Black Hole Algorithm in Computational Offloading for Mobile Cloud Computing

Lehi Quincho
Escuela Profesional de Ciencia de la Computación
Universidad Nacional de San Agustín
Arequipa, Perú
lquincho@unsa.edu.pe

Karen E. Gordillo
Escuela Profesional de Ciencia de la Computación)
Universidad Nacional de San Agustín
Arequipa, Perú
kgordillo@unsa.edu.pe

***Resumen*—En los últimos años el uso de los smartphones se ha incrementado, y una gran cantidad de tareas cotidianas son realizadas en ellos, reemplazando en muchas veces a los ordenadores convencionales, y esto es debido a la portabilidad. Como es evidente, los smartphones no poseen muchas de las características que un ordenador si, como la capacidad de procesamiento, almacenamiento y limitaciones como el tiempo de duración de la batería y ancho de banda. Por tal motivo es necesario enviar sub tareas que mayor costo computacional a la nube para su procesamiento, y procesar las tareas cortas localmente, para así reducir el consumo de batería, todo esto considerando el tiempo que se tomará en transmitir, el ancho de banda y el tiempo de respuesta limite de la nube . En este articulo proponemos el uso del algoritmo de optimización Black Hole para la tarea de Offloading en Mobile Cloud.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introducción

Computational Offloading es un método donde algunas de las tareas de una aplicación móvil pueden ser enviadas para ejecutarse en servidores remotos en la nube, con el fin de ahorrar energía. Sin embargo, el problema de particionar las tareas de la aplicación es generalmente NP-completo. El objetivo principal del algoritmo de offloading es minimizar la energía total utilizada por la aplicación móvil, al tiempo que se cumple una restricción de tiempo de ejecución.

Una tarea a enviar debe transmitirse a través de una red de acceso inalámbrico, y debe considerarse el ancho de banda de transmisión inalámbrica que varía en el tiempo. Un algoritmo de envío adaptable puede determinar las decisiones de descarga dinámicamente de acuerdo con un entorno inalámbrico cambiante.

El presente trabajo está basado en la propuesta [1] que propone un algoritmo de programación dinámica llamado DPH. La programación dinámica es un enfoque de optimización que transforma un problema complejo en una secuencia de problemas más simples. El algoritmo genera cadenas de bits aleatorias de 0 y 1 periódicamente y utilizamos subcadenas cuando mejoran la solución (similar a la optimización genética). También se llenan tablas de programación dinámica para evitar el cálculo adicional de subcadenas comunes. A diferencia del trabajo señalado que usa programación dinámica, se propone el uso del algoritmo de optimización Black Hole [2] que es un algoritmo heurístico que está inspirado en el fenómeno de los agujeros negros. Al igual que otros algoritmos basados en la población, el algoritmo comienza con una población inicial de soluciones candidatas a un problema de optimización y una función objetivo que se desea minimizar o maximizar.

El resto del documento está organizado de la siguiente manera: la Sección II proporciona el modelo del sistema y la formulación del problema. La Sección III presenta el algoritmo propuesto. Los resultados de la evaluación y la simulación se muestran en la sección IV y el documento concluye en la sección V.

II. Modelo del sistema y formulación del problema

Piense en una aplicación que consta de un número de tareas, entre ellas hay tareas no descárgales(es decir, tareas locales) y tareas descargables. Se considera tareas locales a todas aquellas que manejan directamente la interacción con el usuario, acceden a dispositivos de E/S o a información específica en el dispositivo móvil.Por lo tanto, las tareas locales deben ser procesadas localmente por el usuario móvil.

Cuando una aplicación se ejecuta en un smartphone, este posee tareas que pueden ser enviadas a un servidor externo, y otras que no. Normalmente estas tareas locales son las que interacción directa con el usuario como, por lo que deberán ser procesadas localmente en el dispositivo móvil.

II-A. Modelo de red

Una aplicación instalada en un smartphone con conexión inalámbrica como WiFi tendrá que lidiar con la interferencia y la congestión de la red. El dispositivo móvil debe decidir si cada tarea debe procesarse localmente o enviarse a otro servidor, considerando el ancho de banda de transmisión de la red inalámbrica actual.

El tiempo necesario para transferir una tarea entre un dispositivo móvil y la nube a través de un enlace inalámbrico es un problema importante ya que existe una restricción de tiempo de ejecución total para todas las tareas.

II-B. Formulación del problema

Dada una tarea i, donde M_i 0, 1 sea una variable indicadora de ejecución. Sea $M_i = 1$ si la tarea i se ejecuta en el

dispositivo móvil y $M_i = 0$ en caso contrario. Si se ejecuta localmente, el consumo de energía es El_i. Er_i es el consumo de energía del dispositivo móvil cuando la tarea i se ejecuta en la nube, y Et_i es el costo de energía de transmitir la tarea i al servidor de la nube.

