ESCUELA DE CIENCIAS DE LA
COMPUTACIÓN E INFORMÁTICA
UNIVERSIDAD DE COSTA RICA

Proyecto con Python / MPI4PY

Objetivos

Que el estudiante profundice su aprendizaje sobre:

1. El diseño de programas paralelos, eficientes y “escalables” basados en procesos y memoria distribuida usando Python y MPI4PY.
2. La implementación de un programa que mezcla paralelización por datos y tareas.
3. La depuración sistemática de un programa paralelo.
4. La evaluación del desempeño de un programa paralelo utilizando las métricas: “tiempo pared”, “aceleración” y “eficiencia” para mejorarlo.

Descripción del problema

Se deberá elaborar un programa paralelo usando Python y MPI4PY que realice una simulación de un proceso de infección de un virus en un grupo de personas basado en el modelo SIR (https://es.wikipedia.org/wiki/Modelo_SIR) tal como se puede visualizar en el siguiente portal web:

http://netlogoweb.org/launch#http://netlogoweb.org/assets/modelslib/Sample%20Models/Biology/Virus.nlogo.

Los parámetros de entrada de la simulación son (]..] intervalo semi-abierto):

1. Cantidad de personas cpr: int]0..10,000,000].
2. Potencia infecciosa del virus piv: float]0..1[.
3. Probabilidad de recuperación de jóvenes (de 0 a 64 años) prj: float]0..1[.
4. Probabilidad de recuperación de mayores (de 65 a 100 años) prm: float]0..1[.
5. Cantidad de personas originalmente infectadas poi: int]0..100].
6. Tasa de ocupación toc: float]0..1].
7. Duración mínima de la enfermedad (en “tics” que son días) dmn: int]0..100].
8. Duración máxima de la enfermedad dmx: int]0..100] y dmn < dmx.
9. Radio de movilidad de jóvenes rmj: int[0..10].
10. Radio de movilidad de mayores rmm: int[0..10].
11. Velocidad de movimiento de jóvenes por tic vmj: int[0..5].
12. Velocidad de movimiento de mayores por tic vmm: int[0..5].
13. Duración de la simulación en días drc:]0..10000].

Correspondencias con el modelo en línea:

1. cantidad de personas corresponde con “number-people”,
2. potencia infecciosa corresponde con “infectiousness”,
3. probabilidad de recuperación con “chance-recover”, lo que implica que la probabilidad de muerte es 1 - (probabilidad de recuperación).
4. cantidad de personas originalmente infectada no aparece.
5. tasa de ocupación: no aparece pero determinará el tamaño de la matriz con base en la cantidad de personas.
7. la duración se mide en semanas en el modelo en línea, en el nuestro se medirá en días; además usaremos un rango de duración dmn a dmx, en lugar de un valor fijo.

Los resultados o salida de la simulación deben ser:

1. Promedio por tic, porcentaje y cantidad actual de personas Susceptibles.
2. Promedio por tic, porcentaje y cantidad actual de personas Infectadas.
3. Promedio por tic, porcentaje y cantidad actual de personas Resistentes.
4. Promedio por tic, porcentaje y cantidad actual de personas muertas.
5. Promedio final de porcentaje y cantidad de infectados, susceptibles, recuperados y muertos.

Los supuestos son que:

1. Las personas sólo se pueden morir por la infección.
2. No nacen personas nuevas.
3. Con base en cpr y toc el tamaño de la matriz cuadrada se aproxima según la fórmula: tmm = math.floor(math.sqrt(cpr/toc))+1.
4. **NO NECESARIAMENTE tmm % MPI.size == 0. Ni cpr % MPI.size == 0.**
5. SI el radio de movilidad es cero, se interpreta que la persona está libre y puede moverse por toda la matriz.
6. El 90.45% de cpr son jóvenes, y los demás (9.55% son mayores).

Las reglas que rigen el comportamiento de las personas son:

1. Deambulan al azar por el espacio bidimensional, trasladándose, por tic, a una posición en cualquier dirección a partir de su posición actual y de acuerdo a su velocidad y radio de movilidad.
2. Una persona infectada sólo puede contagiar a otras que ocupen su misma posición en el mismo tic, de acuerdo con la probabilidad de infección y el estado de las personas con que comparte la posición.
3. Si coincidieran dos o más personas **infectadas**, junto con otras **no infectadas**, en una misma posición, cada persona infectada podría infectar a cada una de las no infectadas.
4. Una persona puede morir sólo estando infectada y al final de su enfermedad, o sea de la cantidad de días que dura enferma, la cual se determina al azar y oscila entre dmn y dmx.
5. Una persona recuperada gana inmunidad y no infecta a ninguna otra durante una cantidad de días variable en el rango: [200,480], al cabo del cual vuelve a ser Susceptible.

El programa divide el procesamiento con base en los datos (paralelización por datos):

1. p0:
1.1 recibe y valida los parámetros por línea de consola,
1.2 procesa su parte de la matriz que le corresponde,
1.3 genera el archivo con los resultados por cada “tic” y finales,
1.4 genera el gráfico con las curvas correspondes.
2. Todos los procesos se dividen los datos lo más equitativamente que sea posible SIN SUPONER que tmm % MPI.size == 0 ni cpr % MPI.size == 0. Cuáles son los datos que se dividen?

