N° 1
Identificar el problema Lentitud en el rastreo de personas
Medición Ubicación: Clase “World” 66 - 71 Haciendo 8 benchmark obtengo un promedio de 2,4357669 segundos
Hipótesis Hay un problema de redundancia en el update de las personas, ya que las personas sanas buscan comprobar si hay alguien cerca y las infectadas lo mismo. Con que las infectadas o las sanas realicen el trabajo se reduce el trabajo a la mitad.

Descripción de los cambios
Elimino la busqueda en el caso de que esten sanos
`if (near.Any(p => p.Infected))`
`{`
`  Infected = true;`
`}`
Hago que busquen a los cercanos los infectados
`IEnumerable<Person> near = world.ObjectsAt(Position).Cast<Person>();`

Nueva medición Haciendo 8 benchmark obtengo un promedio de 0,0055726 segundos

N° 2
Identificar el problema Con cada infectado disminuye la velocidad del programa
Medición Ubicación “Form1” 63 - 67 1,21639430833333 : 0,6057424 1,22203013833333 : 0,7987361 1,22749820833333 : 0,7897128 1,23288782 : 0,7706408 1,23793861833333 : 0,865211 1,24318583 : 0,8726623 1,24814466166667 : 0,9575213 1,25322763 : 1,0670827 1,257957805 : 1,3406694
Hipótesis Si almaceno a los sanos en una lista aparte debería disminuir la búsqueda a medida que más gente busca. Por lo que en teoría debería mantenerse estable de forma constante.
Descripción de cambios Cree una lista en el mundo llamada “victimas”. Cree un método nuevo (que almacena a los sanos), que lo llama el update del mundo. Hice que el método “ObjectsAt” lea a las víctimas en lugar de a el “GameObjects”.
Nueva medicion Ubicación “Form1” 63 - 67 1,12217620666667 : 0,4583134 1,12974538166667 : 0,4165489 1,137745485 : 0,4330951 1,14544538166667 : 0,4240177 1,152670455 : 0,3925488 1,15982298833333 : 0,386504

1,16665184833333 : 0,3713004
1,17345528166667 : 0,369584
1,17993278 : 0,3507635