Segunda Parte del avance de Tesis.

Resumen

Expondremos las nociones y atributos de los conceptos que nos permita hacer la simulacion entre el mundo distribuido y el mundo de las maquinas de Turing.

En esta parte del documento abordaremos el problema iniciando con cierta estructura de dato que nos proporcionara el Puente entre el mundo Distribuido y el mundo De maquinas de Turing. Una vez que tenemos la maquina de Turing TM a uno de sus elementos podemos observar el atributo *Localidad* a los elementos de la cinta de la maquina de turing, que es un caracter de σ, entonces mas concretamente si tomamos una $w \in L(TM)$ donde $w = w_1 \dots w_n$, entonces podemos denotar su atributo de localidad como $Loc(w_i)$, para cada $w_i \in w$ nosotros usaremos la notacion:

$$LOCATIONS_{TM}(w) = (Loc(w_1), \dots, (Loc(w_n))), \tag{1}$$

Para denotar el vector de Locaciones para una cadena que esta en Lenguaje aceptado por la maquina de Turing TM i.e para cada elemento de $w \in TM$

Por otro lado podemos y de manera Analoga definir la nocion de localidad para la grafica del Mundo Distribuido, de la siguiente manera: para cada proceso p en $V(G)$, le notaremos el atributo de Localidad como $Loc(p)$

$$LOCATIONS_{DC}(G) = (Loc(p_1), \dots, (Loc(p_n))) \tag{2}$$

para denotar el vector de Localidades asignados a los procesos de $p_i \in G$
Podemos denotar la siguiente estructura de dato con esos dos atributos en ambos mundos, es decir cada p proceso en G almacenara esta funcion de manera local:

$$Location\_Function : LOCATIONS_{TM} \rightarrow LOCATIONS_{DC} \tag{3}$$

En particular podemos tener la funcion coordenada de la siguiente manera:

$$Location\_Function_j(Loc_{TM}(w_j)) = Loc_{DC}(v_i) \tag{4}$$

Donde w_i es parte de la cadena total que esta comiendo el Algoritmo, y en particular la maquina de Turing TM y el mapeo de locación, que en efecto hace el puente entre el atributo de los objetos entre ambos modelos que es lo que nos permitira solucionar el problema

Diseño del Algoritmo: Sea un proceso p que esta de manera local consumiendo w_j un pedazo de la cadena, y la funcion de Localidad que es una estructura de dato y ademas tenemos como subrutina a la funcion δ, entonces lo que hara el proceso en una descripcion a un alto nivel, es hacer una llamada recursiva de δ hasta que la funcion $Location\_Function_j(Location(w_j)) \neq loc(p)$ i.e hasta que la cadena que consume la subrutina delta no pertenzca a la localidad de p, es decir que la funcion $Loc\_Function$ sirve como un identificador de las localidades de los caracteres de w con respecto a la localidad de p.

Hagamos el Pseudocodigo en las siguientes lineas:

Algorithm 1 $\pi(w, G)$

init $w = (w_1 \dots w_n)$
sync
for all r **to** m **do**
    init $q_0 = q$ {Inicializacion el Estado Cero}
    init w_j {Inicializacion de la cinta en w_j}
    init $p_i(w_j)$ {El proceso hace una macros de Lectura del pedazo de la cadena que esta comiendo el proceso en cuestion}
    repeat
        call $\delta(q, w_j)$
    until $Location\_Function_j(Loc_{TM}(w_j)) \neq Loc_{DC}(p_i)$
    finally
    if $q_{final} = q_{accept}$ **then**
        return q_{accept}
    else
        send $Msg \leftarrow < q_{final}, loc_{final} >$ {Where Loc_{final} is the output of the recursive call}
    end if
end for

Ejemplo

Una vez que tenemos de manera Abstracta, Podemos exponer de manera concreta la siguiente situacion: tomar una linea de Procesos, de tal manera que el pariente de i es $i+1$, para i en el conjunto de indices de los procesos de $V(G)$, y mas aun el pariente de n es 1. Entonces lo que podemos observar es que el mapeo $Location\_Function$ nos da la intuicion, de lo que esta haciendo el algoritmo, ya que las localidades correspondientes a los procesos $loc(p_j)$ es su correspondiente $loc(w_j)$, entonces una vez que tenemos las llamadas recursivas, los procesos estaran en **wait** o **send** con la entrada que coman y reciban, hasta que la subrutina delta nos escupa la localidad correspondiente de la cadena.