el algoritmo genético aplicado en la producción de una empresa, como se puede observar, existe una diferencia significativa en los tiempos de demora que emplean los tres procesadores para realizar sus procesos hasta llegar a la condición de paro del algoritmo genético.

El tiempo promedio mayor utilizando los tres procesadores mediante la programación paralela es de 26.2 segundos, mientras que el tiempo mayor utilizando un solo procesador fue de casi 50 segundos.



La figura muestra el porcentaje de disminución en los tiempos al comparar la programación paralela. Tomando en cuenta que en la ejecución número 3, se mostró poco más del 64% de disminución en el tiempo, en cambio en la ejecución número 5 se mostró el 46% de disminución, lo cual se debe a la independencia en el trabajo que realiza cada uno de los procesadores utilizados.

Figura 7.2 Porcentajes de disminución de tiempos calculados en las diez ejecuciones del algoritmo genético aplicando la programación paralela



Una desventaja muy importante se presenta cuando falla o se detiene algún procesador, ya que interrumpe la ejecución del algoritmo genético. La ventaja principal que se encontró en este diseño es que no existen dependencias entre los procesadores, por consiguiente, cada uno puede trabajar a diferente ritmo, a esto se debe las diferencias en los tiempos finales de cada procesador, tal como se observó en la tabla 1.

7.3 Conclusiones

La programación paralela permite disminuir el tiempo de ejecución de un programa y puede ser utilizada como una herramienta cuando los recursos de una sola computadora no son suficientes.

En este trabajo se optimizaron los tiempos de producción de ciertos procesos de producción de una empresa. La aplicación puede ser diferente, considerando que los algoritmos genéticos resuelven problemas de optimización.

Para efectos de mostrar resultados y realizar una comparación, en este trabajo solamente se realizaron diez ejecuciones. Es importante mencionar que el algoritmo genética cada vez que se ejecuta muestra uno de los mejores resultados.

La utilización del paralelismo es un punto de desviación para las industrias del software donde se deben adoptar nuevas técnicas. Los programadores deben considerar las técnicas del paralelismo en aquellas aplicaciones que dependen mucho del tiempo; o bien, en las que se espera un incremento importante en la cantidad de datos a manejar.

Agradecimientos

Se agradece el apoyo proporcionado por la Universidad Autónoma de Coahuila para el desarrollo de esta investigación.

Referencias

Cerrolaza, M., & Annicchiarico, W. (1996). *Algoritmos de optimización estructural basados en simulación genética.* Caracas: Consejo de Desarrollo Científico y Humanístico.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning.* Mass: Wesley Publishing .

Holland, J. H. (1975). *Adaptation in natural and artificial systems.* Cambridge: The MIT Press.
Maroto, C., Alcaraz, J., & Ruiz, R. (2002). *Investigación Operativa: Modelos y técnicas de Optimización.* Valencia: Universidad Politécnica de Valencia.

Palma, J. T., Garrido, M. d., Sánchez, F., & Quesada, A. (2003). *Programación concurrente*. Madrid: Thomson.

Sivanandam, S. N., & Deepa, S. N. (2008). *Introduction to Genetic Algorithms.* New York: Springer.

Vázquez, P., Marco, J., Martin, A., & Molinero, X. (2006). *Programación en C++ para ingenieros.* Madrid: Thomposn Editores Espain.