La variable Tl_i es el tiempo de ejecución local para procesar la tarea i, y Tr_i es el tiempo de ejecución remota cuando la tarea i se ejecuta en Está claro que la energía de transmisión utilizada para cargar cada tarea dependerá del ancho de banda de transmisión de la red. Por lo tanto, los cambios en el ancho de banda de la red inalámbrica afectarán la decisión de descarga. Por ejemplo, si suponemos que el tiempo de transmisión de cada tarea es igual al tamaño de cada tarea dividido por la velocidad de transmisión de la red, entonces cualquier variación en la velocidad de transmisión afectará la decisión final de descargar esta tarea o no.

La función de consumo de energía y su tiempo de ejecución correspondiente se definen en (1) y (2):

$$E = \sum (M_i El_i + (1 - M_i) Er_i + (1 - M_i) Et_i) \quad (1)$$

$$T = \sum (M_i Tl_i + (1 - M_i) Tr_i + (1 - M_i) Tt_i) \quad (2)$$

La ejecución en tiempo T de todas las taras deben satisfacer la condición donde $T_{constraint}$ es el tiempo de ejecución máximo requerido.

$$\begin{aligned} &minE \\ &T \leq T_{constraint} \end{aligned} \quad (3)$$

El número de combinaciones de valores binarios M_i para buscar la solución óptima crece exponencialmente con el número de tareas. Nuestro objetivo es determinar qué tareas deben descargarse en el servidor de la nube para minimizar la energía mientras se cumple la restricción de tiempo de ejecución de una aplicación móvil.

III. Algoritmo

Se inicializa una población N con posiciones aleatorias, para nuestro caso cada individuo generado aleatoriamente representará una posible solución representado como un array de 0 y 1, se evalúa la población con la función objetivo que deseamos minimizar, esta función es la de energía y se selecciona el mejor candidato con el menor fitness para ser el agujero negro, el resto forma las estrellas normales.

El agujero negro tiene la capacidad de absorber las estrellas que lo rodean. Después de inicializar el agujero negro y las estrellas, el agujero negro comienza a absorber las estrellas a su alrededor y todas las estrellas comienzan a moverse hacia el agujero negro. La absorción de estrellas por el agujero negro usa la siguiente formula:

$$x_i(t+1) = NOT\ x_i(t)\ OR\ x_{BH} \quad i = 1, 2, ..., N \quad (4)$$

- $X_i(t)$ y $X_i(t+1)$ son ubicaciones de la estrella i en la iteración t y $t+1$ respectivamente
- X_{BH} es la ubicación del agujero Negro en el espacio de busqueda.
- N es el número de estrellas (soluciones candidatas).

Mientras se mueve, una estrella puede alcanzar un lugar con un costo menor que el agujero negro. En tal caso, se intercambian las posiciones. Luego, el algoritmo BH continuará con el agujero negro en la nueva ubicación y las estrellas comenzarán a moverse hacia esta nueva ubicación.

Existe la probabilidad de cruzar el horizonte de eventos durante el movimiento de las estrellas hacia el agujero negro. Cada estrella (solución candidata) que cruza el horizonte de eventos del agujero negro será aspirada por el agujero negro. Cada vez que un candidato (estrella) muere (es absorbido por el agujero negro), otra solución candidata (estrella) nace y se distribuye al azar en el espacio de búsqueda y comienza una nueva búsqueda. Esto se hace para mantener constante el número de soluciones candidatas. La siguiente iteración tiene lugar después de que todas las estrellas se hayan movido.

El radio del horizonte de eventos en el agujero negro se calcula con la siguiente ecuación:

$$R = \frac{f_{BH}}{\sum_{i=1}^{N} f_i} \quad (5)$$

- f_{BH} es el fitness del agujero negro.
- f_i es el fitnes de la estrella i.
- N es el numero de estrellas (soluciones candidatas).

Cuando la distancia entre una estrella (solución candidata) y el agujero negro(mejor solución) es menor a R , la estrella muere y se reemplaza por otra aleatoriamente.

```
Inicializa la población de estrellas con ubicaciones
  aleatorias en el espacio de búsqueda;
while No se cumpla el criterio de finalización do
    Para cada estrella, se evalúa la función objetivo.;
    Seleccionamos la estrella con el mejor fitness
      como Agujero Negro.;
    Se cambia la ubicación de cada estrella de acuerdo
      a la ecuación de absorción.;
    if Una estrella alcanza un costo menor que el
      agujero negro then
        Se intercambia las ubicaciones;
    end
    if Una estrella cruza el horizonte de eventos del
      agujero negro then
        Se reemplaza con una nueva estrella generada
          aleatoriamente en el espacio de búsqueda;
    end
end
```

Algorithm 1: Algoritmo base

IV. Experimentos

Nuestra propuesta fue programada en Python 2.7 y usamos la biblioteca de Numpy, Adoptamos las mismas caracteristicas que fueron usadas en [1] , que corresponden a un Smatphone Nokia N900, con un conjunto de 15 tarea (N=15).

