Pregunta 1

i) func_ASM y func2_c representan una mejora significativa ya que se obtienen menores tiempos comparados a la funcion func_c.

j) Otro posible cuello de botella seria la funcion que genera los numeros de fibonacci, se podria aumentar el speedup si optimizaramos esa funcion por ej, pasarla a assembly en vez de C, etc.

Pregunta 2

a) Esto se debe a que a medida que se agregan mas procesadores, el trabajo total que se puede hacer de manera en multiples procesadores se distribuye entre todos por lo que mientras mas procesadores hay menos trabajo va a hacer cada uno hasta que llega a 0, pero cuando p → infinito, todavia queda la parte del programa que no se puede dividir en procesadores. Con Gustafson-Barsis no pasa eso ya que a medida que se agregan procesadores, el trabajo total tambien debe aumentar, por lo que cada procesador siempre va a hacer la misma cantidad de trabajo.

b) Para poder limitar a Gustafson-Barsis podemos utilizar el overhead que causa tener muchos procesadores debido a la sincronizacion y dependencia de datos.

c) Supongamos que quiera analizar el performance de una aplicacion que calcula la data de un trabajo de aerodinamica. Segun la ley de Amdahl, el speedup maximo que podria obtener al agregar procesadores seria $1 / (f + (1 – f) / p)$ siendo f la parte serial del programa y el tiempo podria reducirse. Si yo quisiera dejar a una pc haciendo el calculo mientras me voy a comer, el tiempo seria constante por lo que podria darle mas informacion al programa para que se tome el tiempo que me toma comer para poder obtener resultados mas exactos, esta seria la ley de Gustafson-Barsis.