Criterios de evaluación

Su programa será evaluado con base en los siguientes cinco criterios básicos, simplicidad, moduralidad, eficacia, eficiencia y escalabilidad:

- Se entiende por “simplicidad” que el programa sea lo más fácil de entender, de depurar y de modificar que sea posible.
- Se entiende por “modularidad” la correcta separación de funciones entre el código de los distintos “módulos”. Un módulo es una clase, una función o grupo de funciones, y la función principal “main()”. Por ejemplo, con una adecuada modularidad el “main()” se encarga de la entrada y salida de datos, así como la generación de mensajes por consola dirigidos al usuario. Las clases NO deben encargarse de las funciones del “main”.
- Se entiende por “eficacia” que el programa cumpla con el objetivo de simular el proceso infeccioso correctamente con base en las reglas anteriores.
- Se entiende por “eficiencia” que el programa cumpla con el objetivo en el menor “tiempo pared” posible.
- Se entiende por “escalabilidad” que:
  - el código se adapte automáticamente mediante la función “comm.size()” a la cantidad de procesos indicada por el usuario. Suponga que la cantidad de procesos por núcleo oscila en el rango de 2 a 5, parte de su trabajo será encontrar el óptimo para su programa,
  - la “eficiencia” mejora cuando se agregan más procesos, aunque en algún momento se cumpla la ley de Amdahl.

La evaluación de la eficiencia y escalabilidad será **comparativa**, lo que significa que su puntaje se establecerá en comparación con los demás trabajos presentados en el grupo:

1. La eficiencia se medirá en términos de qué tan rápido es su programa con base en la comparación del “tiempo pared” en Kabrè con los demás trabajos de su grupo.
2. La escalabilidad se medirá en términos de si su programa mejora significativamente (aceleración o “speedup”) cuando se incrementa la cantidad de procesos que variará en {8, 16}.

Dado que hay muchos factores que intervienen en cada ejecución de un programa, la única forma de comparar las mediciones o métricas especificadas de un programa con las de otros es mediante promedios simples: “promedio de eficacia”, “promedio de tiempo pared” y “promedio de aceleración”. Por tanto usted deberá adjuntar a la entrega de su trabajo un archivo pdf con la siguiente tabla a efecto de que su programa pueda ser debidamente comparado con los demás.

8 procesadores (? prc x ncl)		**16 procesadores (? prc x ncl)**			
cp == 1000000	cp == 5000000	cp == 1000000		cp == 5000000	
tp_{11}	tp_{12}	tp_{21}	ac_1	tp_{22}	ac_2
de-tp_{11}	de-tp_{12}	de-tp_{21}	de-ac_1	de-tp_{22}	de-ac_2

Donde (todos son promedios en segundos):

- tp_{1i} == es el promedio simple de los “tiempo pared” en diez ejecuciones con 8 procesadores, se deberá usar segundos como unidad de tiempo,
- tp_{2i} == es el promedio simple de los “tiempo pared” en diez ejecuciones con 16 procesadores,
- ac_i == tp_{1i} / tp_{2i}, que representa el promedio de las aceleraciones en diez ejecuciones al pasar de 8 a 16 procesadores.
- La última fila representa las desviaciones estándar de tp y ac.

En la tabla anterior no se varía la cantidad de procesos por núcleo porque se supone que usted encontrará la cantidad óptima para su programa y realizará el reporte basándose en esa cantidad que deberá aparecer al final del mismo.

El asistente a cargo validará la tabla aportada y asignará puntaje[1] a cada trabajo, en los rubros de eficiencia y escalabilidad, ordenando la lista de valores correspondiente de menor a mayor y luego dividiéndola en mínimo dos (si la desv-st es pequeña) y máximo cuatro grupos (si la desv-st es grande) con resultados similares, luego asignará:

9 o 10: para los mejores promedios,
7 u 8: para los promedios buenos,
6: para los promedios regulares, y
menos de 6: para los promedios malos.

Tabla de Evaluación

Criterio	%
Simplicidad	5
Modularidad	5
Eficacia	35
Eficiencia	35
Escalabilidad	20
Hasta 10 puntos por un reporte de errores completo **en caso que no haya funcionado** el programa	

Notas importantes:
1. Si el programa no funciona, es decir la simulación es incorrecta, no obtendrá más de 5/100 puntos, o 15/100 si incluye un reporte de errores completo. Obtendrá cero puntos en todos los rubros.
2. Este proyecto deberá realizarse idealmente y a lo más en parejas. NO SE ACEPTARÁ NINGÚN TRABAJO ELABORADO POR MÁS DE DOS PERSONAS.
3. Cada hora de atraso en la entrega se penalizará con -1/100, lo que se aplicará a la nota obtenida.
4. A TODOS LOS ESTUDIANTES INVOLUCRADOS EN UN FRAUDE SE LES APLICARÁ EL ARTÍCULO #5 INCISO C DEL "Reglamento de Orden y Disciplina de los Estudiantes de la Universidad de Costa Rica".
5. NO SUBA ningún otro archivo que no sea de código fuente (*.py) o de datos para evitar la transmisión de virus.

1 Para validar los datos el asistente realizará al menos una prueba con cp== 1,000,000.