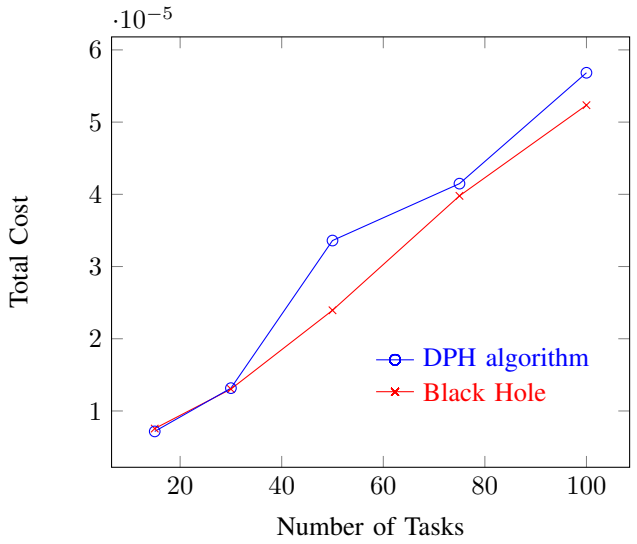

Figura 1. Costo total de energía(J) vs Numero de Tareas

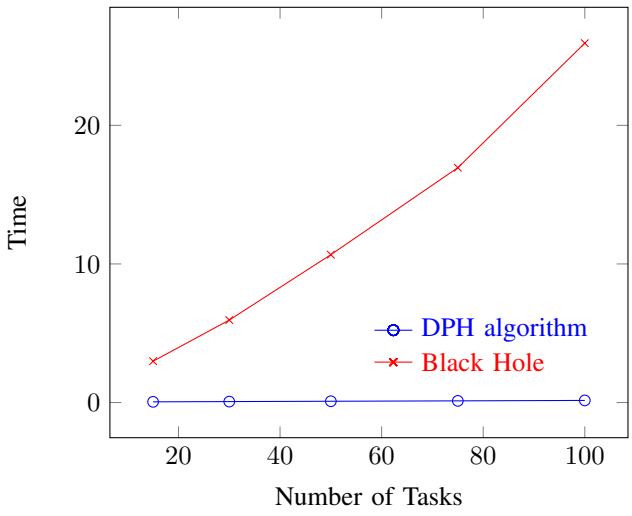

Figura 2. Costo total de energía(J) vs Numero de Tareas

El tiempo de procesamiento local es de $4{,}74x10^{-7}s/bit$ y el consumo de energía es de $3{,}25x10^{-7}$.

El consumo de energía para la transmisión y recepción de datos en el dispositivo móvil son de $1{,}42x10^{-7}J/bit$

El tamaño de datos de entrada y salida están distribuidos aleatoriamente entre 10MB a 30MB y de 1MB a 3 MB respectivamente. El tiempo de transmisión de cada tarea es igual al tamaño de cada tarea / el ratio de transmisión.

El ratio de procesamiento en la nube es de $10x10^9$ ciclos/s, y el poder de consumo de energía de este dispositivo es de 30 mW. El ratio de procesamiento local es de $500x10^6$ ciclos/s.

IV-A. Resultados de la Simulación

Como se puede apreciar en la figura 1 , el costo total obtenido con nuestro algoritmo de Agujero negro ha logrado superar en resultados al algoritmo propuesto, debido a que logra encontrar una configuración más eficiente que minimiza aún más el costo de energía.

La imagen 2 muestra una comparación del tiempo de ejecución del algoritmo base DPH con nuestra propuesta. Se puede apreciar que nuestro algoritmo es más lento debido a que realiza varias iteraciones y trabaja con una población aleatoria, esto hace el calculo más lento.

V. Conclusión

Finalmente podemos concluir que el algoritmo propuesto da unos mejores resultados en la configuración de las tareas que serán enviadas a un servidor de la nube y las tareas que se realizarán localmente, pero este buen resultado es a costa del tiempo de ejecución que puede ser muy lento. Con 100 tareas demora aproximadamente 25 segundos en realizar el calculo, pero también se debe considerar en normalmente porciones de código no poseen 100 tipos de instrucciones diferentes juntas. También se debe de considerar la configuración de los parámetros del algoritmo de Black Hole para una respuesta en un menor tiempo.

Referencias

[1] H. Shahzad and T. H. Szymanski, "A dynamic programming offloading algorithm for mobile cloud computing," in *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, May 2016, pp. 1–5.

[2] A. Hatamlou, "Black hole: A new heuristic optimization approach for data clustering," *Information Sciences*, vol. 222, pp. 175 – 184, 2013, including Special Section on New Trends in Ambient Intelligence and Bio-inspired Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025512005762